ConceptGraphs: Open-Vocabulary 3D Scene Graphs for Perception and Planning

Anonymous Author(s)
Affiliation
Address
email

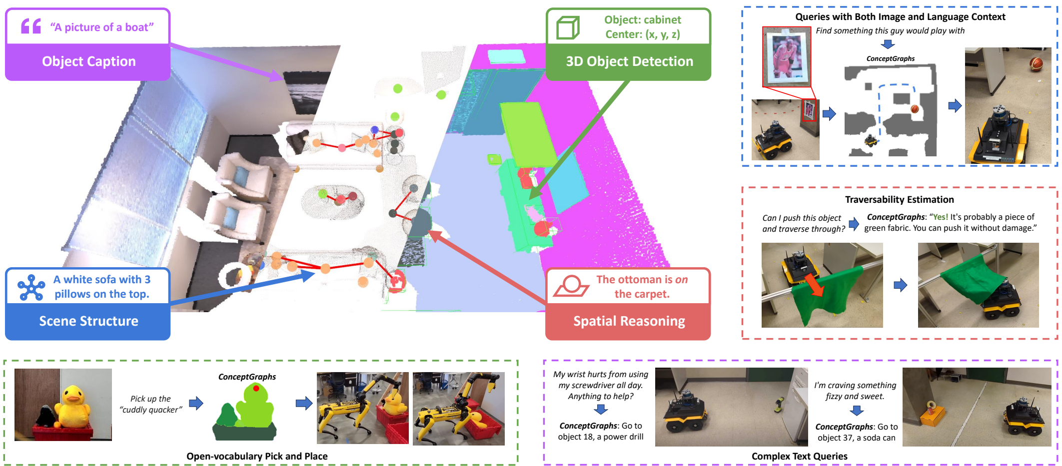

Figure 1: ***ConceptGraphs*** builds open-vocabulary 3D scene graphs. We design an object-based mapping system that (a) only assumes class-agnostic instance masks and fuses them to 3D, (b) extracts language tags for each mapped instance leveraging large vision-language models, and (c) builds a graph of object spatial relations. The object-centric nature of *ConceptGraphs* allows easy map maintenance and promotes scalability, and the graph structure provides relational information within the scene. Furthermore, our scene graph representations are easily mapped to natural language formats to interface with LLMs, enabling them to answer complex scene queries and granting robots access to useful facts about surrounding objects, such as traversability and utility. We implement and demonstrate *ConceptGraphs* on a number of real-world robotics tasks across wheeled and legged mobile robot platforms. (Anonymized project page)

Abstract: For robots to perform a wide variety of tasks, they require a 3D representation of the world that is semantically rich, yet compact and efficient for task-driven perception and planning. Recent approaches have attempted to leverage features from large vision-language models to encode semantics in 3D representations. However, these approaches tend to produce maps with per-point feature vectors, which do not scale well in larger environments, nor do they contain semantic spatial relationships between entities in the environment, which are useful for downstream planning. In this work, we propose *ConceptGraphs*, an open-vocabulary graph-structured representation for 3D scenes. *ConceptGraphs* is built by leveraging 2D foundation models and fusing their output to 3D by multi-view association. The resulting representations generalize to novel semantic classes, without the need to collect large 3D datasets or finetune models. We demonstrate the utility of this representation through a number of downstream planning tasks that are specified through abstract (language) prompts and require complex reasoning over spatial and semantic concepts. To explore the full scope of our experiments and results, we encourage visiting our anonymized project webpage.

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

1 Introduction

Scene representation is one of the key design choices that can facilitate downstream planning for a variety of tasks, including mobility and manipulation. Robots need to build these representations online from onboard sensors as they navigate through an environment. For efficient execution of complex tasks such representations should be: *scalable and efficient to maintain*, as the volume of the scene and the duration of the robot's operation increases; *open-vocabulary*, not limited to making inferences about a set of concepts that is predefined at training time, but capable of handling new objects and concepts at inference time; and *with a flexible level of detail* to enable planning over a range of tasks, from ones that require dense geometric information for mobility and manipulation, to ones that need abstract semantic information and object-level affordance information for task planning. We propose ***ConceptGraphs***, a 3D scene representation method for robot perception and planning that satisfies all the above requirements.

Closed-vocabulary semantic mapping in 3D. Early works reconstruct the 3D map through online algorithms like simultaneous localization and mapping (SLAM) [1, 2, 3, 4, 5] or offline methods like structure-from-motion (SfM) [6, 7]. Aside from reconstructing 3D geometry, recent works also use deep learning-based object detection and segmentation models to reconstruct the 3D scene representations with dense semantic mapping [8, 9, 10, 11] or object-level decomposition [12, 13, 14, 15]. While these methods achieve impressive results in mapping semantic information to 3D, they are closed-vocabulary and their applicability is limited to object categories annotated in their training datasets.

3D scene representations using foundation models. There have been significant recent efforts [16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30] focused on building 3D representations by leveraging *foundation models* - large, powerful models that capture a diverse set of concepts and accomplish a wide range of tasks [31, 32, 33, 34, 35]. Such models have excelled in tackling open-vocabulary challenges in 2D vision. However, they require an "internet-scale" of training data, and no 3D datasets exist yet of a comparable size. Recent works have therefore attempted to *ground* the 2D representations produced by image and language foundation models to the 3D world and show impressive results on open-vocabulary tasks, including language-guided object grounding [17, 18, 24, 26, 36], 3D reasoning [37, 38], robot manipulation [39, 40] and navigation [41, 42]. These approaches project dense per-pixel features from images to 3D to build explicit representations such as pointclouds [17, 18, 19, 20, 21] or implicit neural representations [16, 22, 23, 24, 25, 26, 27, 28, 29, 30].

However, such methods have two key limitations. First, assigning every point a semantic feature vector is highly redundant and consumes more memory than necessary, greatly limiting scalability to large scenes. Second, these dense representations do not admit an easy decomposition – this lack of structure makes them less amenable to dynamic updates to the map (crucial for robotics).

3D scene graphs. 3D scene graphs (3DSGs) address the second limitation by compactly and efficiently describing scenes with graph structures, with nodes representing objects and edges encoding inter-object relationships [43, 44, 45, 46, 47]. These approaches have enabled building real-time systems that can dynamically build up hierarchical 3D scene representations [48, 49, 50], and more recently shown that various robotics planning tasks can benefit from efficiency and compactness of 3DSGs [51, 52]. However, existing work on building 3D scene graphs has been confined to the closed-vocabulary setting, limiting their applicability to a small set of tasks.

1.1 Overview of Our Contribution

In this work, we mitigate all the aforementioned limitations and propose ***ConceptGraphs***, an open-vocabulary and object-centric 3D representation for robot perception and planning. In *ConceptGraphs*, each object is represented as a node with geometric and semantic features, and relationships among objects are encoded in the graph edges. At the core of *ConceptGraphs* is an object-centric 3D mapping technique that integrates geometric cues from conventional 3D mapping sys-

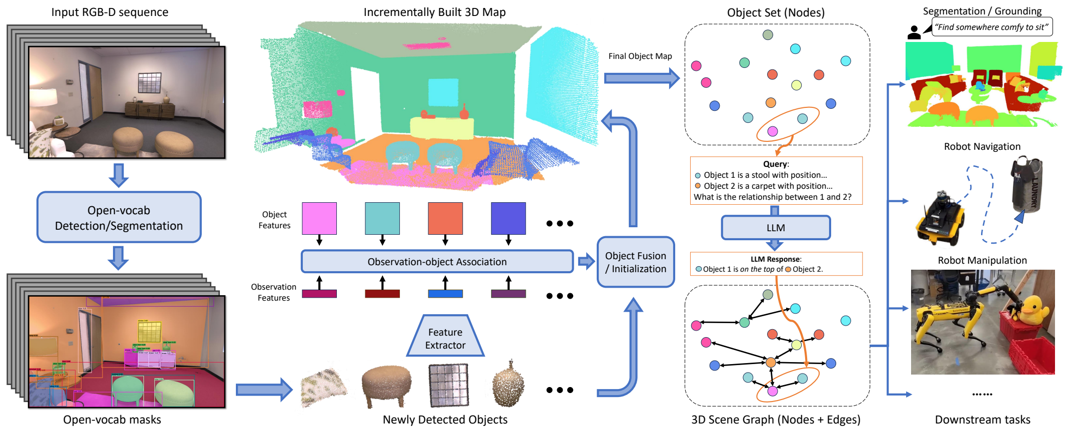

Figure 2: *ConceptGraphs* builds an open-vocabulary 3D scene graph from a sequence of posed RGB-D images. We use generic instance segmentation models to segment regions from RGB images, extract semantic feature vectors for each, and project them to a 3D point cloud. These regions are incrementally associated and fused from multiple views, resulting in a set of 3D objects and associated vision (and language) descriptors. Then large vision and language models are used to caption each mapped 3D objects and derive inter-object relations, which generates the edges to connect the set of objects and form a graph. The resulting 3D scene graph provides a structured and comprehensive understanding of the scene and can further be easily translated to a text description, useful for LLM-based task planning.

tems, and semantic cues from vision and language foundation models [31, 33, 34, 53, 54, 55]. Objects are assigned language tags by leveraging large language models (LLMs) [32] and large vision-language models (LVLMs) [55], which provide semantically rich descriptions and enable free-form language querying, all while using off-the-shelf models (no training/finetuning). The 3D scene graph [43, 44, 45, 46, 47] structure allows us to efficiently represent large scenes with a low memory footprint and makes for efficient task planning.

In experiments, we demonstrate that *ConceptGraphs* is able to discover, map, and caption a large number of objects in a scene. Further, we conduct real-world trials on multiple robot platforms over a wide range of downstream tasks, including manipulation, navigation, localization, and map updates. To summarize, our key **contributions** are:

- We propose a novel object-centric mapping system that integrates geometric cues from traditional 3D mapping systems and semantic cues from 2D foundation models.
- We construct open-vocabulary 3D scene graphs; efficient and structured semantic abstractions for perception and planning.
- We implement *ConceptGraphs* on real-world wheeled and legged robotic platforms, demonstrating downstream perception and planning for complex/abstract language queries.

2 Method

ConceptGraphs builds a compact, semantically rich representation of a 3D environment. Given a set of posed RGB-D frames, we run a class-agnostic segmentation model to obtain candidate objects, associate them across multiple views using geometric and semantic similarity measures, and instantiate nodes in a 3D scene graph. We then use an LVLM to caption each node and an LLM to infer relationships between adjoining nodes, which results in edges in the scene graph. This resultant scene graph is open-vocabulary, encapsulates object properties, and can be used for a multitude of downstream tasks including segmentation, object grounding, navigation, manipulation, localization, and remapping. The approach is illustrated in Fig. 2. For implementation details, see appendix

2.1 Object-based 3D Mapping

Object-centric 3D representation: Given a sequence of RGB-D observations $\mathcal{I} = \{I_1, I_2, \ldots, I_t\}$, *ConceptGraphs* constructs a map, a 3D scene graph, $\mathcal{M}_t = \langle \mathbf{O}_t, \mathbf{E}_t \rangle$, where $\mathbf{O}_t = \{\mathbf{o}_j\}_{j=1 \ldots J}$ and $\mathbf{E}_t = \{\mathbf{e}_k\}_{k=1 \ldots K}$ represent the sets of objects and edges, respectively. Each object $\mathbf{o}_j$ is characterized by a 3D point cloud $\mathbf{p}_{\mathbf{o}j}$ and a semantic feature vector $\mathbf{f}_{\mathbf{o}j}$. This map is built incrementally, incorporating each incoming frame $I_t = \langle I_t^{\text{rgb}}, I_t^{\text{depth}}, \theta_t \rangle$ (color image, depth image, pose) into the existing object set $\mathbf{O}_{t-1}$, by either adding to existing objects or instantiating new ones.

Class-agnostic 2D Segmentation: When processing frame I_t, a class-agnostic segmentation model $\text{Seg}(\cdot)$ is used to obtain a set of masks $\{\mathbf{m}_{t,i}\}_{i=1 \ldots M} = \text{Seg}(I_t^{\text{rgb}})$ corresponding to candidate objects[1]. Each extracted mask $\mathbf{m}_{t,i}$ is then passed to a visual feature extractor (CLIP [31], DINO [53]) to obtain a visual descriptor $\mathbf{f}_{t,i} = \text{Embed}(I_t^{\text{rgb}}, \mathbf{m}_{t,i})$. Each masked region is projected to 3D, denoised using DBSCAN clustering, and transformed to the map frame. This results in a pointcloud $\mathbf{p}_{t,i}$ and its corresponding unit-normalized semantic feature vector $\mathbf{f}_{t,i}$.

Object Association: For every newly detected object $\langle \mathbf{p}_{t,i}, \mathbf{f}_{t,i} \rangle$, we compute semantic and geometric similarity with respect to all objects $\mathbf{o}_{t\text{-}1,j} = \langle \mathbf{p}_{\mathbf{o}j}, \mathbf{f}_{\mathbf{o}j} \rangle$ in the map that shares any partial geometric overlap. The geometric similarity $\phi_{\text{geo}}(i,j) = \text{nnratio}(\mathbf{p}_{t,i}, \mathbf{p}_{\mathbf{o}j})$ is the proportion of points in point cloud $\mathbf{p}_{t,i}$ that have nearest neighbors in point cloud $\mathbf{p}_{\mathbf{o}j}$, within a distance threshold of δ_{nn}. The semantic similarity $\phi_{\text{sem}}(i,j) = \mathbf{f}_{t,i}^T \mathbf{f}_{\mathbf{o}j}/2 + 1/2$ is the normalized cosine distance between the corresponding visual descriptors.[2] The overall similarity measure $\phi(i,j)$ is a sum of both: $\phi(i,j) = \phi_{\text{sem}}(i,j) + \phi_{\text{geo}}(i,j)$. We perform object association by a greedy assignment[3] strategy where each new detection is matched with an existing object with the highest similarity score. If no match is found with a similarity higher than δ_{sim}, we initialize a new object.

Object Fusion: If a detection $\mathbf{o}_{t\text{-}1,j}$ is associated with a mapped object $\mathbf{o}_j$, we fuse the detection with the map. This is achieved by updating the object semantic feature as $\mathbf{f}_{\mathbf{o}j} = (n_{\mathbf{o}_j} \mathbf{f}_{\mathbf{o}j} + \mathbf{f}_{t,i})/(n_{\mathbf{o}_j} + 1)$, where $n_{\mathbf{o}_j}$ is the number of detections that have been associated to $\mathbf{o}_j$ so far; and updating the pointcloud as $\mathbf{p}_{t,i} \cup \mathbf{p}_{\mathbf{o}j}$, followed by down-sampling to remove redundant points.

Node Captioning: Once the entire image sequence has been processed, a vision-language model, denoted $\text{LVLM}(\cdot)$, is used to generate object captions. For each object, the associated image crops from the *best*[4] 10 views are passed to the language model with the prompt "*describe the central object in the image*" to generate a set of initial rough captions $\hat{\mathbf{c}}_j = \{\hat{\mathbf{c}}_{j,1}, \hat{\mathbf{c}}_{j,2}, \ldots, \hat{\mathbf{c}}_{j,10}\}$ for each detected object $\mathbf{o}_j$. Each set of captions is then refined to the final caption by passing $\hat{\mathbf{c}}_j$ to another language model $\text{LLM}(\cdot)$ with a prompt instruction to summarize the initial captions into a coherent and accurate final caption $\mathbf{c}_j$.

2.2 Scene Graph Generation

Given the set of 3D objects $\mathbf{O}_T$ obtained from the previous step, we estimate their spatial relationships, i.e., the edges $\mathbf{E}_T$, to complete the 3D scene graph. We do this by first estimating potential connectivity among object nodes based on their spatial overlaps. We compute the 3D bounding box IoU between every pair of object nodes to obtain a similarity matrix (i.e., a dense graph), which we prune by estimating a minimum spanning tree (MST), resulting in a refined set of potential edges among the objects. To further determine the semantic relationships, for each edge in the MST, we input the information about the object pair, consisting of object captions and 3D location, to a

[1] Without loss of generality, $\text{Seg}(\cdot)$ may be replaced by open-/closed-vocabulary models to build category-specific mapping systems.

[2] For the sake of brevity, we only describe the best-performing geometric and semantic similarity measures. For an exhaustive list of alternatives, please see our project website and code.

[3] While we also experimented with optimal assignment strategies such as the Hungarian algorithm, we experimentally determined them to be slower and offer only a minuscule improvement over greedy association.

[4] We maintain a running index of the number of noise-free points each view contributes to the object point cloud.

language model LLM. The prompt instructs the model to describe the likely spatial relationship between the objects, such as "*a on b*" or "*b in a*", along with the underlying reasoning. The model outputs a relationship label with an explanation detailing the rationale. The use of an LLM allows us to extend the nominal edge type defined above to other output relationships a language model can interpret, such as "a backpack *may be stored in* a closet" and "sheets of paper *may be recycled in* a trash can". This results in an open-vocabulary 3D scene graph $\mathcal{M}_T = (\mathbf{O}_T, \mathbf{E}_T)$, a compact and efficient representation for use in downstream tasks.

2.3 Robotic Task Planning through LLMs

To enable users to carry out tasks described in natural language queries, we interface the scene graph $\mathcal{M}_T$ with an LLM. For each object in $\mathbf{O}_T$, we construct JSON-structured text descriptions that include information about its 3D location (bounding box) and its node caption. Given a text query, we task the LLM to identify the most relevant object in the scene. We then pass the 3D pose of this object to the appropriate pipeline for the downstream task (e.g., grasping, navigation). This integration of *ConceptGraphs* with an LLM is easy to implement, and enables a wide range of open-vocabulary tasks by giving robots access to the semantic properties of surrounding objects[5] (see Sec. 3).

2.4 Implementation Details

The modularity of *ConceptGraphs* enables any appropriate open/closed-vocabulary segmentation model, LLM, or LVLM to be employed. Our experiments use Segment-Anything (SAM) [33] as the segmentation model $\text{Seg}(\cdot)$, and the CLIP image encoder [31] as the feature extractor $\text{Embed}(\cdot)$. We use LLaVA [55] as the vision-language model LVLM and GPT-4 [32] (`gpt-4-0613`) for our LLM. The voxel size for point cloud downsampling and nearest neighbor threshold δ_{nn} are both 2.5cm. We use 1.1 for the association threshold δ_{sim}.

We also develop a variant of our system, *ConceptGraphs-Detector* (*CG-D*), where we employ an image tagging model (RAM [54]) to list the object classes present in the image and an open-vocabulary 2D detector (Grounding DINO [34]) to obtain object bounding boxes[6]. In this variant, we need to separately handle detected background objects (wall, ceiling, floor) by merging them regardless of their similarity scores.

3 Experiments

3.1 Scene Graph Construction

We first evaluate the accuracy of the 3D scene graphs output by the *ConceptGraphs* system in Table 1. For each scene in the Replica dataset [56], we report scene graph accuracy metrics for both *CG* and the detector-variant *CG-D*. The open-vocabulary nature of our system makes automated evaluation of the quality of nodes and edges in the scene graph challenging. We instead evaluate the scene graph by engaging human evaluators on Amazon Mechanical Turk (AMT). For each node, we compute precision as the fraction of nodes for which at least 2 of 3 human evaluators deem the node caption correct. We also report the number of valid objects retrieved by

	scene	node prec.	valid objects	duplicates	edge prec.
CG	room0	0.78	54	3	0.91
	room1	0.77	43	4	0.93
	room2	0.66	47	4	1.0
	office0	0.65	44	2	0.88
	office1	0.65	23	0	0.9
	office2	0.75	44	3	0.82
	office3	0.68	60	5	0.79
	Average	**0.71**	-	-	**0.88**
CG-D	room0	0.56	60	4	0.87
	room1	0.70	40	3	0.93
	room2	0.54	49	2	0.93
	office0	0.59	35	0	1.0
	office1	0.49	24	2	0.9
	office2	0.67	47	3	0.88
	office3	0.71	59	1	0.83
	Average	**0.61**	-	-	**0.91**

Table 1: **Accuracy of constructed scene graphs**: node precision indicates the accuracy of the label for each node (as measured by a human evaluator); valid objects is the number of human-recognizable objects (mturkers used) discovered by our system; duplicates are the number of redundant detections; edge precision indicates the accuracy of each estimated spatial relationship (again, as evaluated by an mturker)

[5] For large scenes where the description length of the scene graph exceeds the context length of the LLM, one can easily substitute in alternative (concurrent) LLM planners [52].

[6] We discard the (often noisy) *tags* produced by the image tagging model, relying instead on our node captions.

each variant by asking evaluators whether they deem each node a valid object. Both CG and CG-D identify a number of valid objects in each scene, and incur only a small number (0-5) of duplicate detections. The node labels are accurate about 70% of the time; most of the errors are incurred due to errors made by the LVLM employed (LLaVA [55]). The edges (spatial relationships) are labeled with a high degree of accuracy (90% on average).

3.2 3D Semantic Segmentation

ConceptGraphs focuses on the construction of the open-vocabulary 3D scene graphs for scene understanding and planning. For completeness, in this section, we also use an open-vocabulary 3D semantic segmentation task to evaluate the quality of the obtained 3D maps. To generate the semantic segmentation, given a set of class names, we compute the similarity between the fused semantic feature of each object node and the CLIP text embeddings of the phrase `an image of {class}`. Then the points associated with each object are assigned to the class with the highest similarity, which gives a point cloud with dense class labels. In Table 2, we report the semantic segmentation results on the Replica [56] dataset, following the evaluation protocol used in ConceptFusion [17]. We also provide an additional baseline, ConceptFusion+SAM, by replacing the Mask2Former used in ConceptFusion with the more performant SAM [33] model. As shown in Table 2, the proposed *ConceptGraphs* performs comparably with or better than ConceptFusion, which has a much larger memory footprint.

	Method	mAcc	F-mIoU
Privileged	CLIPSeg (rd64-uni) [57]	28.21	39.84
	LSeg [58]	33.39	51.54
	OpenSeg [59]	41.19	53.74
Zero-shot	MaskCLIP [60]	4.53	0.94
	Mask2former + Global CLIP feat	10.42	13.11
	ConceptFusion [17]	24.16	31.31
	ConceptFusion [17] + SAM [33]	31.53	**38.70**
	ConceptGraphs (Ours)	**40.63**	35.95
	ConceptGraphs-Detector (Ours)	38.72	35.82

Table 2: Open-vocabulary semantic segmentation on the Replica [56] dataset. **Privileged** methods specifically finetune the pretrained models for semantic segmentation. **Zero-shot** approaches do not need any finetuning and are evaluated off the shelf.

3.3 Object Retrieval based on Text Queries

We assess the capability of *ConceptGraphs* to handle complex semantic queries, focusing on three key types.

- Descriptive: E.g., *A potted plant*.
- Affordance: E.g., *Something to use for temporarily securing a broken zipper*.
- Negation: E.g., *Something to drink but not soda*.

Dataset	Query Type	Model	R@1	R@2	R@3	# Queries
Replica	Descriptive	CLIP	0.59	0.82	0.86	20
		LLM	0.61	0.64	0.64	
	Affordance	CLIP	0.43	0.57	0.63	5
		LLM	0.57	0.63	0.66	
	Negation	CLIP	0.26	0.60	0.71	5
		LLM	0.80	0.89	0.97	
Lab	Descriptive	CLIP	1.00	–	–	10
		LLM	1.00	–	–	
	Affordance	CLIP	0.40	0.60	0.60	10
		LLM	1.00	–	–	
	Negation	CLIP	0.00	–	–	10
		LLM	1.00	–	–	

Table 3: Object retrieval from text queries on the Replica and REAL Lab scenes. We measure the top-1, top-2, and top-3 recall. CLIP refers to object retrieval using cosine similarity, whereas LLM refers to having an LLM parse the scene graph and return the most relevant object.

We evaluate on the Replica dataset [56] and a real-world scan of a lab, where we staged a number of items including clothes, tools, and toys. For Replica, human evaluators on AMT annotate captions for SAM mask proposals, which serve as both ground truth labels and descriptive queries. We created 5 affordance and negation queries for each scene type (office & room) in Replica and 10 queries of each type for the lab scan, ensuring that each query corresponds to at least one relevant object. We manually select relevant objects as ground truth for each query.

We use two object retrieval strategies: CLIP-based and LLM-based. CLIP selects the object with the highest similarity to the query's embedding, while the LLM goes through the scenegraph nodes to identify the object with the most relevant caption. Table 3 shows that CLIP excels with descriptive queries but struggles with complex affordance and negation queries [61]. For example, CLIP inaccurately retrieves a backpack for the broken zipper query, whereas the LLM correctly identifies a roll of tape. The LLM performs well across the board, but is limited by the accuracy of the node captions, as discussed in Section 3.1. Since the lab has a larger variety of objects to choose from, the LLM finds compatible objects for complex queries more reliably there.

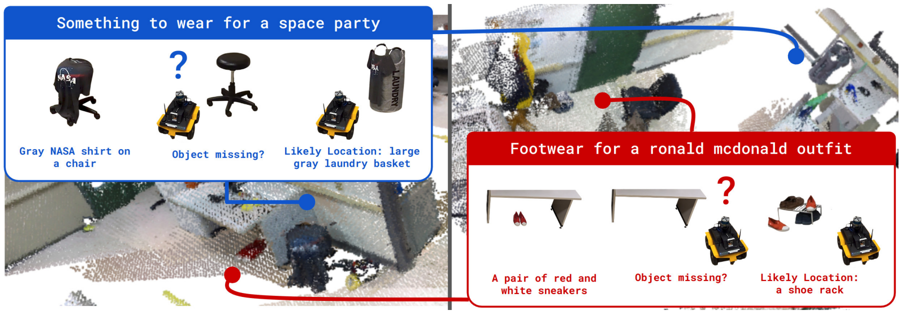

Figure 4: A Jackal robot answering user queries using the *ConceptGraphs* representation of a lab environment. We first query an LLM to identify the most relevant object given the user query, then validate with an LVLM if the target object if is at the expected location. If not, we query the LLM again to find a likely location or container for the missing object. (Blue) When prompted with `something to wear for a space party`, the Jackal attempts to find a grey shirt with a NASA logo. After failing to detect the shirt at the expected location, the LLM reasons that it could likely be in the laundry bag. (Red) The Jackal searches for red and white sneakers after receiving the user query `footwear for a Ronald McDonald outfit`. The LLM redirects the robot to a shoe rack after failing to detect the sneakers at their initial mapped potision.

3.4 Complex Visual-Language Queries

To assess the performance of *ConceptGraphs* in a real-world environment, we carry out navigation experiments in a lab environment with a Clearpath Jackal UGV. The robot is equipped with a VLP-16 LiDAR and a forward-facing Realsense D435i camera.

The Jackal needs to respond to abstract user queries and navigate to the most relevant object (Figure 1). By using an LVLM [55] to add a description of the current camera image to the text prompt, the robot can also answer visual queries. For example, when shown a picture of Michael Jordan and prompted with `Something this guy would play with`, the robot finds a basketball.

3.5 Object Search and Traversability Estimation

In this section, we showcase how the interaction between the *ConceptGraphs* representation and an LLM can enable a mobile robot to access a vast knowledge base of everyday objects. Specifically, we prompt an LLM to infer two additional object properties from *ConceptGraphs* captions: i) the location where a given object is typically found, and ii) if the object can be safely pushed or traversed by the Jackal robot. We design two tasks around the LLM predictions.

Object search: The robot receives an abstract user query and must navigate to the most relevant object in the *ConceptGraphs* map. Using an LVLM [55], the robot then checks if the object is at the expected location. If not, it queries an LLM to find a new plausible location given the captions of the other objects in the representation. In our prompt, we nudge the LLM to consider typical containers or storage locations. We illustrate two such queries where the target object is moved in Figure 4.

Traversability estimation: As shown in Fig. 3, we design a real-world scenario where the robot finds itself enclaved by objects. In this scenario, the robot must push around multiple objects and create a path to the goal state. While traversability can be learned through experience [62], we show that grounding LLM knowledge in a 3D map can grant similar capabilities to robotic agents.

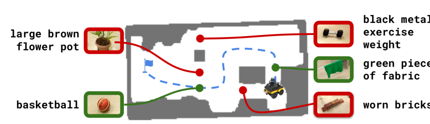

Figure 3: The Jackal robot solving a traversability challenge. All paths to the goal are obstructed by objects. We query an LLM to identify which objects can be safely pushed or traversed by the robot (green) and which objects would be too heavy or hinder the robot's movement (red). The LLM relies on the *ConceptGraphs* node captions to make traversability predictions and we add the non-traversable objects to the Jackal costmap for path planning. The Jackal successfully reaches the goal by going through a curtain and pushing a basketball, while also avoiding contact with bricks, an iron dumbbell, and a flower pot.

3.6 Open-Vocabulary Pick and Place

To illustrate how *ConceptGraphs* can act as the perception backbone for open-vocabulary mobile manipulation, we conducted a series of experiments with a Boston Dynamics Spot Arm robot. Using an onboard RGBD camera and a *ConceptGraphs* representation of the scene, the Spot robot responds to the query `cuddly quacker` by grabbing a duck plush toy and placing it in a nearby box (Figure 1). In the supplementary video, Spot completes a similar grasping maneuver with a mango when prompted with the query `something healthy to eat`.

3.7 Localization and Map Updates

ConceptGraphs can also be used for object-based localization and map updates. We showcase this with a 3-DoF (x, y and yaw) localization and remapping task in the AI2Thor [63, 64] simulation environment, where a mobile robot uses a particle filter to localize in a pre-built *ConceptGraphs* map of the environment. During the observation update step of particle filtering, the robot's detections are matched against the objects in the map based on the hypothesized pose, in a similar way as described in Section 2.1. Refer to our video explainer for a demonstration.

3.8 Limitations

Despite its impressive performance, *ConceptGraphs* has failure modes that remain to be addressed in future work. First, node captioning incurs errors due to the current limitations of LVLMs like LLaVA [55]. Second, our 3D scene graph occasionally misses small or thin objects and makes duplicate detections. This impacts downstream planning, particularly when the incorrect detection is crucial to planning success. Additionally, the computational and economic costs of our system include multiple LVLM (LLaVA [55]) and one or more proprietary LLM inference(s) when building and querying the scenegraph, which may be significant.

4 Concurrent Work

We briefly review recent and unpublished pre-prints that are exploring themes related to open-vocabulary object-based factorization of 3D scenes. Concurrently to us, [65, 66] have explored open-vocabulary object-based factorization of 3D scenes. Where [65] assumes a pre-built point cloud map of the scene, [66] builds a map on the go. Both approaches associate CLIP descriptors to the reconstruction, resulting in performance comparable to our system's CLIP variant, which struggles with queries involving complex affordances and negation, as shown in Table 3. OGSV [67] is closer to our setting, building an open-vocabulary 3D scene graph from RGB-D images. However, [67] relies on a (closed-set) graph neural network to predict object relationships; whereas *ConceptGraphs* relies on LLMs, eliminating the need to train an object relation model.

5 Conclusion

In this paper, we introduced *ConceptGraphs*, a novel approach to open-vocab object-centric 3D scene representation that addresses key limitations in the existing landscape of dense and implicit representations. Through effective integration of foundational 2D models, *ConceptGraphs* significantly mitigates memory constraints, provides relational information among objects, and allows for dynamic updates to the scene—three pervasive challenges in current methods. Experimental evidence underscores *ConceptGraphs*' robustness and extensibility, highlighting its superiority over existing baselines for a variety of real-world tasks including manipulation and navigation. The versatility of our framework also accommodates a broad range of downstream applications, thereby opening new avenues for innovation in robot perception and planning. Future work may delve into integrating temporal dynamics into the model and assessing its performance in less structured, more challenging environments.

References

[1] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE international symposium on mixed and augmented reality*, pages 127–136. IEEE, 2011. 2

[2] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison. Elasticfusion: Dense slam without a pose graph. In *Robotics Science and Systems*, 2015. 2

[3] B. Wen, J. Tremblay, V. Blukis, S. Tyree, T. Müller, A. Evans, D. Fox, J. Kautz, and S. Birchfield. BundleSDF: Neural 6-dof tracking and 3d reconstruction of unknown objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 606–617, 2023. 2

[4] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison. iMAP: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021. 2

[5] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022. 2

[6] J. L. Schonberger and J.-M. Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 2

[7] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision*, pages 501–518. Springer, 2016. 2

[8] J. McCormac, A. Handa, A. Davison, and S. Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *IEEE International Conference on Robotics and automation (ICRA)*, pages 4628–4635. IEEE, 2017. 2

[9] M. Runz, M. Buffier, and L. Agapito. Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 10–20. IEEE, 2018. 2

[10] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. Fusion++: Volumetric object-level slam. In *international conference on 3D vision (3DV)*, pages 32–41. IEEE, 2018. 2

[11] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji. Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4205–4212. IEEE, 2019. 2

[12] J. Qian, V. Chatrath, J. Yang, J. Servos, A. P. Schoellig, and S. L. Waslander. POCD: probabilistic object-level change detection and volumetric mapping in semi-static scenes. In K. Hauser, D. A. Shell, and S. Huang, editors, *Robotics Science and Systems*, 2022. 2

[13] J. Qian, V. Chatrath, J. Servos, A. Mavrinac, W. Burgard, S. L. Waslander, and A. P. Schoellig. POV-SLAM: probabilistic object-aware variational SLAM in semi-static environments. In K. E. Bekris, K. Hauser, S. L. Herbert, and J. Yu, editors, *Robotics Science and Systems*, 2023. 2

[14] K. Li, D. DeTone, Y. F. S. Chen, M. Vo, I. Reid, H. Rezatofighi, C. Sweeney, J. Straub, and R. Newcombe. ODAM: Object detection, association, and mapping using posed rgb video. In *Proceedings of International Conference on Computer Vision*, 2021. 2

[15] M. Zins, G. Simon, and M.-O. Berger. OA-SLAM: Leveraging objects for camera relocalization in visual slam. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 720–728. IEEE, 2022. 2

[16] K. Liu, F. Zhan, J. Zhang, M. Xu, Y. Yu, A. E. Saddik, C. Theobalt, E. Xing, and S. Lu. 3d open-vocabulary segmentation with foundation models. *arXiv preprint arXiv:2305.14093*, 2023. 2

[17] K. M. Jatavallabhula, A. Kuwajerwala, Q. Gu, M. Omama, G. Iyer, S. Saryazdi, T. Chen, A. Maalouf, S. Li, N. V. Keetha, A. Tewari, J. B. Tenenbaum, C. M. de Melo, K. M. Krishna, L. Paull, F. Shkurti, and A. Torralba. ConceptFusion: Open-set multimodal 3d mapping. In *Robotics: Science and Systems*, 2023. doi:10.15607/RSS.2023.XIX.066. 2, 6

[18] S. Peng, K. Genova, C. Jiang, A. Tagliasacchi, M. Pollefeys, T. Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 815–824, 2023. 2

[19] R. Ding, J. Yang, C. Xue, W. Zhang, S. Bai, and X. Qi. Lowis3d: Language-driven open-world instance-level 3d scene understanding. *arXiv preprint arXiv:2308.00353*, 2023. 2

[20] R. Ding, J. Yang, C. Xue, W. Zhang, S. Bai, and X. Qi. PLA: Language-driven open-vocabulary 3d scene understanding. In *Proceedings of Computer Vision and Pattern Recognition*, 2023. 2

[21] J. Zhang, R. Dong, and K. Ma. CLIP-FO3D: Learning free open-world 3d scene representations from 2d dense clip. *arXiv preprint arXiv:2303.04748*, 2023. 2

[22] V. Tschernezki, I. Laina, D. Larlus, and A. Vedaldi. Neural feature fusion fields: 3d distillation of self-supervised 2d image representations. In *International Conference on 3D Vision (3DV)*. IEEE, 2022. 2

[23] S. Kobayashi, E. Matsumoto, and V. Sitzmann. Decomposing nerf for editing via feature field distillation. *Neural Information Processing Systems*, 35:23311–23330, 2022. 2

[24] N. M. M. Shafiullah, C. Paxton, L. Pinto, S. Chintala, and A. Szlam. Clip-fields: Weakly supervised semantic fields for robotic memory. In K. E. Bekris, K. Hauser, S. L. Herbert, and J. Yu, editors, *Robotics: Science and Systems*, 2023. doi:10.15607/RSS.2023.XIX.074. 2

[25] N. Tsagkas, O. Mac Aodha, and C. X. Lu. Vl-fields: Towards language-grounded neural implicit spatial representations. *arXiv preprint arXiv:2305.12427*, 2023. 2

[26] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. LERF: Language embedded radiance fields. In *International Conference on Computer Vision (ICCV)*, 2023. 2

[27] C. Huang, O. Mees, A. Zeng, and W. Burgard. Audio visual language maps for robot navigation. *arXiv preprint arXiv:2303.07522*, 2023. 2

[28] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola. Distilled feature fields enable few-shot manipulation. In *International Conference on Robot Learning*, 2023. URL `https://openreview.net/forum?id=Rb0nGIt_kh5`. 2

[29] F. Engelmann, F. Manhardt, M. Niemeyer, K. Tateno, M. Pollefeys, and F. Tombari. Open-set 3d scene segmentation with rendered novel views, 2023. 2

[30] K. Mazur, E. Sucar, and A. J. Davison. Feature-realistic neural fusion for real-time, open set scene understanding. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023. 2

[31] 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*. PMLR, 2021. 2, 3, 4, 5

[32] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 2, 3, 5

[33] 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. *Proceedings of International Conference on Computer Vision*, 2023. 2, 3, 5, 6

[34] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 2, 3, 5

[35] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 2

[36] Y. Hong, Y. Du, C. Lin, J. Tenenbaum, and C. Gan. 3d concept grounding on neural fields. *Neural Information Processing Systems*, 2022. 2

[37] Y. Hong, C. Lin, Y. Du, Z. Chen, J. B. Tenenbaum, and C. Gan. 3d concept learning and reasoning from multi-view images. In *Proceedings of Computer Vision and Pattern Recognition*, 2023. 2

[38] Y. Hong, H. Zhen, P. Chen, S. Zheng, Y. Du, Z. Chen, and C. Gan. 3d-llm: Injecting the 3d world into large language models. *Neural Information Processing Systems*, 2023. 2

[39] M. Shridhar, L. Manuelli, and D. Fox. CLIPort: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, volume 164, pages 894–906. PMLR, 2021. 2

[40] S. Sharma, A. Rashid, C. M. Kim, J. Kerr, L. Y. Chen, A. Kanazawa, and K. Goldberg. Language embedded radiance fields for zero-shot task-oriented grasping. In *International Conference on Robot Learning*, 2023. URL `https://openreview.net/forum?id=k-Fg8JDQmc`. 2

[41] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song. Clip on wheels: Zero-shot object navigation as object localization and exploration. *arXiv preprint arXiv:2203.10421*, 2022. 2

[42] D. Shah, B. Osiński, S. Levine, et al. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *International Conference on Robot Learning*. PMLR, 2023. 2

[43] M. Fisher, M. Savva, and P. Hanrahan. Characterizing structural relationships in scenes using graph kernels. *ACM Trans. Graph.*, 30(4):34, 2011. 2, 3

[44] P. Gay, J. Stuart, and A. Del Bue. Visual graphs from motion (vgfm): Scene understanding with object geometry reasoning. In *Asian Conference on Computer Vision*. Springer, 2019. 2, 3

[45] 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 *Proceedings of International Conference on Computer Vision*, October 2019. 2, 3

[46] U.-H. Kim, J.-M. Park, T.-J. Song, and J.-H. Kim. 3-d scene graph: A sparse and semantic representation of physical environments for intelligent agents. *IEEE transactions on cybernetics*, 50(12):4921–4933, 2019. 2, 3

[47] J. Wald, H. Dhamo, N. Navab, and F. Tombari. Learning 3d semantic scene graphs from 3d indoor reconstructions. In *Proceedings of Computer Vision and Pattern Recognition*, 2020. 2, 3

[48] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, and L. Carlone. Kimera: From slam to spatial perception with 3d dynamic scene graphs. *The International Journal of Robotics Research*, 40(12-14):1510–1546, 2021. 2

[49] N. Hughes, Y. Chang, and L. Carlone. Hydra: A real-time spatial perception system for 3d scene graph construction and optimization. *arXiv preprint arXiv:2201.13360*, 2022. 2

[50] S.-C. Wu, J. Wald, K. Tateno, N. Navab, and F. Tombari. Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences. In *Proceedings of Computer Vision and Pattern Recognition*, 2021. 2

[51] C. Agia, K. M. Jatavallabhula, M. Khodeir, O. Miksik, V. Vineet, M. Mukadam, L. Paull, and F. Shkurti. Taskography: Evaluating robot task planning over large 3d scene graphs. In *International Conference on Robot Learning*. PMLR, 2022. 2

[52] K. Rana, J. Abou-Chakra, S. Garg, J. Haviland, I. Reid, and N. Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable task planning. In *International Conference on Robot Learning*, 2023. URL `https://openreview.net/forum?id=wMpOMO0Ss7a`. 2, 5, 19

[53] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 3, 4

[54] Y. Zhang, X. Huang, J. Ma, Z. Li, Z. Luo, Y. Xie, Y. Qin, T. Luo, Y. Li, S. Liu, et al. Recognize anything: A strong image tagging model. *arXiv preprint arXiv:2306.03514*, 2023. 3, 5

[55] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023. 3, 5, 6, 7, 8, 13, 18, 19

[56] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou, K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. D. Nardi, M. Goesele, S. Lovegrove, and R. Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 5, 6, 14

[57] T. Lüddecke and A. Ecker. Image segmentation using text and image prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7086–7096, 2022. 6

[58] B. Li, K. Q. Weinberger, S. Belongie, V. Koltun, and R. Ranftl. Language-driven semantic segmentation. In *International Conference on Learning Representations*, 2022. 6

[59] G. Ghiasi, X. Gu, Y. Cui, and T.-Y. Lin. Scaling open-vocabulary image segmentation with image-level labels. In *European Conference on Computer Vision*, pages 540–557. Springer, 2022. 6

[60] C. Zhou, C. C. Loy, and B. Dai. Extract free dense labels from clip. In *European Conference on Computer Vision (ECCV)*, 2022. 6

[61] Y. Du, S. Li, and I. Mordatch. Compositional visual generation with energy based models. In *Neural Information Processing Systems*, 2020. 6

[62] S. Levine and D. Shah. Learning robotic navigation from experience: principles, methods and recent results. *Philosophical Transactions of the Royal Society B*, 378(1869):20210447, 2023. 7

[63] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. 8

[64] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, K. Ehsani, J. Salvador, W. Han, E. Kolve, A. Kembhavi, and R. Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems*, 35:5982–5994, 2022. 8

[65] A. Takmaz, E. Fedele, R. W. Sumner, M. Pollefeys, F. Tombari, and F. Engelmann. Openmask3d: Open-vocabulary 3d instance segmentation. *arXiv preprint arXiv:2306.13631*, 2023. 8

[66] S. Lu, H. Chang, E. P. Jing, A. Boularias, and K. Bekris. OVIR-3d: Open-vocabulary 3d instance retrieval without training on 3d data. In *International Conference on Robot Learning*, 2023. URL `https://openreview.net/forum?id=gVBvtRqU1_`. 8

[67] H. Chang, K. Boyalakuntla, S. Lu, S. Cai, E. P. Jing, S. Keskar, S. Geng, A. Abbas, L. Zhou, K. Bekris, and A. Boularious. Context-aware entity grounding with open-vocabulary 3d scene graphs. In *International Conference on Robot Learning*, 2023. URL `https://openreview.net/forum?id=cjEI5qXoT0`. 8

[68] E. Jelavic, J. Nubert, and M. Hutter. Open3d slam: Point cloud based mapping and localization for education. In *Robotic Perception and Mapping: Emerging Techniques, ICRA 2022 Workshop*, page 24. ETH Zurich, Robotic Systems Lab, 2022. 18

[69] M. Labbé and F. Michaud. Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics*, 36(2):416–446, 2019. 18

[70] Z. Sun, S. Shen, S. Cao, H. Liu, C. Li, Y. Shen, C. Gan, L.-Y. Gui, Y.-X. Wang, Y. Yang, K. Keutzer, and T. Darrell. Aligning large multimodal models with factually augmented rlhf. 2023. 19

Appendix

A1 3D Scene Graph: Generating Node Captions

Once we build an object-level map of the scene using the methodology described in Sec. 2.1, we extract and summarize captions for each object. We first extract upto the 10 *most-informative* views for each object, by tracking the number of (noise-free) 3D points that each image segment contributes to an object in the map[7]. Intuitively, these views offer the best image views for the object. We run each view through an LVLM, here LLaVA-7B [55], to generate an image caption. We use the same prompt across all images: *describe the central object in this image*.

We found the captions generated by LLaVA-7B to be incoherent or unreliable across all viewpoints. To alleviate this, we employed GPT-4 as a caption summarizer, to map all of the LLaVA-7B captions to a coherent object tag (or optionally, declare the object as an invalid detection). We use the following GPT-4 system prompt:

[7]We track these statistics throughout the mapping lifecycle; meaning that we do not impose any additional computational overhead to determine the 10 best views per object

```
Identify and describe objects in scenes.  Input and output must
be in JSON format.  The input field ’captions’ contains a list of
image captions aiming to identify objects.  Output ’summary’ as a
concise description of the identified object(s).  An object mentioned
multiple times is likely accurate.  If various objects are repeated
and a container/surface is noted such as a shelf or table, assume
the (repeated) objects are on that container/surface.  For unrelated,
non-repeating (or empty) captions, summarize as ’conflicting (or empty)
captions about [objects]’ and set ’object_tag’ to ’invalid’.  Output
’possible_tags’ listing potential object categories.  Set ’object_tag’
as the conclusive identification.  Focus on indoor object types, as the
input captions are from indoor scans.
```

Listing 1: GPT-4 system prompt used for caption summarization

A2 LLM Planner: Implementation details

For task planning over 3D scene graphs, we use GPT-4 (`gpt-4-0613`) with a context length of 8K tokens[8]. We first convert each node in the 3D scene graph into a structured text format (here, a JSON string). Each entry in the JSON list corresponds to one object in the scene, and contains the following attributes:

1. *object id*: a unique (numerical) object identifier
2. *bounding box extents*: dimensions of each side of the bounding cuboid
3. *bounding box center*: centroid of the object bounding cuboid
4. *object tag*: a brief tag describing the object
5. *caption*: a one-sentence caption (possibly encoding mode details than present in the object tag

Here is a sample snippet from the scene graph for the `room0` scene of the Replica [56] dataset.

```
[
{
    id: 2,
    bbox_extent: [2.0, 0.7, 0.6],
    bbox_center: [-0.6, 1.1, -1.2],
    object_tag: wooden dresser or chest of drawers,
    caption: A wooden dresser or chest of drawers
},
{
    id: 3,
    bbox_extent: [0.6, 0.5, 0.4],
    bbox_center: [2.8, -0.4, -0.8],
    object_tag: vase,
    caption: a white, floral-patterned vase (or possibly a ceramic bowl)
},
...
...
{
    id: 110,
    bbox_extent: [1.2, 0.6, 0.0],
    bbox_center: [2.2, 2.1, 1.2],
    object_tag: light fixture,
    caption: a light fixture hanging from the ceiling
}
]
```

Listing 2: Sample text entries in the 3D scene graph

[8]We also prototyped variants of this approach on off-the-shelf LLMs with larger context lengths, such as Claude-2 with a context length of 32K tokens, and found it to work reliably.

For planning using the above text description of the scene, we instantiate the system prompt for GPT-4 as follows:

```
The input to the model is a 3D scene described in a JSON format. Each
    entry in the JSON describes one object in the scene, with the
    following five fields:
1. "id": a unique object id
2. "bbox_extent": extents of the 3D bounding box for the object
3. "bbox_center": centroid of the 3D bounding box for the object
4. "object_tag": a brief (but sometimes inaccurate) tag categorizing the
    object
5. "caption": a brief caption for the object

Once you have parsed the JSON and are ready to answer questions about the
     scene, say "I'm ready".

The user will then begin to ask questions, and the task is to answer
    various user queries about the 3D scene. For each user question,
    respond with a JSON dictionary with the following fields:
1. "inferred_query": your interpretaion of the user query in a succinct
    form
2. "relevant_objects": list of relevant object ids for the user query (if
     applicable)
3. "query_achievable": whether or not the user-specified query is
    achievable using the objects and descriptions provided in the 3D
    scene.
4. "final_relevant_objects": A final list of objects relevant to the user
    -specified task. As much as possible, sort all objects in this list
    such that the most relevant object is listed first, followed by the
    second most relevant, and so on.
5. "explanation": A brief explanation of what the most relevant object(s)
     is(are), and how they achieve the user-specified task.
```

Listing 3: System prompt for the LLM Planner interface

Here are some example GPT-4 responses on our queries

```
User: I need a quick gift. Help!

LLM-Planner: {
inferred_query: Find suitable object for a gift.,
relevant_objects: [0, 6, 7, 23, 25, 31],
query_achievable: true,
final_relevant_objects: [6],
explanation: The most suitable object for a gift could be the ceramic
    vase (id 6). Vases are commonly gifted items and this one could
    potentially be filled with a plant or flower arrangement, making a
    lovely present.
}

User: Need something to use as a paperweight

LLM-Planner: {
inferred_query: Find object to use as a paperweight.,
relevant_objects: [0, 6, 7, 8, 15, 23, 29, 31],
query_achievable: true,
final_relevant_objects: [6],
explanation: The ceramic vase (id 6) could potentially be used as a
    paperweight due to its small size and assumed weight.
}
```

Listing 4: Sample queries and raw outputs from the LLM Planner

A3 Example Text Queries

Here we provide some of the text queries used in our experiments.

Replica Dataset Descriptive Queries: For each room, we randomly selected 20 ground truth annotations collected via Amazon Mechanical Turk (AMT). Here is a sample from `room0` and `office0`.

`office0` Descriptive Queries:

1. *This is a trash can against the wall next to a sofa.*
2. *A chaise lounge right next to a small table.*
3. *This is a television.*
4. *This is a dropped, tiled ceiling in what appears to be a classroom for children.*
5. *This is a plant and it is next to the screens.*
6. *This is the back of a chair in front of a screen.*
7. *A small table in front of a large gray sectional couch.*
8. *This is an armless chair and it's opposite a coffee table by the sofa.*
9. *This is a plug-in and it is on the floor.*
10. *These are table legs and they are underneath the table.*
11. *These are chairs and they are next to a table.*
12. *A diner style table in front of two chairs.*
13. *These are rocks and they are on the wall.*
14. *This is the right panel of a lighted display screen.*
15. *This is a planet and it is on the wall.*
16. *This is an electronic display screen showing a map, on the wall.*
17. *This is a couch and it is between a table and the wall.*
18. *This is a garbage can and it is in front of the wall.*
19. *This is a rug and it is on the floor.*
20. *This is a table that is above the floor.*

`room0` Descriptive Queries:

1. *This is a pillow and this is on top of a couch.*
2. *A pillow on top of a white couch.*
3. *This is a couch and it is under a window.*
4. *This is a stool and it is on top of a rug.*
5. *This is a side table under a lamp.*
6. *This is a ceiling light next to the window.*
7. *This is an end table and it is below a lamp.*
8. *These are books and they are on the table.*
9. *This is a couch and it is in front of the wall.*
10. *White horizontal blinds in a well lit room.*
11. *This is a striped throw pillow on the loveseat.*
12. *The pillow is on top of the chair.*
13. *This is a window and it is next to a door.*

14. *This is a hurricane candle and it is on top of a cabinet.*
15. *This is a vase and it is on top of the table.*
16. *This is a vent in the ceiling.*
17. *This is a fish and it is on top of a cabinet.*
18. *This is a window behind a chair.*
19. *This is a trash can against a wall.*
20. *Two cream colored cushioned chairs with blue pillows adjacent to each other.*

Replica Dataset Affordance Queries for Office Scenes:

1. *Something to watch the news on*
2. *Something to tell the time*
3. *Something comfortable to sit on*
4. *Something to dispose of wastepaper in*
5. *Something to add light into the room*

Replica Dataset Affordance Queries for Room Scenes:

1. *Somewhere to store decorative cups*
2. *Something to add light into the room*
3. *Somewhere to set food for dinner*
4. *Something I can open with my keys*
5. *Somewhere to sit upright for a work call*

Replica Dataset Negation Queries for Office Scenes:

1. *Something to sit on other than a chair*
2. *Something very heavy, unlike a clock*
3. *Something rigid, unlike a cushion*
4. *Something small, unlike a couch*
5. *Something light, unlike a table*

Replica Dataset Negation Queries for Room Scenes:

1. *Something small, unlike a cabinet*
2. *Something light, unlike a table*
3. *Something soft, unlike a table*
4. *Something not transparent, unlike a window*
5. *Something rigid, unlike a rug*

REAL Lab Scan Descriptive queries:

1. *A pair of red and white sneakers*
2. *A NASA t-shirt*
3. *A Rubik's cube*
4. *A basketball*
5. *A toy car*
6. *A backpack*

7. *An office chair*
8. *A pair of headphones*
9. *A yellow jacket*
10. *A laundry bag*

REAL Lab Affordance Queries:

1. *Something to use to disassemble or take apart a laptop*
2. *Something to use for cooling a CPU*
3. *Something to use for carrying books day to day*
4. *Something to use for temporarily securing a broken zipper*
5. *Something to use to help a student understand how a computer works*
6. *An object that is used in a sport involving rims and nets*
7. *Something to keep myself from getting distracted by loud noises*
8. *Something to help explain math proofs to a student*
9. *Something I can use to protect myself from the harsh winter in Canada*
10. *Something fun to pass the time with*

REAL Lab Negation Queries:

1. *A toy for someone who dislikes basketball*
2. *Shoes that you wouldn't wear to something formal*
3. *Something to protect me from the rain that's not an umbrella*
4. *Shoes that are not red and white*
5. *Something to make a cape with that's not green*
6. *Something to drink other than soda*
7. *Something to use for exercise other than weights*
8. *Something to wear unrelated to space or science*
9. *Something light to store belongings, not a backpack*
10. *Something to play with that's not a puzzle or colorful*

A4 Navigation Experiments

For our navigation experiments with the Jackal robot. Our robot is equipped with a VLP-16 lidar and a foward-facing Realsense D435i camera. We begin by building a pointcloud of a lab space using the onboard VLP-16 and Open3d SLAM [68]. The initial Jackal pointcloud does not include task-relevant objects and is downprojected to a 2D costmap for navigation using the base Jackal ROS stack.

We then stage two separate scenes with different objects: one for object search and another for traversability estimation. In both cases, we map the scene with an Azure Kinect Camera and rely on RTAB-Map [69] to obtain camera poses and the scene point cloud. We proceed to build a *ConceptGraphs* representation and register the scene point cloud with the initial Jackal map. For our navigation experiments, we only use the objects $\mathbf{O}_T$.

For object search queries, we use the LLM Planner described in Section A2 as part of a simple state machine. The robot first attempts to go look at the 3D coordinates of the most relevant object identified in $\mathbf{O}_T$ by the LLM Planner. We then pass the onboard camera image to LLaVA [55] and ask if the target object is in view. If not, we remove the target object from the scene graph and ask the LLM Planner to provide a new likely location for the object in the scene with the following GPT-4 system prompt:

```
The object described as 'description' is not in the scene.  Perhaps
someone has moved it, or put it away.  Let's try to find the object by
visiting the likely places, storage or containers that are appropriate
for the missing object (eg:  a a cabinet for a wineglass, or closet for a
broom).  The new query is:  find a likely container or storage space
where someone typically would have moved the object described as
'description'?
```

Listing 5: GPT system prompt for object localization.

For traversability estimation, we task GPT to classify a given object as traversable or non-traversable based on its description and possible tags. The system prompt is:

```
You are a wheeled robot that can push a maximum of 5 pounds or 2.27 kg.
Can you traverse through or push an object identified as 'description'
with possible tags 'possible_tags'?  Specifically, is it possible for you
to push the object out of its path without damaging yourself?
```

Listing 6: GPT-4 system prompt for traversability estimation.

We then take the pointclouds of each non-traversable objects and downproject them in the Jackal costmap before launching the navigation episode. The goal is provided in this case as a specific pose in the room.

For all experiments in this section, we run a local instance of LLaVA offboard on a desktop when needed and otherwise use the GPT-4 API for LLM queries.

A5 Limitations

As indicated in Sec. 3.8, there are a few failure modes of *ConceptGraphs* that remain to be addressed in subsequent work. In particular, the LLaVA-7B [55] model used for node captioning misclassifies a non-negligible number of small objects as *toothbrushes* or *pairs of scissors*. We believe that using more performant vision-language models, including instruction-finetuned variants of LLaVA [70] can alleviate this issue to a large extent. This will, in turn, improve the node and edge precisions of 3D scene graphs beyond what we report in Table 1.

In this work, we do not explicitly focus on improving LLM-based planning over 3D scene graphs. We refer the interested reader to concurrent work, SayPlan [52], for insights into how one might leverage the hierarchy inherent in 3D scene graphs, for efficient planning.