

# SlideSLAM: Sparse, Lightweight, Decentralized Metric-Semantic SLAM for Multi-Robot Navigation

Xu Liu<sup>\*†</sup>, Jiuzhou Lei<sup>\*</sup>, Ankit Prabhu<sup>\*</sup>, Yuezhan Tao, Igor Spasojevic, Pratik Chaudhari, Nikolay Atanasov, Vijay Kumar

**Abstract**—This paper develops a real-time decentralized metric-semantic SLAM algorithm that enables a heterogeneous robot team to collaboratively construct object-based metric-semantic maps. The proposed framework integrates a data-driven front-end for instance segmentation from either RGBD cameras or LiDARs and a custom back-end for optimizing robot trajectories and object landmarks in the map. To allow multiple robots to merge their information, we design semantics-driven place recognition algorithms that leverage the informativeness and viewpoint invariance of the object-level metric-semantic map for inter-robot loop closure detection. A communication module is designed to track each robot’s observations and those of other robots whenever communication links are available. The framework supports real-time, decentralized operation onboard the robots and has been integrated with three types of aerial and ground platforms. We validate its effectiveness through experiments in both indoor and outdoor environments, as well as benchmarks on public datasets and comparisons with existing methods. The framework is open-sourced and suitable for both single-agent and multi-robot real-time metric-semantic SLAM applications.

**Index Terms**—Metric-Semantic SLAM; Multi-Robot Systems; Aerial Systems: Perception and Autonomy; SLAM

## I. INTRODUCTION

Robotic systems are expected to make an impact in demanding applications, such as forest inventory management, orchard yield estimation, infrastructure inspection and household assistance. This demands interpreting human instructions given in semantically meaningful terms relating to objects and properties in the robot’s environment. To execute such missions autonomously, robots must understand their environment beyond its geometric structure and perceive it at a semantic



Figure 1: *Robot platforms used in our experiments.* We utilize three types of robots for our experiments: two aerial platforms, the Falcon 250 UAV (left) and the Falcon 4 UAV (middle), and one ground platform, the Scarab UGV (right). The Light Detection and Ranging (LiDAR)-equipped robot (Falcon 4) is primarily used for outdoor operations due to its size and superior sensing capabilities. The RGB and Depth (RGBD) camera-based robots (Falcon 250 and Scarab) are more suitable for cluttered indoor environments due to their smaller footprints. All three platforms have GPS-denied autonomous navigation capabilities, enabling them to safely explore cluttered environments using only onboard computation and sensing.

level. This requires building and maintaining a semantically meaningful representation of the environment that encodes actionable information (e.g., timber volume and health in forests, corrosion in infrastructure, survivors’ locations in natural disasters). Such a representation has to be storage efficient for the robots to maintain over large-scale missions, and has to allow efficient optimization for Simultaneous Localization and Mapping (SLAM).

While traditional SLAM approaches [1], [2] offer excellent accuracy in geometric perception, including estimating robot poses and reconstructing 3D geometric structures (points, surfaces, voxels), they are often insufficient to support large-scale missions, particularly when used with multiple heterogeneous robots for real-time autonomy. The challenges include managing the excessive computational load, meeting the high storage demands of large-scale maps, and handling loop closure and map merging operations. Moreover, they offer limited generalizability across different robot platforms or sensing modalities and lack semantic map representations required for executing semantically meaningful tasks.

Motivated by this, recent advances in SLAM have pushed toward constructing metric-semantic maps [13], [11], [10], [17], [18]. However, they are typically limited to single-robot setups. For scalability and efficiency, an ideal system should support deployment across heterogeneous robot teams, enabling effective collaboration without relying on centralized communication or global localization infrastructure. Extending single-robot metric-semantic SLAM to decentralized operation across multiple heterogeneous robots introduces several unique

<sup>\*</sup> Equal contribution. <sup>†</sup> Corresponding author.

Manuscript received January 9, 2025; revised May 1, 2025; accepted September 2, 2025. This article was recommended for publication by Editor Javier Civera upon evaluation of the reviewers’ comments. This work was supported by funding from the IoT4Ag Engineering Research Center funded by NSF (NSF EEC-1941529), National Robotics Initiative 3.0: Innovations in Integration of Robotics (NSF 21-559), NIFA grant 2022-67021-36856, NSF grant CCR-2112665, and the ARL DCIST CRA W911NF-17-2-0181.

X. Liu is with Microsoft, Redmond, WA 98052, USA (e-mail: xuliu5@microsoft.com); J. Lei is with Texas A&M University, College Station, TX 77840, USA (e-mail: jiuzl@tamu.edu); and I. Spasojevic is with the University of California, Riverside, CA 92521, USA (e-mail: igors@ucr.edu). This work was done while they were with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104, USA.

A. Prabhu, Y. Tao, P. Chaudhari, and V. Kumar are with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104, USA (e-mail: {praankit, yztao, pratikac, kumar}@upenn.edu).

N. Atanasov is with the Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093, USA (e-mail: natanasov@ucsd.edu).

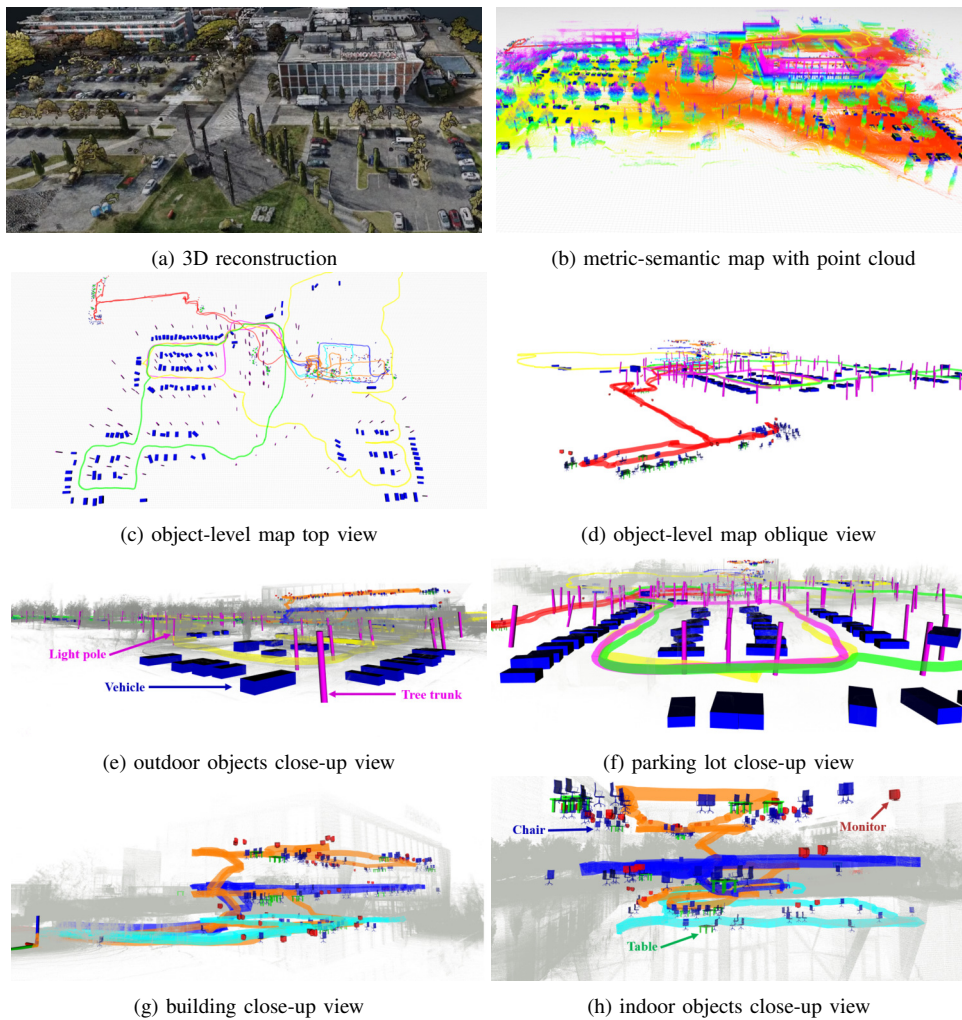


Figure 2: *Metric-semantic SLAM results from seven data sequences collected by heterogeneous robots.* Trajectories in different colors correspond to different data sequences. Fig. 2a shows a 3D reconstruction of the Pennovation campus at the University of Pennsylvania. Outdoor objects, such as vehicles, tree trunks, and light poles are mapped as shown in Fig. 2e. Indoor objects, such as chairs, tables, and monitors are mapped as shown in Fig. 2h. Fig. 2b shows the same metric-semantic map overlaid on top of an accumulated point cloud constructed by our Falcon 4 UAV. Fig. 2c shows an orthophoto depicting the merged metric-semantic map of three parking lots and two buildings constructed by seven robots. Fig. 2g and Fig. 2h show a zoomed-in view of one of the lab buildings

challenges. These include performing efficient place recognition and map merging among robots, maintaining consistent and probabilistically sound sensor measurement fusion under intermittent communication, and enabling collaboration among heterogeneous platforms.

To address these challenges, we develop a multi-robot decentralized metric-semantic SLAM system that is open-sourced<sup>1</sup> and suitable for real-time autonomous navigation and exploration onboard resource-constrained single-robot as well as heterogeneous multi-robot platforms. We enable this by mapping the environment using a sparse map representation that explicitly models objects using simple shapes and can be used for task planning and long-horizon SLAM. The proposed sparse object-level map representation offers numerous advantages. First, it avoids the high runtime and memory demands of using raw point clouds or images, which is particularly im-

portant for large-scale missions with Size, Weight, and Power (SWaP) constrained robot teams. Second, the lightweight nature of our representation ensures that robot-to-robot information sharing is feasible, even with limited communication bandwidth. This, coupled with our semantics-driven place recognition and loop closure algorithm, enables efficient detection of inter-robot loop closures. Third, it provides a direct and intuitive representation for robots to accomplish high-level semantically meaningful tasks, such as actively exploring to search and reduce uncertainties in objects of interest. Fourth, unlike traditional approaches based on dense geometric features that usually have to marginalize variables frequently, which unavoidably leads to loss of information, our representation enables us to keep track of actionable information over a much larger scale. Finally, it provides a unified representation across different sensing modalities.

We summarize our **contributions** as follows.

*Algorithm:* We develop a real-time decentralized metric-

<sup>1</sup>Code: [https://github.com/KumarRobotics/SLIDE\\_SLAM](https://github.com/KumarRobotics/SLIDE_SLAM). Project website: <https://xurobotics.github.io/slideslam/>.

Table I: *Related work*. This table compares SlideSLAM against state-of-the-art methods, focusing on key attributes for enabling semantics-in-the-loop autonomy with robot teams. In this table, decentralized methods support operation with asynchronous observation updates from other robots. Semantic localization means that objects are used for localization between consecutive key poses. Semantic loop closure indicates that objects are explicitly used for place recognition and loop closures. Optimization over object models involves explicitly modeling objects as geometric shapes and optimizing them jointly with robot poses over time. Real-time with autonomy means that the metric-semantic SLAM system operates in real-time on-board the robots and is fully integrated into an autonomous exploration or navigation system.

	Year	Multi-Robot	Decentralized, Asynchronous	Semantic Localization	Semantic Loop Closure	Optimization over Object Models	Real-time w/ Autonomy
<b>SlideSLAM</b>	2024	✓	✓	✓	✓	✓	✓
Tao et al. 3D Active [3]	2024			✓	✓		✓
Chang et al. Hydra-Multi [4]	2023	✓		✓	✓	✓	
Liu et al. Active Metric-Semantic [5]	2023	✓		✓		✓	✓
Wu et al. An Object SLAM [6]	2023			✓	✓	✓	
Tian et al. Kimera-Multi [7]	2022	✓	✓				
Liu et al. Large-Scale [8]	2021			✓		✓	✓
Shan et al. Orc-VIO [9]	2020			✓		✓	
Yang et al. Cube SLAM [10]	2019			✓		✓	
Nicholson et al. Quadric SLAM [11]	2019			✓		✓	
Choudhary et al. Distributed Mapping [12]	2017	✓	✓	✓			
Bowman et al. Probabilistic [13]	2017			✓			
Salas-Moreno et al. SLAM++ [14]	2013			✓		✓	
Cunningham et al. DDF-SAM [15], [16]	2013	✓	✓				

semantic SLAM framework that supports heterogeneous aerial and ground robots, which includes

- a computationally efficient back-end that uses our object-level metric-semantic map representation, and a flexible front-end that supports both LiDAR and RGBD sensors,
- semantics-driven place recognition algorithms that use sparse object-level maps for map merging,
- a decentralized multi-robot collaboration module that facilitates information sharing, even under intermittent communication conditions.

*System integration:* We integrate and deploy the proposed framework on a heterogeneous team of robots, demonstrating its capacity to enable semantics-in-the-loop autonomous navigation and exploration in various indoor and outdoor environments. The system operates in real time onboard SWaP-constrained robots, while maintaining moderate computation and memory demands.

*Experiments:* We conduct extensive real-world experiments and provide thorough empirical results and analysis that highlight the efficiency, accuracy, and robustness of our system. We have also made our framework available to the public.

## II. RELATED WORK

In this section, we categorize the related works into four key areas: metric-semantic SLAM, place recognition, multi-robot SLAM, and semantics-in-the-loop navigation. Table I provides a quick summary of several key related works. The rest of this section provides a more comprehensive overview of the related works in each category.

### A. Metric-semantic SLAM

Unlike traditional SLAM, metric-semantic SLAM constructs a map that encodes geometric features and semantic information on objects of interest. Metric-semantic SLAM has gained significant success and popularity in the past decade [21], driven by the rapid development of deep learning

techniques that extract semantic information from sensor data. A variety of map representations are used in existing metric-semantic SLAM literature. Some use dense semantic maps, such as meshes [17], volumetric maps [22], [23], surfels [24], and 2.5D grid maps [25], [26], [27]. Others use sparse object-level maps with prior information of object shapes, such as centroids [13], cubes [10], ellipsoids [11], cylinders [28], structured object models with prior shape constraints [29], [9], and mesh-based object models [30], [14]. Following the recent trends in neural implicit representation, some prior works [31], [32], [33] have also incorporated it into a map representation capable of metric-semantic SLAM, where objects are modeled implicitly using latent features from neural networks and used as localization constraints and for object association. While dense metric-semantic maps are suitable for obstacle avoidance and facilitate more fine-grained modeling of objects, their computational and memory demands, especially in SWaP-constrained platforms, can be significant. Thus, sparse object-level maps are desirable for real-time downstream tasks such as active information gathering, object manipulation and multi-robot collaboration. This is because semantic information is necessary for robots to perceive their environment and execute tasks in a semantically meaningful manner. Sparsity helps robots reduce resource demands on computation, storage, and communication. A recent work in this space is ConceptGraphs [18], which builds open-vocabulary, semantically rich 3D object maps. However, ConceptGraphs is not designed for real-time, onboard operations on resource-constrained robots. Its detection and 3D object mapping pipelines require extensive computational resources. By contrast, our work seeks to develop a metric-semantic SLAM framework that utilizes sparse explicitly modeled objects and supports real-time decentralized operations within a heterogeneous robot team.

### B. Place recognition

Place recognition and loop closure address the challenge of identifying whether a robot revisits a previously mapped

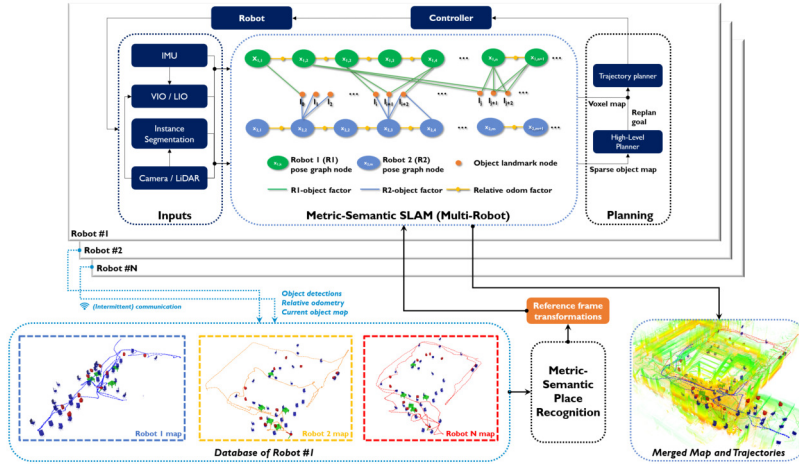


Figure 3: *System Diagram*. Our system takes in data streams from each robot’s onboard sensors, which can be either an RGBD camera or a LiDAR, and performs instance segmentation to extract semantic object features. Meanwhile, low-level odometry, either Visual-Inertial Odometry (VIO) [19] or LiDAR-Inertial Odometry (LIO) [20], provides relative-motion estimates between consecutive key poses. Next, the metric-semantic SLAM algorithm takes in such semantic observations and relative motion estimates, and constructs a factor graph consisting of both robot pose nodes and object landmark nodes. Meanwhile, our multi-robot communication module (see Fig. 4) opportunistically leverages connectivity to share lightweight semantic observations among robots in a decentralized way. Based on this shared information, our metric-semantic place recognition algorithm constantly checks for possible inter-robot loop closures at a fixed rate. Once a loop closure is detected, the resulting transformation between each pair of robots is used to transform all observations into each robot’s reference frame. These observations are then added to their own factor graphs, forming a merged metric-semantic map. Note that the entire perception-action loop runs in a decentralized manner onboard each robot. Besides the obvious differences in control algorithms, the planning modules and the front-end processing algorithms are also different across each robot platform. This is due to the need to accommodate the differences in sensing modalities (RGBD and LiDAR), operating environments (indoor, urban, and forest), and traversal modes (ground and aerial). However, the core metric-semantic SLAM framework remains the same.

location. This enables odometry drift correction in single-robot systems (intra-robot loop closure) and map merging in multi-robot systems (inter-robot loop closure). Prevalent approaches typically use consistency graphs. These methods operate on the assumption that pairwise distances between points or landmarks remain unchanged across two candidate maps [34], [35], [36], [37]. The largest set of consistent associations is then identified and used to estimate the transformation. However, this assumption is invalid with the object-level map representations, where object detection and localization errors introduce additional inconsistency. Some geometric-based place recognition methods, such as [38], design hand-crafted features for LiDAR point clouds that are translation and rotation invariant to match them accurately and establish robust place recognition. Other methods like [39] generate similar global matching features and perform relative pose refinement using deep neural networks. While these methods demonstrate impressive place recognition performance, they may have high memory usage when storing a large number of geometric features or require large amounts of data to fine-tune the neural network for place recognition in novel environments. Conversely, semantic maps are much sparser, provide richer information and can be abstracted from any sensing modality, LiDAR or camera. Some approaches leverage this fact and propose descriptor-based methods based on the spatial relationship of semantic objects. In [40], a random walk descriptor encoding semantic labels of neighboring nodes is designed for each semantic node. A semantic histogram-descriptor-based method is proposed in [41], which improves the efficiency of

graph matching compared to the random-walk-based method. [6] also leverages random walk descriptors with semantic information but further adds information on object parameters, angles, and distance between an object node and its neighboring nodes to filter out false candidates. In [42], Urquhart tessellations are derived based on the positions of semantic landmarks (tree trunks in forests). Polygon-based descriptors are then computed to describe the local neighborhood of the robot, which are then used for place recognition. Hierarchical descriptors consisting of appearance-based descriptors like Distributed Bag of Words (DBoW2) [43] and higher level descriptors encoding information about a node’s neighboring objects or places are designed in [44], [4]. Drawing inspiration from some of these prior works, we design two complementary place recognition algorithms that leverage our metric-semantic map representation from diverse sensing modalities (LiDAR and RGBD Camera) and adopt exhaustive search-based and descriptor-based strategies, respectively.

### C. Multi-robot SLAM

Multi-robot SLAM systems expand upon their single-robot counterparts by incorporating two crucial modules: inter-robot loop closure and multi-agent graph optimization. Numerous studies address the challenge of place recognition, as detailed in Section II-B. Representative works in multi-robot SLAM include [45], [46], which rely on a central base station or agent to merge measurements from multiple robots, and [47], [48], [49], [50], which operate in a distributed or decentralized manner. Distributed multi-robot systems involve robots that

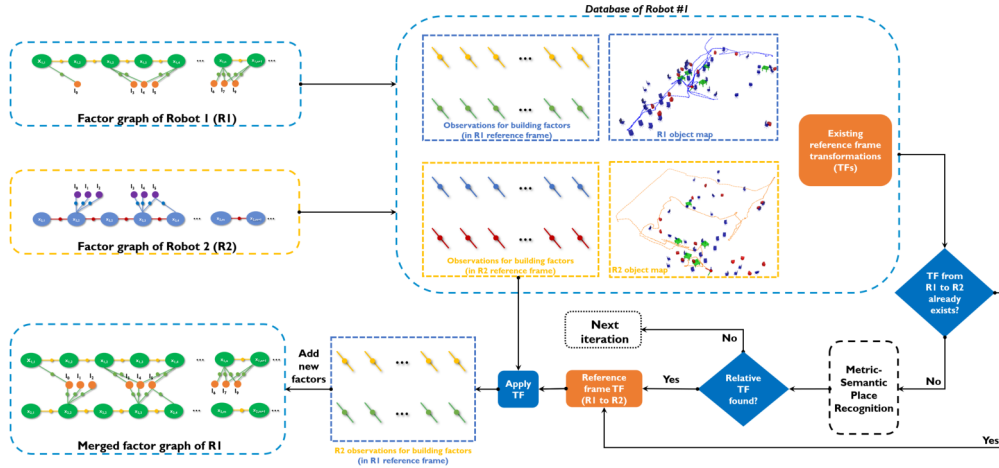


Figure 4: *Multi-robot collaboration module for decentralized metric-semantic SLAM.* The robots share lightweight metric-semantic observations necessary for constructing the factors between object landmarks and robot poses in the factor graph, which include the detected objects and the odometry relative motion estimate (w.r.t. the previous pose) associated with each key pose in the factor graph. Once the metric-semantic place recognition module successfully finds a loop closure with another robot, the shared observations from that robot will be transformed into the current robot’s reference frame and added to the factor graph of the current robot.

coordinate through message passing, possibly with central nodes, while decentralized systems operate without a central decision point, enhancing robustness by allowing each robot to make independent decisions. Decentralized multi-robot SLAM is a promising framework in multi-robot research. It enables robots to operate independently and collaborate opportunistically when communication links are established, allowing them to accomplish complex tasks such as large-scale collaborative exploration without relying on infrastructure.

However, the prior works mentioned above only contain geometric information during mapping. Recent efforts extend the field to metric-semantic mapping. [7] presents a system producing a dense 3D semantic mesh, incorporating distributed loop closure detection and distributed robust pose-graph optimization based on Riemannian block coordinate descent [51]. However, semantic information is used primarily in the map representation rather than for localization and loop closure. Another related work, [52], proposes a distributed Pose Graph Optimization (PGO) framework based on a distributed Gaussian-Seidel approach [53] by considering only overlapping constraints among robots in the optimization and extracting objects as landmarks. Object-level semantic landmarks, because they are viewpoint invariant, can facilitate collaboration among heterogeneous robots. Few works exist to solve the problem of collaborative SLAM using heterogeneous sensors. Among them, [54] fuses local pose graphs from different SLAM algorithms into a global pose graph. However, it is limited to vision-based sensors, which require sharing images associated with each pose for map merging. Our work proposes a more generic and efficient system towards decentralized metric-semantic SLAM that supports real-time operation on SWaP-constrained heterogeneous robot platforms with either RGBD or LiDAR sensors.

#### D. Semantics-in-the-loop navigation and exploration

Most autonomous navigation systems rely on geometric maps. Although such maps are reliable for obstacle avoidance,

they lack semantic information that robots can leverage to improve their state estimation and planning while navigating the environment. Object-based landmarks help minimize odometry drift, thus benefiting autonomous navigation if executed onboard in real-time [8]. Recent work goes beyond improving odometry but uses semantic information for active uncertainty reduction planning [55], [56], [5], and multi-robot collaborative planning [27]. The use of semantic information in active SLAM is relatively uncommon [57]. Some representative work in this field includes [58], [59]. However, there has not yet been a real-time decentralized metric-semantic SLAM framework that enables a team of heterogeneous robots to explore GPS-denied environments while reasoning about the semantic information and leveraging it to facilitate multi-robot collaboration. Our work seeks to develop such a framework.

### III. PROBLEM FORMULATION

Given an unknown environment and a team of heterogeneous robots, our objective is to construct a hierarchical metric-semantic map of the environment without relying on any infrastructure. The robots can communicate with each other opportunistically. Each robot must estimate its pose and the state of the environment and use this information to determine actions for navigation and exploration in real time.

#### A. Preliminaries

Consider  $K$  robots, indexed from 1 through  $K$ . The map of the  $k$ -th robot  $\mathcal{M}^{(k)}$  consists of a set of object landmarks  $\{\ell_1^k, \ell_2^k, \dots, \ell_{n_k}^k\}$  belonging to a set of pre-defined classes (i.e., object categories). An object is described by its class and a state vector, including its position, orientation, and shape information. For example, a cuboid has three shape parameters describing its width, length, and height. A cylinder has one parameter describing its radius (we do not estimate its height as it is unobservable for the majority of measurements due to limited field of view). Ellipsoid models are simplified to have

two shape parameters (radius and height) and known identity orientation. Objects of each class are modeled using one of the aforementioned three shapes as specified by the user. For each key pose of robot  $k$ , denoted by  $\mathbf{x}_t^k$ , the associated measurements include object detections from this pose and the relative motion from the previous key pose. The details of the shape models, robot measurements, and the factor graph used for optimization are presented in Section IV.

### B. Dec-Metric-Semantic SLAM for multi-robot exploration

In this section, we state the general active decentralized metric-semantic SLAM problem (Dec-Metric-Semantic SLAM). Let  $\mathcal{I}_t^{(k)}$  be all past observations that robot  $k$  has accrued by time  $t$ , from its own measurements, as well as from messages it has received from other robots. At time  $t$ , every robot  $k$  executes an exploration or navigation policy:

$$\mu_t^{(k)} : \mathcal{I}_t^{(k)} \rightarrow \mathcal{A}, \quad (1)$$

where  $\mathcal{A}$  is an action set of robot  $k$  that can include, for example, waypoints to follow or low-level control inputs. We formulate the exploration problem as:

$$\max_{\substack{(\mu_t^{(k)})_{1 \leq t \leq T, \\ 1 \leq k \leq K}} \mathbb{E} \left[ \sum_{k=1}^K \sum_{t=1}^T \mathcal{U}_t(\mathbf{x}_t^{(k)}, \mathcal{M}; \mathcal{I}_t^{(k)}) \right], \quad (2)$$

where  $\mathcal{M}$  represents the (static) map of the environment, and  $\mathbf{x}_t^{(k)}$  the state of robot  $k$  at time  $t$ . The function  $\mathcal{U}_t$  is a (potentially) time-varying utility function that quantifies the accuracy with which a robot can estimate specified quantities of interest given all the information it has amassed thus far.

In our metric-semantic SLAM formulation, the quantities of interest include the robot poses  $\mathbf{x}_t^{(k)}$  and the map of the environment  $\mathcal{M}$ . At each time  $t$ , robot  $k$  receives odometry measurements

$$\mathbf{h}_t^{(k)} = \mathbf{x}_t^{(k)} \ominus \mathbf{x}_{t-1}^{(k)} + \mathbf{w}_t^{(k)}, \quad (3)$$

where  $\ominus$  computes the relative pose between two key poses, and map measurements

$$\mathbf{m}_t^{(k)} = h(\mathbf{x}_t^{(k)}, \mathcal{M}) + \mathbf{v}_t^{(k)}. \quad (4)$$

We assume  $(\mathbf{w}_t^{(k)})_{t \leq T, k \leq K}$  and  $(\mathbf{v}_t^{(k)})_{t \leq T, k \leq K}$  are independent random variables with zero mean and covariance  $\Lambda_k$  and  $\Gamma_k$ , respectively. Currently, both variables  $\mathbf{w}_t^{(k)}$  and  $\mathbf{v}_t^{(k)}$  are set to constant values since key poses  $\mathbf{x}_t^{(k)}$  are sampled at a fixed robot travel distance interval. However, in our implementation, we provide an interface to allow scaling  $\mathbf{w}_t^{(k)}$  according to travel distance and modify  $\mathbf{v}_t^{(k)}$  accordingly to account for sensor and object detection noise.

We let  $C_t \in \{0, 1\}^{K \times K}$  be the time-varying symmetric adjacency matrix of the undirected graph modeling the robot's communication. At every time  $t$ , any distinct pair of robots  $i$  and  $j$  communicate if and only if  $(C_t)_{i,j} = (C_t)_{j,i} = 1$ . We denote the message that robot  $i$  sends to robot  $j$  at time  $t$  via  $r_{(i \rightarrow j)}(t)$ . We denote the neighbors of any robot  $k$  in the graph at time  $t$  via

$$\mathcal{N}_t^{(k)} = \{j \in [K] \setminus \{k\} : (C_t)_{j,k} = 1\}, \quad (5)$$

where  $[K] := \{1, \dots, K\}$ . For any distinct pair of robots  $(i, j)$ , we denote their last meeting time up to time  $t$  as

$$L_t^{(i,j)} = \max\{s \leq t : (C_s)_{i,j} = 1\}, \quad (6)$$

noting the symmetry in the superscript, i.e.,  $L_t^{(i,j)} = L_t^{(j,i)}$ . Then, the information state of robot  $k$  may be defined recursively as

$$\mathcal{I}_t^{(k)} = \bigcup_{j \in [K] \setminus \{k\}} \mathcal{I}_{L_t^{(k,j)}} \cup \bigcup_{s \in [t]} \left( \{\mathbf{h}_s^{(k)}\} \cup \{\mathbf{m}_s^{(k)}\} \right), \quad (7)$$

with initial condition  $\mathcal{I}_0^{(k)} = \emptyset$  for all  $k \in [K]$ .

We develop an approach that uses a set of policies  $\mu$  (e.g., active SLAM, exploration, etc.) that take as input compressed representations of  $\mathcal{M}$  using a custom data structure, i.e., a set of object landmarks with corresponding poses, shapes, and class labels, as well as a custom rule for updating the data structure both using new measurements from the robot and information from its communication with other robots. The exploration or navigation policy of robot  $k$  may then be defined via

$$\mu_t^{(k)}(\mathcal{I}_t^{(k)}) := \mu_t^{(k)}(\Pi(\mathcal{I}_t^{(k)})), \quad (8)$$

where the information projection operator  $\Pi$  is given by

$$\Pi(\mathcal{I}_t^{(k)}) = (\hat{\mathbf{x}}_{1:t}^{(k)}, \hat{\mathcal{M}}_t^{(k)}). \quad (9)$$

$\Pi$  extracts an estimate of the previous trajectory of robot  $k$ , as well as its estimate of the map via

$$\hat{\mathbf{x}}_{1:t}^{(k)}, \hat{\mathcal{M}}_t^{(k)} = \operatorname{argmax}_{\mathbf{x}_{1:t}^{(k)}, \mathcal{M}} P(\mathbf{x}_{1:t}^{(k)}, \mathcal{M} \mid \mathcal{I}_t^{(k)}), \quad (10)$$

where  $P(\mathbf{x}_{1:t}^{(k)}, \mathcal{M} \mid \mathcal{I}_t^{(k)})$  represents the joint probability distribution of the robot  $k$ 's trajectory from the start up to time  $t$ , and the map  $\mathcal{M}$ , which includes object landmarks, conditioned on the information state  $\mathcal{I}_t^{(k)}$  of the robot  $k$ . Finally, we use the following form of messages

$$r_{(i \rightarrow j)}(t) = (\mathcal{I}_t^{(i)}, \hat{\mathcal{M}}_t^{(i)}). \quad (11)$$

In principle, robot  $j$  can construct  $\hat{\mathcal{M}}_t^{(i)}$  from  $\mathcal{I}_t^{(i)}$ . Given the fact that the compressed representation  $\hat{\mathcal{M}}_t^{(i)}$  we use is memory efficient, we directly share this information to avoid the extra computation brought about by the reconstruction step. The map  $\hat{\mathcal{M}}_t^{(i)}$  is shared only for checking inter-robot loop closures and estimating the transformations. Once a valid inter-robot loop closure is established,  $\mathcal{I}_t^{(i)}$  is used to form the updated  $\mathcal{I}_t^{(j)}$  as mentioned in Eq. (7).  $\mathcal{I}_t^{(j)}$  will then be used to update  $\hat{\mathcal{M}}_t^{(j)}$ .

## IV. METRIC-SEMANTIC SLAM

### A. Approach overview

We next turn to an overview of the approach for computing the information projection operator  $\Pi$ , which is one of the core contributions of this paper. Unlike traditional geometric-only methods such as point cloud registration, our method leverages both *geometric* and *semantic* information to find common components of maps estimated by different robots, and then refines such estimates using both its own measurements and

measurements made by other robots. We illustrate this idea with a pair of robots  $k$  and  $j$  at time  $t$ , though the argument can be readily generalized to simultaneous interactions of a larger number of robots. Robot  $k$  solves the maximum likelihood state estimation problem at time  $t$  as follows. For every robot  $j \in \mathcal{N}_t^{(k)}$ , we consider the message  $r_{j \rightarrow k}(t) = (\mathcal{I}_t^{(j)}, \hat{\mathcal{M}}_t^{(j)})$ . First we determine common map observations made by robots  $k$  and  $j$  up to time  $t$  by performing inter-robot loop-closure detection on  $\hat{\mathcal{M}}_t^{(j)}$  and  $\hat{\mathcal{M}}_t^{(k)}$ . The output of the loop-closure module yields an estimate of the relative transformation  ${}^{(k)}T_{(j)} = ({}^{(k)}R_{(j)}, {}^{(k)}\mathbf{t}_{(j)}) \in \mathbb{SE}(3)$  between reference frames of robot  $j$  and robot  $k$ . In particular, a point with coordinates  ${}^{(j)}\mathbf{p}$  in the reference frame of robot  $j$  has coordinates  ${}^{(k)}\mathbf{p} = ({}^{(k)}R_{(j)} {}^{(j)}\mathbf{p} + {}^{(k)}\mathbf{t}_{(j)})$  in the reference frame of robot  $k$ . This transformation is then used to determine which landmarks in the map of robot  $j$  correspond to landmarks that also exist in the map of robot  $k$ . Thereafter, the trajectories of both robot  $j$  and  $k$  are optimized together with the union of object models in the map, taking care to associate measurements of the same landmark taken by two robots with the same variable in the factor graph:

$$\hat{\mathbf{x}}_{1:t}^{(j)}, \hat{\mathbf{x}}_{1:t}^{(k)}, \hat{\mathcal{M}}_t^{(k)} = \underset{\mathbf{x}_{1:t}^{(j)}, \mathbf{x}_{1:t}^{(k)}, \mathcal{M}}{\operatorname{argmax}} P(\mathbf{x}_{1:t}^{(j)}, \mathbf{x}_{1:t}^{(k)}, \mathcal{M} \mid \mathcal{I}_t^{(k)}). \quad (12)$$

The above equation provides details on the information projection operator  $\Pi$ .

Our system works on robots with different types of sensors, including RGBD cameras and LIDARs, as illustrated in Fig. 3 and detailed in Section V-A. Our metric-semantic SLAM framework handles such heterogeneity by using different front-end pipelines. We integrate the proposed metric-semantic SLAM framework with our autonomous exploration and navigation stack as detailed in Section V-B. In the rest of this section, we will provide details on individual modules of the proposed framework.

## B. Map representation

Table II: *Requirements for map representation.* Generic implies that the map must be adaptable to various sensing modalities and environments. Informative means that the map should contain both metric and semantic information. Sparse indicates that the map needs to be memory efficient for storage and sharing.

	Heterogeneous robots and environments	Place recognition and loop closure	Real-time exploration and navigation
Generic	✓	✓	
Informative		✓	✓
Sparse	✓	✓	✓

The metric-semantic SLAM framework needs to satisfy several design attributes. These include (1) support for heterogeneous robot teams, where the robots may carry different sensors, operate in different environments, and share information with each other via intermittent communication, (2) support for inter-robot place recognition for map merging, (3) capability of real-time operation under the constraints of onboard computation and memory resources, so that the robot can use the estimated pose and the metric-semantic map to guide its navigation.

As summarized in Table II, to support different kinds of sensors (RGBD cameras and LiDARs) and objects in different environments, we require the map representation to be generic. To work with multiple robots and allow the sharing of information with limited communication bandwidth, we require the map representation to be sparse. To enable efficient and accurate place recognition and loop closure, we additionally require the map representation to be informative so that the robots can distinguish different semantic objects even under perceptually aliased conditions. Finally, to enable efficient, large-scale, and real-time autonomous exploration, we also require the map to be sparse and informative. The informativeness allows the exploration planner to better understand the environment, such as the uncertainties in the semantic objects, and generate informative paths.

Therefore, we design a generic, sparse, and informative metric-semantic map representation. It contains a set of objects, each represented by a semantic class and a state vector that describes its model parameters, including position, orientation, and shape information. Such a map representation, as illustrated in Fig. 2, can be maintained throughout the entire mission of the robot while enabling tasks such as localization, mapping, map sharing, place recognition, map merging, and active exploration over a large scale and in real time.

The metric-semantic SLAM problem can be broken down into two subproblems: (a) determining the discrete semantic labels and data association of detected objects and (b) optimizing over the continuous variables of robot poses and object model parameters (pose and shape). In this work, we approach problem (a) using deep neural networks for detection and assignment algorithms for data association, and problem (b) by first converting object observations into factors in a factor graph and then using an incremental smoothing and mapping algorithm (iSAM2) [60], [61] to optimize it. In the rest of this section, we provide details on these two steps.

## C. Object detection and modeling

The front-end of our SLAM framework is responsible for processing raw data from different sensors and converting it into object-level observations. The process is broken down into three components: 1) object detection or instance segmentation; 2) object instance tracking to accumulate observations from different views; 3) shape model fitting to the instances based on the class labels.

For semantic segmentation on point cloud from LiDAR, we trained RangeNet++ [62] with a small modified backbone. This model performs segmentation on the range image, which is a spherical projection of a point cloud. This operation drastically decreases the inference time compared to performing inference on a 3D data structure such as a point cloud and makes real-time inference onboard SWaP-constrained platforms possible.

For the RGBD sensor, we use YOLOv8 [63] to perform instance segmentation on RGB images and backproject the pixels in each object instance’s segmented mask into point clouds using depth information. We apply depth-based thresholding within each object instance’s mask, keeping only the

points within a certain range of depth percentile, to avoid including noisy pixels.

In addition to supporting the aforementioned closed-set object detection methods that only enable the robot to perceive a subset of the semantic information in the world characterized by pre-identified labels, we provide support to incorporate open-vocabulary object detectors into our system. Specifically, we utilize the YOLO-World model [64] as our open-vocabulary object detector due to its real-time onboard performance. The YOLO-World model processes pairs of images and a list of text prompts, where we specify a set of object categories of interest. The model then outputs object detections corresponding to each query in the text prompts. We apply depth thresholding to these detections, similar to the approach used for YOLOv8.

Once the point cloud per detection is extracted, these detected objects are tracked over time using the Hungarian assignment algorithm [65]. Tracking instances across time and from different viewpoints ensures robust geometry of objects and minimizes false positives by rejecting those tracks that only appeared once or very few times, which is necessary for metric-semantic mapping. Finally, the algorithm also helps with robustness against moving objects by tracking and accumulating their point clouds. For example, an accumulated instance of a moving object appears elongated when compared to its true size, and such objects can then be filtered out by simply applying a threshold on their dimensions. This preliminary tracking step is used to more robustly obtain object-level observations. In the factor graph optimization back end, there is a separate object tracking step that associates these object observations with existing object landmarks to construct factors in the factor graph.

Once a robust object instance is obtained, depending on the type of object, a shape model is fitted to the instance. Objects such as vehicles use a cuboid model, tree trunks and light poles use a cylinder model, and irregularly shaped objects that cannot be properly categorized into either a cuboid or cylinder use an ellipsoid model. Having determined an appropriate model for the object instance, these models are converted into factors to be added to the factor graph for the SLAM back-end optimization.

We highlight that we do not rely on a specific object detector. Our implementation has a modular front-end design, enabling the seamless integration of alternative object detectors in the future.

#### D. Factor graph optimization with object models

After acquiring the object detections associated with each pose of the robot, we now formulate the back-end optimization problem for the metric-semantic SLAM. When a new object detection arrives, we first check if we can associate it with existing object landmarks in the map. Upon the first observation of the object, we will initialize an object landmark. After that, once we associate new object models with existing object landmarks, we use the detection-landmark matches to form factors between the current pose and the matched landmarks in the factor graph. A valid object detection-landmark association

needs to meet two criteria: (1) they have the same semantic label, and (2) their distance and model difference are within a certain threshold.

A critical step to form such factors is to define the constraining relationship (i.e., graph edges) between the object and robot pose graph vertices. This means we need to define the measurement models that convert observations of relative odometry and object models into factors. We denote the matrix form of the robot pose  $\mathbf{x}_t$  as  $\mathbf{H}_s^w$ , where  $\mathbf{R}_s^w$  is the rotational component and  $\mathbf{t}_s^w$  is the translational component. In the rest of this section, we will describe this process in detail.

1) *Cuboid factors*: First, we define the state vector of the cuboid object model as:  $\ell^g = [\mathbf{r}; \mathbf{t}; \mathbf{d}]$ , where  $\mathbf{r} = [r_x, r_y, r_z]^\top$  is the rotation vector,  $\mathbf{t} = [t_x, t_y, t_z]^\top$  is the translation vector, and  $\mathbf{d} = [d_x, d_y, d_z]^\top$  is the size.

We first recover the cuboid's  $\mathbb{SE}(3)$  pose based on its state vector  $\ell_i^g = [\mathbf{r}; \mathbf{t}; \mathbf{d}]$ , and transform it from the reference frame to the body frame using  $\mathbf{H}_{\text{cub}}^s = \mathbf{H}_w^s \mathbf{H}_{\text{cub}}^w$ . The error function for cuboidal objects is as follows:

$$\mathbf{e}_{\text{cub}} = \begin{bmatrix} \mathbf{log}((\mathbf{H}_{\text{cub}}^s(\mathbf{z}))^{-1}(\mathbf{H}_w^s \mathbf{H}_{\text{cub}}^w))^\vee \\ \mathbf{d} - \mathbf{d}(\mathbf{z}) \end{bmatrix}, \quad (13)$$

where  $\vee$  is vee operator that maps the  $\mathbb{SE}(3)$  transformation matrix into  $6 \times 1$  vector,  $\mathbf{log}$  is the log map, and  $(\cdot)(\mathbf{z})$  are corresponding measurements.

2) *Cylinder factors*: First, we define the state vector of the cylinder object model as:  $\ell_i^g = [\mathbf{b}; \mathbf{n}; r]$ , where  $\mathbf{b} = [b_x, b_y, b_z]^\top$  is the origin of the axis ray,  $\mathbf{n} = [n_x, n_y, n_z]^\top$  is the direction of the axis ray, and  $r$  is the radius.

Similarly, the treatment of cylinder factors has been derived in [5]. We calculate the expected measurement and actual measurement of cylinder objects from  $\ell_i^g$  and  $\ell_i^g(\mathbf{z})$ . We define the error function for cylindrical objects as:

$$\mathbf{e}_{\text{cyl}} = \begin{bmatrix} (\mathbf{R}_w^s \mathbf{b} + \mathbf{t}_w^s) - \mathbf{b}(\mathbf{z}) \\ \mathbf{R}_w^s \mathbf{n} - \mathbf{n}(\mathbf{z}) \\ r - r(\mathbf{z}) \end{bmatrix}, \quad (14)$$

where  $(\cdot)(\mathbf{z})$  are corresponding measurements.

3) *Ellipsoid factors*: All objects that are not associated with a cylinder or cuboid shape are represented by ellipsoid landmarks. Note that our ellipsoid model is simplified to have only two dimensions (i.e., the cross-section is a circle) and identity orientation. For each ellipsoid object, the state vector is:  $\ell_i^g = [\mathbf{c}, \mathbf{d}_e]^\top$ , where  $\mathbf{c} = [c_x, c_y, c_z]^\top$  represents 3D position of the centroid, and  $\mathbf{d}_e = [d_r, d_h]^\top$  represents the radius and height.

Given the expected ellipsoid landmark model  $\ell_i^g = [\mathbf{c}, \mathbf{d}_e]^\top$  and the actual measurements which consists of range-bearing measurements  $rg(\mathbf{z}), \theta(\mathbf{z}), \phi(\mathbf{z})$  as well as the dimension measurements  $\mathbf{d}_e(\mathbf{z}) = [d_r(\mathbf{z}), d_h(\mathbf{z})]^\top$ , the measurement error  $\mathbf{e}_{\text{ellip}}$  is derived as follows:

Let  $[c'_x, c'_y, c'_z] = \mathbf{R}_t^\top(\mathbf{c} - \mathbf{t}_t)$  denote the landmark transformed into the body frame,  $rg_{\text{exp}} = \sqrt{c'^x_2 + c'^y_2 + c'^z_2}$  represent the expected range measurement, and  $\theta_{\text{exp}} = \tan^{-1}\left(\frac{c'_y}{c'_x}\right)$  and  $\phi_{\text{exp}} = \tan^{-1}\left(\frac{c'_z}{\sqrt{c'^x_2 + c'^y_2}}\right)$  indicate the expected bearing

measurements. The measurement error for centroids of ellipsoid objects is as follows:

$$\mathbf{e}_{\text{ellip}} = \begin{bmatrix} rg_{\text{exp}} - rg(\mathbf{z}) \\ \theta_{\text{exp}} - \theta(\mathbf{z}) \\ \phi_{\text{exp}} - \phi(\mathbf{z}) \end{bmatrix}. \quad (15)$$

Since the GTSAM solver [61] already provides range and bearing measurement models, we directly utilize these for optimizing the centroid positions. For dimension estimation, we use the moving average method as follows:  $\mathbf{d}_e^{\text{updated}} = (1 - \alpha) \cdot \mathbf{d}_e + \alpha \cdot \mathbf{d}_e(\mathbf{z})$ , where  $\alpha$  controls the weight given to new measurements.

4) *Odometry factors*: The robot state at time  $t$  is represented by  $\mathbf{x}_t$ , which contains the  $\mathbb{SE}(3)$  pose. For each key pose  $\mathbf{x}_t$ , we compute the relative motion from its preceding key pose using the low-level odometry readings from either VIO [19] or LIO [20], depending on the sensor that the robot carries. Specifically, this is calculated as  $\Delta \mathbf{x}_t = (\mathbf{x}_{t-1}^{\text{odom}})^{-1} \circ \mathbf{x}_t^{\text{odom}}$ . Our factor graph incorporates this relative motion as an odometry factor. This approach mitigates the cumulative drift commonly associated with low-level odometry systems by not directly using the low-level odometry as pose priors. Instead, we utilize the odometry estimates to interpolate between consecutive key poses that are proximate in both space and time, where the precision of the low-level odometry provides reliable measurements.

We use the measurement models defined above to convert the observations and build a factor graph with nodes for both robot poses and object landmarks and factors between them, as illustrated in Fig. 3. We implemented our custom cuboid and cylinder factors to be compatible with the GTSAM factor graph optimization library [61]. In addition, we use GTSAM’s built-in relative  $\mathbb{SE}(3)$  pose measurement model for the odometry factor and the range-bearing measurement model for ellipsoid object landmarks.

### E. Place recognition and loop closure

Our system is designed with a modular architecture, enabling users to integrate alternative place recognition and loop closure algorithms to meet their unique requirements. In this paper, we have developed two example algorithms that leverage our object-level metric-semantic map representation. At a high level, they can be summarized as follows:

- 1) *SlideMatch*: This approach uses an exhaustive search strategy by sampling candidate transformations, applying them to the first metric-semantic map, and matching the transformed map against the second map to identify potential loop closures.
- 2) *SlideGraph*: This descriptor-based approach begins by performing data association through the matching of triangle descriptors derived from each metric-semantic map, using both class labels and position information of the objects. The initial associations are then refined using a graph-theoretic framework [36], which effectively eliminates outliers.

The two example algorithms exhibit different behaviors and offer complementary performance across various scenarios.

Generally, we recommend using SlideGraph as the primary algorithm, with SlideMatch serving as an alternative option, as further discussed in detail in Section V-J.

1) *SlideMatch*: The SlideMatch algorithm takes a pair of object-level metric-semantic maps from either the same robot (i.e., historical map and current map for loop closure) or a pair of different robots. SlideMatch algorithm checks within a search region if a valid loop closure is found and outputs the relative transformation between the reference frames of these two maps. The search region of the algorithm encompasses continuous ranges of X and Y positions and yaw angles, where the relative transformation between the two maps may exist. The search procedure is carried out over the search region using a user-defined search resolution for discretization.

*Preprocessing*: For inter-robot place recognition, we designed a preprocessing step that zero-centers the two object-level maps before performing data association. This step reduces the search region and computation of the SlideMatch algorithm, especially when the two robots start far apart. Furthermore, it enables the algorithm to automatically calculate the search region by setting the X and Y search ranges to encompass the region where the two maps overlap and the yaw search range to  $(-\pi, \pi]$ .

*Anytime implementation*: The anytime implementation progressively improves the quality of place recognition until the user-specified compute budget is exhausted. Quality is measured by the number of inlier matches. This approach guarantees the algorithm returns the best estimate within the given computation time budget. A valid loop closure is considered to be found if the matching score of the best estimate exceeds the valid loop closure inlier threshold, which is described in more detail below. This anytime procedure is achieved by gradually increasing the search region for data association until it covers the entire region or the compute budget is used up.

*Data association*: The data association step involves finding the possible set of landmarks that are common in both maps. Candidate transformation samples are generated from the search region by discretizing the search space across the X, Y, and yaw dimensions, based on the user-specified resolutions. Specifically, the algorithm iterates through all possible X and Y position and yaw angle samples within the current region calculated by the anytime process. Within each iteration, the objects in the first map are transformed using the sampled X, Y, and yaw, and matched against the objects in the second map. The algorithm then evaluates the matching score, which is the total sum of all valid object matches between the two maps. Similar to the data association in Section IV-D, a valid object model match must have the same semantic label, and their centroid distance and model difference are within the given threshold. At the end of each iteration, if the matching score exceeds the current best matching score, it will be recorded along with the valid object matches. After all iterations, the best matching score will be compared with the valid loop closure threshold to determine whether a valid loop closure has been identified.

2) *SlideGraph*: The SlideGraph algorithm, unlike the exhaustive search-based SlideMatch algorithm, adopts a

descriptor-based matching approach and combines it with a robust outlier rejection method, CLIPPER [36], for efficient and robust loop closures. While the CLIPPER framework can achieve robust data association by formulating the problem as a maximum clique search in a weighted graph, when an initial data association is unavailable, CLIPPER can still operate under an all-to-all association assumption. However, this quickly becomes too memory demanding and computationally expensive as the number of object landmarks increases in the map.

To address these limitations, we propose a method for generating initial data associations using descriptor matching. Inspired by prior research [42], our descriptor design applies Delaunay triangulation to two sets of object landmarks derived from metric-semantic maps. Candidate simplices, either triangles (2D landmarks) or tetrahedrons (3D landmarks), from each set are compared based on their sorted vertex-to-centroid distances. A valid match is found if the distance metrics between two simplices fall within a predefined threshold. From these matched simplices, the corresponding vertices are extracted as an initial set of object landmark associations. The semantic class labels of the object landmarks are used to further reject false candidate matches. This initial matching step significantly reduces the search space for the CLIPPER framework, which subsequently refines these matches by filtering out false associations through its graph-theoretic optimization. The total number of valid matches resulting from this refinement is treated as the matching score, which is then compared against a threshold to determine whether a valid loop closure has been detected. This two-stage approach ensures robust data association while maintaining computational feasibility, even for large landmark sets.

3) *Least square optimization*: Finally, a least-squares optimization is applied to obtain the relative transformation estimates between the reference frames of the two metric-semantic maps, using the data associations provided by either the SlideMatch or SlideGraph algorithm. Note that we only use estimates for X, Y, Z, and yaw. This is because roll and pitch are directly observable from the IMU, and their estimates from VIO or LIO drift negligibly over time. This optimization aims to minimize the Euclidean distance between pairs of matched landmarks found in the data association step.

#### F. Decentralized graph optimization for multi-robot SLAM

As overviewed in Section III-B, for multi-robot scenarios, each robot will exchange information with other robots to find inter-robot loop closures and perform decentralized factor graph optimization. When communication is established, the robots will share the lightweight observations stored in their databases, which include object models detected by each key pose and the relative odometry between each pair of key poses. Besides, the robots will also share a metric-semantic map containing all object models so that inter-robot loop closure can be effectively performed. Messages received from other robots are saved in the database before an inter-robot loop closure is found. Once the transformation between the reference frames of two robots is found, all the previously

received key poses and object detections from other robots will be transformed into the host robot’s reference frame. In this way, we can add both the pose nodes and the nodes for detected object models associated with them from other robots to the host robot’s factor graph in a similar way as described in Section IV-D. This process is illustrated in Fig. 3. The communication and decentralized graph optimization module is illustrated in Fig. 4. Although each robot needs to perform a pose graph optimization using information from all other robots, the induced burden on computation and communication bandwidth is still small due to the sparsity and memory-efficient representation of object models.

One natural question that may arise is why we share such lightweight observations between robots instead of sharing marginalized factor graphs (e.g., as in [15], [16])? We justify our design choice by the fact that the object-level metric-semantic map that we use is significantly sparser than traditional geometric representations such as point, line, or planar features. In addition, the number of objects detected at each pose is much smaller compared to geometric features (e.g., corner points). In terms of performance, our representation allows us to be memory and computationally efficient, as supported by the experimental results in Table VI. In terms of statistical consistency, our approach does not suffer from spurious information loss due to linearization.

## V. RESULTS AND ANALYSIS

### A. Robot platform overview

Our robot team consists of three types of aerial and ground robots: the Falcon 250 UAV, the Falcon 4 UAV, and the Scarab UGV, as illustrated in Fig. 1. The specifications of these robots are provided in Table III.

The Falcon 250 UAV [3] is a lightweight vision-driven aerial platform capable of autonomous exploration and navigation in cluttered, multi-floor indoor environments. It carries an Intel Realsense D435i RGBD camera, which is the primary sensor

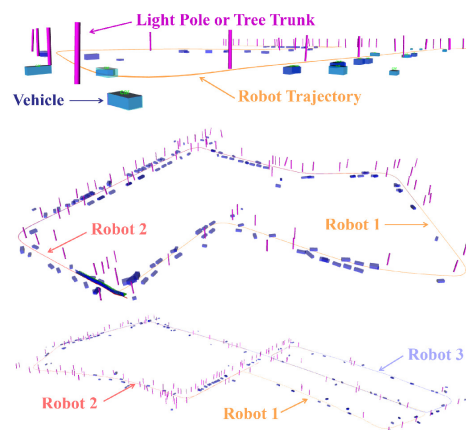


Figure 5: *Metric-semantic SLAM results on the KITTI dataset*. The top, middle, and bottom panels show the results of experiments involving one, two, and three robots, respectively. Each cuboid represents a vehicle while each cylinder represents either a tree trunk or a light pole. The estimated robot trajectories are shown in orange, red, and blue for the first, second, and third robots, respectively.

Table III: *Robot platform details.* Specifications on three types of robot platforms used in our experiments.

	Compute	Primary Sensor	Battery Life	Autonomous Navigation Speed
Falcon 4 UAV	Intel i7-10710U CPU	Ouster OS1-64	30 mins	3-10 m/s
Falcon 250 UAV	Intel i7-10710U CPU	Intel Realsense D435i	7 mins	2-5 m/s
Scarab UGV	Intel i7-8700K CPU	Intel Realsense D455	45 mins	0.5 m/s

for metric-semantic SLAM and autonomous exploration. It is equipped with a VOXL board, which is used for VIO.

The Falcon 4 UAV [8] is equipped with a 3D LiDAR, a VectorNav VN-100 IMU, and a Pixhawk 4 flight controller. The platform has a flight time of  $\sim 30$  minutes with all onboard sensors and computer running.

The Scarab UGV platform is a ground wheeled robot carrying an Intel Realsense D455 camera, which is the primary sensor for metric-semantic SLAM and autonomous exploration. It is also equipped with a 2D Hokuyo LIDAR for low-level odometry and obstacle avoidance.

All modules in our software system, including instance segmentation, metric-semantic SLAM, exploration, planning, and control, as shown in Fig. 3, execute in real time onboard the robots.

### B. Deployment for autonomous exploration

We integrate the metric-semantic SLAM algorithm with our autonomous exploration and navigation systems for all three types of robot platforms. The integration process and the use of semantic maps differ for different sensing modalities. In [5], an active metric-semantic SLAM system with similar map representations is proposed for LiDAR-based UAVs. In contrast, the RGB-D-based UAV and UGV platforms in [3] face larger odometry drifts. To address this, they use the outputs of the metric-semantic SLAM framework to not only explore frontiers to gather more information but also to actively establish semantic loop closures so that the state estimation uncertainty can be reduced. We extend [3] by (a) upgrading the SLAM module with the metric-semantic SLAM framework proposed in this work so that multi-robot collaboration can be achieved, (b) deploying the active metric-semantic SLAM system that was originally designed for Falcon 250 UAV on a UGV platform (i.e., the Scarab UGV platform), and (c) supporting a heterogeneous team of aerial and ground robots to share information and collaboratively construct metric-semantic maps.

### C. Experiment design

We designed and conducted a set of experiments to evaluate our framework and its modules. We conducted four sets of experiments. Here, we provide an overview of each experiment:

1) *Experiment 1:* This experiment is designed to evaluate our decentralized metric-semantic SLAM framework in both indoor and outdoor environments. We utilize the RGBD-equipped Falcon 250 UAV and Scarab UGV, along with the LiDAR-equipped Falcon 4 UAV, each operated by hand-carrying, autonomous navigation, and manual piloting, respectively. These experiments demonstrate the versatility of our metric-semantic SLAM system across heterogeneous platforms and environments. The experimental setup is divided into two distinct sub-experiments.

(1.a) The Falcon 4 UAV is manually piloted across three different parking lots, while the Falcon 250 is hand-carried through four sequential runs inside two buildings. Each of the seven runs starts with a pre-defined pose. To test the capabilities of our semantics-driven place recognition and loop closure algorithms, we create unknown initial poses for each run by streaming the data from different starting timestamps. During data playback, the sensor data from each robot is processed on a base station with an i7-10750H CPU (comparable to CPUs onboard the robots), where each robot operates on a separate ROS node. We enable communication between the ROS nodes of different robots at predetermined intervals, effectively replicating an intermittent and opportunistic communication paradigm. This setup allows us to evaluate the robustness and efficacy of our decentralized metric-semantic SLAM framework across heterogeneous robots and environments.

(1.b) This sub-experiment begins with the Falcon 4 UAV and the Scarab UGV inside a building. The Falcon 4, hand-carried, maps the entire first floor, while the Scarab autonomously navigates and maps the same area. The initial poses of the robots are unknown, posing a challenge in place recognition and map merging. This experiment demonstrates the effectiveness of our semantics-driven place recognition and map merging module in handling data from heterogeneous sensing modalities, including LiDARs and RGBD cameras.

2) *Experiment 2:* A multi-robot simultaneous indoor autonomous exploration experiment where the Falcon 250 and Scarab platforms actively explore the building and jointly construct a metric-semantic map. All robots in the experiment explore the environment fully autonomously and share information through either constantly communicating with each other or communicating with the base station at the end of the mission. The robots leverage the metric-semantic map in real time for their active SLAM systems, which trades off between exploration and uncertainty reduction. This experiment can be further broken down into two sub-experiments.

(2.a) The Falcon 250 and Scarab platforms start at different locations on the first floor of a building. They then autonomously explore the environment to the best of their abilities and build a metric-semantic map. At the end of the mission, they share their information to the base station in an attempt to merge them. The transformation between the initial starting positions of the two robots is unknown. This experiment demonstrates the ability of our system to operate on a heterogeneous team of robots and be efficient and lightweight enough to leave enough computational headroom to support their autonomy stack.

(2.b) Three Scarab platforms are tasked with the operation of autonomously exploring the first floor of a building. They build a metric-semantic map in a decentralized manner and share information through constant communication using Wi-Fi in an attempt to construct a merged map. In addition

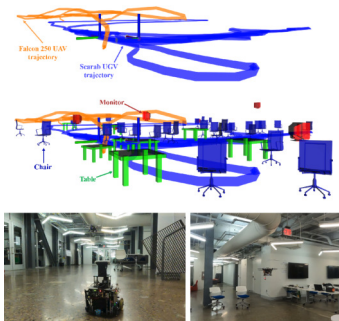


Figure 6: *Exploration and metric-semantic SLAM with heterogeneous robot teams.* This figure shows the trajectories and metric-semantic map constructed from autonomous exploration experiments by aerial and ground robots. The Falcon 250 UAV explores in 3D (orange trajectory), while the Scarab UGV explores in 2D (blue trajectory). Our method can merge maps across UAVs and UGVs by leveraging the viewpoint invariance of semantic landmarks.

to demonstrating efficient operations on resource-constrained robots, this experiment also highlights our multi-robot decentralized metric-semantic SLAM module and how it processes sensor information to facilitate online collaborative SLAM with inter-robot loop closures and map merging.

3) *Experiment 3:* An experiment in forest environments where the Falcon 4 is manually piloted to different sections of a forest for a total of three sequential runs to map all the tree trunks. The Falcon 4 platform starts from the same position for each run (for ground truthing purposes) but the data is processed in a manner that mimics a three-robot simultaneous operation with an unknown initial transformation between the robots. This was achieved by processing the three bags of data and streaming them from different timestamps such that it replicates the situation of three robots starting at various locations with a sufficient translation between them. Forest environments are a steep contrast to urban environments as they are more cluttered with dense and homogeneous object landmarks (i.e. tree trunks). By experimenting in such environments, we showcase our large-scale mapping efficiency and our ability to merge maps from multiple robots in dense and perceptually aliased environments. In addition, obtaining information on semantic objects in forests, such as tree trunks and branches, can enhance silviculture practices, quantifying carbon sequestration, and combating climate change.

4) *Experiment 4:* In the final experiment, we tested the proposed framework on the publicly available Semantic KITTI dataset. This allowed us to demonstrate the algorithm’s versatility and effectiveness in scenarios beyond our custom robot platforms and experimental setups.

#### D. Indoor-outdoor LiDAR-RGBD metric-semantic SLAM without inter-robot loop closures

Fig. 6 shows qualitative results on metric-semantic SLAM for experiment 1.a as described in Section V-C. Outdoor objects such as cars, tree trunks, and light poles and indoor objects such as chairs, tables, and monitors are detected and mapped using our sparse metric-semantic map representation. Object models for vehicles are represented by blue cuboids,

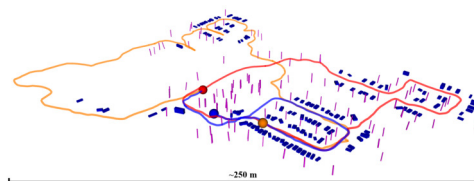


Figure 7: *Multi-robot metric-semantic SLAM in an outdoor experiment.* Results from an experiment involving three Falcon 4 UAV flights. The trajectories, marked by red, blue, and yellow bold lines, start from different positions (red, blue, and yellow dots) but converge back to the same end location (red dot). Our semantics-driven place recognition algorithm was able to detect inter-robot loop closures, which are then used to merge the maps. In the map, vehicle models are represented by blue cuboids, and models for tree trunks or light poles are shown by magenta cylinders.

while tree trunks and light pole landmarks are shown as pink cylinders. The ellipsoid models of chairs, tables, and monitors are replaced with corresponding Computer-Aided Design (CAD) models. This is done only for visualization purposes and an intuitive interpretation of the metric-semantic map, especially in highly cluttered indoor environments. These models are colored blue, green, and red, respectively, according to their detected semantic class. This experiment shows the capabilities of our metric-semantic SLAM framework in integrating data from multiple robots equipped with either LiDARs or RGBD cameras, enabling the construction of a unified metric-semantic map that includes both indoor and outdoor areas.

#### E. Urban and forest LiDAR-only metric-semantic SLAM with inter-robot loop closures

Fig. 7 shows qualitative results on multi-robot metric-semantic SLAM, demonstrating inter-robot loop closure and map merging capabilities in outdoor environments using LiDAR-equipped Falcon 4 UAVs. This experiment utilizes data from experiment 1.a. In this specific setup, the transformation between the robots is unknown and thus has to be estimated to merge their maps. Fig. 10 shows qualitative results for the forest experiments, with the overhead view illustrating that the environment is large-scale, unstructured, and features dense homogeneous objects (i.e. trees). In both experiments, our semantics-driven place recognition algorithm reliably detects inter-robot loop closures while accurately estimating relative transformations between the robots.

The quantitative results on inter-robot localization are shown in Table IV. In urban environments, our algorithm utilizes semantic object models of various classes and shapes to estimate the relative 3D position and yaw between each pair of robots. In forest settings, our method leverages models of tree trunks for inter-robot localization. The SlideMatch algorithm achieves an average position error of 0.13 m and a yaw error of  $0.68^\circ$ , with standard deviations of 0.07 m and  $1.76^\circ$ , respectively. In contrast, the SlideGraph algorithm shows an average position error of 0.24 m and a yaw error of  $-0.57^\circ$ , with standard deviations of 0.28 m and  $1.60^\circ$ , respectively.

Despite significant variations in object classes, appearance, and density when transitioning from urban to forest environ-

Table IV: *Inter-robot localization errors*. SM and SG refer to the SlideMatch and SlideGraph algorithms, respectively. Position error is calculated as the L2 norm of the errors in the X, Y, and Z coordinates of the relative transformation between two robots’ reference frames. In general, the SlideGraph algorithm is more efficient and requires less parameter tuning compared to SlideMatch, while still providing comparable results. However, both algorithms encounter difficulties in scenarios where object landmarks are noisy, particularly with RGBD-based sensing. SlideGraph is relatively more prone to failures in such conditions due to insufficient pairwise consistency in the positions of object landmarks.

Experiment (Sensor)	Position Err. (SM) [m]	Yaw Err. (SM) [°]	Position Err. (SG) [m]	Yaw Err. (SG) [°]
Outdoor-Urban Falcon 4 #0 to Falcon 4 #1 (LiDAR)	0.235	-0.2	0.074	0.3
Outdoor-Urban Falcon 4 #1 to Falcon 4 #2 (LiDAR)	0.159	-0.3	0.240	0.9
Outdoor-Urban Falcon 4 #0 to Falcon 4 #2 (LiDAR)	0.135	-0.9	0.848	0.7
Outdoor-Forest Falcon 4 #0 to Falcon 4 #2 (LiDAR)	0.040	3.1	0.101	0.3
Outdoor-Forest Falcon 4 #1 to Falcon 4 #2 (LiDAR)	0.159	3.2	0.049	-3.0
Outdoor-Forest Falcon 4 #0 to Falcon 4 #1 (LiDAR)	0.054	-0.8	0.126	-2.6
Cross Sensing Modality (LiDAR-RGBD)	0.542	-4.0	0.578	2.0
Indoor Scarab 45 to Scarab 40 (RGBD)	0.221	-3.4	0.952	-2.3
Indoor Scarab 40 to Scarab 41 (RGBD)	0.165	0.0	0.253	0.1
Indoor Scarab 45 to Scarab 41 (RGBD)	0.460	3.7	1.040	-6.1
Indoor Aerial-Ground (RGBD)	0.245	-2.4	–	–

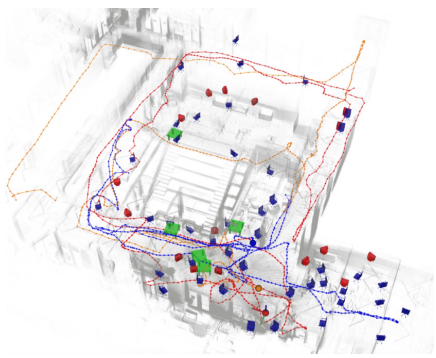


Figure 8: *Metric semantic map constructed in real time onboard three Scarab UGVs*. The UGVs autonomously explore the environment while communicating with each other. Blue, orange and red trajectories correspond to three trajectories from the three UGVs. Our semantic place recognition algorithm was able to perform inter-robot loop closure, which is used to construct a merged metric-semantic map. The gray-colored accumulated point cloud is not generated by these UGVs but by a LiDAR robot for visualization purposes. The figure overlays the trajectories with the metric-semantic map to qualitatively assess the validity of the results.

ments, both algorithms consistently demonstrate robust and accurate performance. Furthermore, the SlideGraph algorithm requires less parameter tuning and, unlike SlideMatch which relies on a computationally expensive exhaustive search, is more efficient.

#### F. Indoor RGBD-only aerial-ground metric-semantic SLAM with inter-robot loop closures

In this experiment, the starting position and orientation of the robots vary significantly, with up to 6 m in position and a 90° difference in yaw. Fig. 8 shows the results constructed by merging sensor measurements from three UGVs autonomously exploring the environment. Our semantics-driven place recognition algorithm was able to detect and accurately estimate relative transformations as robots accumulate observations. Based on the estimated relative transformations, our metric-semantic factor graph fuses the information from multiple robots to construct a merged metric-semantic map.

The quantitative results are shown in Table IV. For the indoor experiment with multiple Scarab UGVs, our SlideMatch

algorithm achieves an average position error of 0.28 m and a yaw error of 0.10°, with standard deviations of 0.13 m and 2.90°, respectively. Our SlideGraph algorithm achieves an average position error of 0.75 m and a yaw error of -2.77°, with standard deviations of 0.35 m and 2.55°, respectively.

Fig. 6 shows the results constructed by the Falcon 250 and one of the UGVs that autonomously explore the environment. The invariance of semantic object models to viewpoint changes enables our algorithm to merge measurements from different viewpoints as observed by heterogeneous aerial and ground robots. As shown in the last row of Table IV, the SlideMatch algorithm successfully identifies inter-robot loop closures and estimates the relative transformation with a position error of 0.245 m and a yaw error of 2.35°. However, we observed that aerial-ground localization exhibits slightly larger errors along the Z-axis compared to localization errors among identical robot platforms. This discrepancy arises because aerial robots primarily observe the upper parts of objects, while ground robots focus on the lower parts, leading to greater variation in Z-axis estimates. In contrast, the SlideGraph algorithm fails to perform inter-robot loop closures between RGBD-based aerial and ground robots. This limitation is caused by reduced pairwise consistency between object positions in the map, which stems from noisy RGB-D sensing combined with the fact that the aerial and ground robots observe different parts of the objects.

In summary, while noisy RGBD sensing can introduce inaccuracies in the localization and modeling of individual objects, a collection of object landmarks provides the robots with an informative description of the environment. This enables them to establish loop closures even under drastic viewpoint changes. In addition, it is important to note that the increased noise levels require a careful selection of parameters, such as the search discretization in SlideMatch or the descriptor matching threshold in SlideGraph.

#### G. Indoor LiDAR-RGBD metric-semantic SLAM with inter-robot loop closures

Fig. 9 shows qualitative results on multi-robot metric-semantic SLAM on the cross sensing modality experiment (i.e., experiment 1.b in Section V-C). The accumulated point

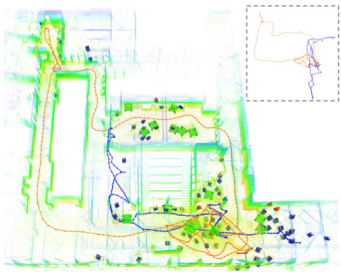


Figure 9: *Cross sensing modality place recognition.* A LiDAR-equipped robot is represented by the orange trajectory and an RGBD-based robot by the blue trajectory. The rest of the panels show the merged metric-semantic map constructed by the robots and overlaid on top of the accumulated LiDAR point cloud. The upper-right panel illustrates the robots’ trajectories in their own reference frames. This comparison shows how the place-recognition and loop-closure algorithm effectively registers the two robots with different sensing modalities into a common reference frame despite the drastic differences in raw sensor data and the initial poses.

cloud captured by the LiDAR robot is overlaid on the semantic object models mapped by both the LiDAR and the RGBD robots. Note that there is a relative transformation of 2.51 m along X, -5.37 m along Y, and  $90^\circ$  along yaw between the reference frames of the two robots. Our algorithm leverages the invariance of semantic object models across sensing modalities. This is a unique advantage of our method given that geometric features extracted from RGB or depth images significantly differ from those derived from LiDAR point clouds. This capability enables accurate estimation of the transformation between robots equipped with different sensors—specifically *LiDAR* and *RGBD*—allowing us to fuse their measurements and create a merged metric-semantic map.

Quantitative results are shown in Table IV. Our semantics-driven place recognition algorithms demonstrate the ability to establish loop closures between robots equipped with different sensors. SlideMatch estimates the relative transformation with a position error of 0.542 m and a yaw error of  $-4.00^\circ$ , while SlideGraph results in a position error of 0.578 m and a yaw error of  $2.00^\circ$ . The accuracy of inter-robot localization in cross-sensing-modality scenarios is slightly lower compared to single-modality experiments, particularly those involving LiDAR-based robots. This performance decline can be attributed to differences in fields of view, object detection accuracy, and noise levels between LiDAR and RGBD sensors. These factors lead to greater discrepancies in object modeling, which, in turn, reduce the precision of inter-robot localization.

#### H. Object mapping quantitative results

We report precision, recall, and F1 scores on object mapping results of both cars and tree trunks across multiple experiments. We first obtain the ground truth by surveying the environment and manually marking down the positions of objects. Here, we focus the object mapping results on cars and tree trunks since ground truth for them can be reliably acquired when compared to other objects such as chairs, tables or monitors. This is because cars are usually parked in clearly separated parking spaces and tree trunks are often spaced apart

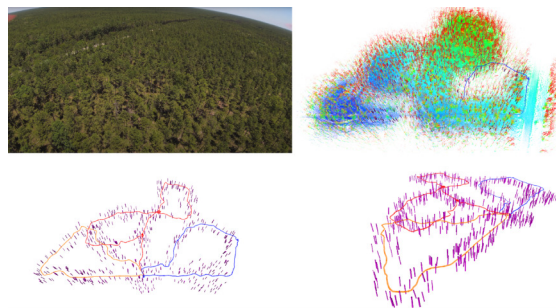


Figure 10: *Forest experiments.* This figure illustrates the overhead UAV view of the environment (top left), metric-semantic map with point clouds (top right), and object-based map constructed by three under-canopy UAV flights (bottom row). Three robots with their trajectories colored in red, yellow, and blue can merge their information to the starting location with unknown initial transformations between their starting poses. The figure shows metric-semantic maps of tree trunks (modeled as cylinders) overlaid on top of an aggregated point cloud showing the forest environment. The trajectory of each robot, in this case, is  $\sim 1$  km in length, and 860 trees have been mapped. The mapped trees span an area of  $\sim 160,000$   $m^2$ .

at regular intervals. A True Positive (TP), or a valid object match, occurs when an object in our metric-semantic map is confirmed to exist at a position that aligns with the ground truth. A False Positive (FP) refers to an object that appears in our metric-semantic map but is not present in the ground truth. Conversely, a False Negative (FN) refers to an object that is present in the ground truth but does not appear in our metric-semantic map. These values are then used to calculate precision, recall, and F1 scores. The results can be found in Section V-H. Note that those results are acquired using the smallest backbone (darknet-13 as detailed in [5]) for the semantic segmentation model, which is less accurate but can run inference in real time using the onboard NUC computer. The results can be further improved using larger backbones such as darknet-26 or darknet-53, which are supported in [62]. As shown in Section V-H, the average precision for cars is 0.967 and for tree trunks is 0.946. The average recall for cars is 0.881 and for tree trunks is 0.909. The F1 score is 0.922 for cars and 0.927 for tree trunks. In summary, through these metrics, we demonstrate that our method is able to reliably detect and map objects that exist in the environment, with very few false positives or false negatives. This result is important for downstream tasks such as inventory estimation.

Table V: *Quantitative results of object detection: Ground Truth (G.T.) vs. Estimated (Est.).* Precision is evaluated by determining the proportion of detected objects that accurately match the ground truth, thus measuring the algorithm’s object mapping accuracy. Recall indicates the likelihood of the algorithm successfully detecting an object present in the environment, reflecting the algorithm’s object mapping sensitivity.

Envir.	G.T. Cars	Est. Cars	Precision	Recall	F1
Lot 1	42	35	1.00	0.833	0.909
Lot 2	61	59	0.932	0.902	0.917
Lot 3	32	29	1.00	0.906	0.951
Total	135	123	0.967	0.881	0.922
Envir.	G.T. Trees	Est. Trees	Precision	Recall	F1
Total	77	74	0.946	0.909	0.927

## I. Communication usage and runtime analysis

1) *Discussion on scalability:* For the factor graph optimization, suppose we have  $G$  variables for each object landmark and  $B$  total object landmarks. The total number of variables for all landmarks is  $S = G \cdot B$ . When adding a new factor, the number of variables that need to be solved for is  $G$  for a new object factor, and 9 for a new odometry (i.e.,  $\mathbb{SE}(3)$  relative pose) factor in the factor graph optimization process [60].  $G$  is a small number, and specifically in our formulation, the variables corresponding to each cuboid, cylinder, and ellipsoid semantic landmark node are 9, 9, and 5, respectively. Therefore, this complexity becomes  $\mathcal{O}(1)$  for adding any new factor. Upon loop closure, the optimization runs in time  $\mathcal{O}(\max(F, S)^{1.5})$  [60]. Practically, in traditional SLAM systems that are based on geometric features, the number of features usually far exceeds the number of robot poses. As a result,  $\max(F, S) = S$ , which is usually at the order of  $100F$  (i.e. each key frame tracks 100 features) or even larger. Due to the use of sparse semantic landmarks in the proposed system,  $S$  will usually be of the order  $\mathcal{O}(F)$ . Therefore, to sum up, the complexity of factor graph optimization is  $\mathcal{O}(1)$  during normal operation and  $\mathcal{O}(F^{1.5})$  during loop closures.

When running with multi-robot teams, the framework runs in a decentralized manner; therefore, for each robot, the complexity scales linearly with the size of the robot team (that is,  $K$ ). As a result, the complexity of decentralized factor graph optimization is  $\mathcal{O}(K)$  when adding new factors coming from the robot team, and  $\mathcal{O}((K \cdot F)^{1.5})$  during loop closures. In practice, the size of the robot team  $K$  is much smaller than the number of poses for each robot  $F$ ; therefore, the decentralized factor graph optimization can be executed in real-time onboard the robot.

2) *Experimental results:* In this section, we show the computational and communication usage when running our metric-semantic SLAM framework. All results are obtained using a computer equipped with an Intel i7-10750H CPU, without the support of a GPU. This CPU's performance is comparable to that of the onboard computers in the robots. In Table VI, the average and maximum size of communication packets are summarized when robots operate with intermittent communication. To replicate intermittent communication, each robot publishes ROS messages to each other every 10 seconds. Benefiting from the sparse metric-semantic map representation, our algorithm is shown to be communication efficient when deployed on real robots. The similarity in communication usage across different robots stems from the fact that we replicate all robots as being within communication range by streaming their data on different ROS nodes running on a laptop. For indoor experiments, communication is facilitated via Wi-Fi. Although each robot shares all the information in its database with neighboring robots within the communication range, we can further reduce the communication burden by implementing a bookmarking system that only publishes new information since the last communication between robots.

Additionally, the runtime for factor adding and graph optimization, and loop closure (if applicable), is reported in Table VI. Except for the loop closure module, which is used to

identify inter-robot loop closures and operates on a separate CPU thread, all other modules are triggered each time new observations arrive.

The front-end runtime depends on the object segmentation and detection model. Our results show that on a *single thread* of Intel i7-10750H CPU, the Rangenet++ [62] front-end with our custom darknet-13 model runs at 1.5 hertz (Hz). The front-end with the YOLOv8 [63], specifically, YOLOv8m, runs at 1 Hz per CPU thread. When multiple CPU threads are used, the inference speed can be increased. For example, YOLOv8m can run at 3 Hz using 4 out of 12 CPU threads. The cuboid, cylinder, and ellipsoid modeling process runs at the corresponding object detection rate.

The back-end factor graph adding and graph optimization processes show different patterns across different environments. For the indoor environment, the average runtime of graph optimization is 2~4 ms, with 68 landmarks. The outdoor urban environment exhibits an average runtime of graph optimization of 10.14 ms, with 249 landmarks. The average time per landmark in this setting is 0.041 ms. Due to the increased density of landmarks in the outdoor forest environment, the average runtime spikes to 36.46 ms, with a significantly higher number of landmarks at 860.

These results illustrate that the average runtime required for factor graph adding and graph optimization increases with the number of landmarks. This relationship is influenced by the presence of inter-robot loop closures in these experiments, which makes the computational complexity lie between observing new landmarks and processing loop closures.

The runtime also increases with the trajectory length, which is partly due to more pose nodes being added to the graph. However, the correlation between trajectory length and runtime is not as strong as the correlation between the number of landmarks and runtime. An increase in the runtime per meter traveled by the robot is observed as the environment transitions from less complex settings (indoor) to more complex ones (outdoor forest). This trend also shows the challenges and computational costs associated with large-scale environments.

Although the runtime of the semantics-driven place recognition and loop closure algorithms is longer than that of factor graph optimization, the resulting latency is acceptable. In addition, since the semantics-driven place recognition and loop closure algorithm operate in a separate CPU thread, this latency does not directly impact real-time localization and navigation. Once the computation is complete, the estimated loop closure information can be used by the factor graph to update both the map and the robot trajectories.

In addition to the importance of *sparse semantic* landmarks in facilitating long-range autonomous exploration and navigation, our results on communication provide another reason for using semantic landmarks in multi-robot collaboration tasks. The average communication per landmark is 3.96 kilobytes (KB) for indoor environments, 1.84 KB for outdoor environments, and 2.03 KB for forest environments. The average communication bandwidth per meter traveled by the robot is 1.65 KB in indoor settings, 1.85 KB in outdoor settings, and 5.13 KB in forest settings. As the density of landmarks in the environment increases, communication overhead also

Table VI: *Runtime and communication results.* The runtime for factor addition and graph optimization accounts for handling both the observation of new landmarks and loop closure scenarios. The runtime of semantics-driven place recognition and loop-closure algorithms includes both loop closure detection and relative transformation estimation. Note that the place-recognition and loop-closure process runs on a separate thread. Therefore, its latency does not impact real-time operation. SlideGraph requires significantly lower computational runtime, particularly in environments with fewer object landmarks. This efficiency is attributed to the use of descriptor-based matching, where computational demands are influenced not by the scale of the environment but only by the number of object landmarks. In contrast, SlideMatch exhaustively searches across the entire search region, and its runtime depends on various factors, such as the search discretization. The communication usage is measured by the size of messages published by each robot. Here, the average usage is calculated by dividing the total size of all messages by the number of messages, while the maximum size corresponds to the size of the last messages since they usually contain the most observations.

	Number of Objects	Robot ID	Trajectory Length [m]	Runtime [ms]		Communication Usage [MB]	
				Factor Adding and Graph Optimization	Inter-Robot Loop Closure SlideGraph / SlideMatch	Average Communication Usage	Maximal Communication Usage
Indoor	68	0	172.072	2.429	22 / 3638	0.304	0.578
		1	155.930	4.008	24 / 1614	0.304	0.575
		2	194.625	2.372	28 / 2252	0.309	0.576
Outdoor Urban	249	0	699.543	9.351	95 / 131	0.458	0.725
		1	264.264	7.763	86 / 123	0.453	0.727
		2	410.194	13.296	91 / 168	0.466	0.725
Outdoor Forest	860	0	326.476	28.702	209 / 1124	1.702	2.686
		1	367.798	28.638	406 / 2091	1.735	2.687
		2	746.870	52.046	375 / 1836	1.742	2.686

increases. However, the overall communication bandwidth required remains minimal. This modest communication demand is particularly important when robots operate under intermittent and bandwidth-limited communication conditions. In addition, this efficiency in communication also shows the effectiveness of *sparse semantic* mapping in minimizing resource usage.

To sum up, the results show that, although there is some variance across different environments due to the difference in density of objects, the use of *sparse semantic* map representations offers the robot a rich understanding of its surroundings while maintaining efficiency in computation, storage, and communication. Our proposed method not only facilitates real-time execution of exploration and navigation tasks but also enables multi-robot collaboration across various environments on a large scale.

#### J. Analysis of SlideMatch and SlideGraph algorithms

Throughout these experiments, we have observed unique advantages, limitations, and deployment considerations for both the SlideGraph and SlideMatch algorithms. SlideGraph demonstrates superior performance with minimal parameter tuning in scenarios where the object-level metric-semantic map experiences relatively low noise in object position estimates. Therefore, SlideGraph is generally the preferred option, particularly for LiDAR-based experiments or scenarios involving large search spaces. On the other hand, the SlideMatch algorithm can sometimes achieve successful loop closure in situations where the SlideGraph method fails, particularly in experiments with large errors in object position estimates. This is because the pairwise consistency assumption used in [36] becomes less suitable under such conditions. Such situations are common in indoor RGBD-based object mapping, where cluttered spaces lead to small differences in object-to-object distances and high noise in position estimates due to noisy depth measurements, noisy local odometry, limited field of view, and complex object shapes. Although semantic labels can help disambiguate these measurements to some

extent, it remains challenging to use pairwise consistency to distinguish outliers from inliers. In contrast, SlideMatch leverages the entire map and operates similarly to a *voting* method, making it more tolerant to noise in local pairwise object distances. As a result, SlideMatch performs better in such worst-case scenarios, albeit at the cost of significant computational overhead.

#### K. Benchmarks

In this section, we present a series of benchmark experiments evaluating our system’s performance in terms of localization accuracy, computational efficiency, and memory usage. We report results from (1) evaluating our framework on standard public datasets and (2) comparing it with other open-source methods. We note that benchmarking the entire system is challenging due to the differences in components across other metric-semantic SLAM systems. Therefore, we opt to isolate and benchmark key modules instead. This allows for a fair and focused comparison of our system.

1) *Benchmark on Semantic KITTI*: In addition to conducting experiments with our own robots, we also tested our algorithm framework on the publicly available Semantic KITTI dataset. The framework takes in the point clouds, initial pose estimation from SuMa++ [24], and ground truth segmentation, and estimates the metric-semantic map as well as the robot trajectory.

The qualitative results, as shown in Fig. 5, demonstrate the ability of the proposed framework to construct metric-semantic maps and merge measurements from multiple robots. We conducted a quantitative evaluation of our SlideGraph algorithm for inter-robot loop closure using sequences #5 and #7. Sequence #7 was divided into three intervals, i.e., (0, 600), (400, 1000), and (500, 1100), for the first, second, and third robots, respectively. Sequence #5 was similarly divided into three intervals, i.e., (0, 1000), (700, 2000), and (1600, 2760), to create datasets for each of the three robots. Our algorithm successfully identified inter-robot loop closures and accurately estimated the relative transformations between the

Table VII: *Benchmark comparison between SlideSLAM (SlideGraph) and existing methods.* For these benchmarks, we report results between robot 0 and robot 1, as referenced by the Semantic KITTI experiments in Section V-K1 and forest experiments in Section V-E. Note that the map sizes for both CLIPPER and SlideSLAM are the same, as they use the same object maps as inputs. Also, note that for the memory and runtime benchmarks of LCDNet, the results reported are the memory consumed for each individual point cloud and the runtime to process a pair of point clouds. In practice, it scales linearly with the number of point clouds. Cells filled with a “-” indicate that the algorithm failed.

Metric	Trans. Err (m)			Abs. Yaw. Err (degree)			Map / Input Size			Runtime (ms)		
	CLIPPER	LCDNet	Ours	CLIPPER	LCDNet	Ours	CLIPPER (full maps)	LCDNet (per scan)	Ours (full maps)	CLIPPER (CPU)	LCDNet (per scan pair on GPU)	Ours (CPU)
KITTI 05	0.04	0.09	0.03	0.04	0.22	0.08	7.2 KB	0.84 MB	7.2 KB	152	1580	40
KITTI 07	0.06	0.05	0.09	0.01	0.21	0.09	7.3 KB	0.84 MB	7.3 KB	202	1580	50
Forest	-	1.10	0.13	-	0.1	2.6	16 KB	0.84 MB	16 KB	-	1580	209

robots. For sequence #5, the SlideGraph algorithm results in a position error of 0.03 m and 0.05 m, and a rotation error of  $0.080^\circ$  and  $0.065^\circ$  for the transformations between robot 0 and robot 1, and between robot 1 and robot 2, respectively. In sequence #7, the algorithm results in a position error of 0.093 m and 0.095 m, and a rotation error of  $0.096^\circ$  and  $0.097^\circ$ . For sequence #5 and sequence #7, the total trajectory lengths of three robots are 2.8 km and 1.2 km. The absolute trajectory errors of the trajectories of all three robots are 0.998 m (vs. 1.154 m from SuMa++) and 0.962 m (vs. 1.148 m from SuMa++), respectively. The relative translation errors are 0.111% and 0.255%. The relative rotation errors are 0.0011 degrees/m and 0.00256 degrees/m.

These results not only demonstrate the accuracy of inter-robot localization but also illustrate the effectiveness of the proposed algorithm framework when deployed on public datasets, showing its potential as a plug-and-play tool for multi-robot metric-semantic SLAM.

#### 2) Benchmark against existing loop closure methods:

To provide a comprehensive comparison, we benchmark SlideSLAM against two existing approaches from the literature. These include: (1) LCDNet [39], a deep-learning-based loop closure method for LiDAR data; and (2) CLIPPER [36], a graph-theoretic algorithm for data association.

LCDNet performs scan-to-scan place recognition using a learned 3D feature extractor and registration network. We benchmark the algorithm using weights that are pretrained on the KITTI dataset. We use the same sequences and splits of Semantic KITTI as described in the previous benchmark. We also benchmark LCDNet on our forest data, as described in the previous sections. The benchmarks for LCDNet were conducted on a GPU-equipped desktop (with an RTX 3070 GPU). The inter-robot localization results are shown in Table VII. The results are comparable in the KITTI environment. However, for the forest environment, SlideSLAM provides better localization results. This shows the advantage of our method to generalize to different environments as long as the objects are detected. Our method requires no additional fine-tuning data for the loop closure step, whereas LCDNet does. Moreover, we also benchmark the memory requirements to store the observations or maps required for the loop closure step. For LCDNet, these representations can be individual processed point cloud scans (0.84 MB), point features extracted from the PV-RCNN network for each point cloud scan (4.18 MB), or the final global descriptors of those features for each point cloud scan (10.0 MB). As a result, enabling inter-robot loop closure detection with LCDNet requires a minimum bandwidth of 0.84 MB per LiDAR scan. In contrast, as shown in the table, the entire object map of SlideSLAM requires less than

10 KB for the KITTI sequences, which contain hundreds of LiDAR scans. The sparsity of our map representation offers a key advantage for large-scale multi-robot experiments, as it significantly reduces both computational and communication requirements compared to methods operating on less compact map representations.

CLIPPER [36] serves as the data association backbone of our SlideSLAM (SlideGraph) module. However, the original CLIPPER lacks semantic information and relies on exhaustive pairwise matching over all object pairs, resulting in high computational demands in complex scenes (e.g., each of our forest maps consists of hundreds of landmarks). Our SlideGraph enhances CLIPPER by using semantic class labels and polygon-descriptor-based initial data association, enabling us to significantly reduce the hypothesis space. CLIPPER is benchmarked on similar sequences on Semantic KITTI as detailed in the previous section, and also on our forest dataset. The results are shown in Table VII. In challenging environments such as forests, the original CLIPPER algorithm encountered memory overflows due to the large number of landmarks, which significantly expanded the hypothesis space. Take the forest environment as an example: a pair of maps with approximately 300 and 800 landmarks would generate around 240,000 hypotheses. In contrast, our SlideGraph algorithm prunes this raw hypothesis space down to approximately 3,000, significantly reducing the size of the consistency graph by 2 orders of magnitude. As a result, SlideGraph consistently achieved robust performance with lower memory usage and shorter runtimes.

## VI. DISCUSSION ON LIMITATIONS AND FUTURE WORK

In this section, we address a few limitations of the current system and how they motivate our future work. Firstly, the proposed sparse map representation is limited to three simple shapes for modeling objects. While it helps maintain sparsity, the map may not always be visually informative with these shapes, and they may fail to model the finer metric details for objects with complex shapes, which are needed for downstream tasks such as robot manipulation. In addition, they may lead to information loss that results in inaccuracies or failures to identify loop closures, especially in environments with ambiguous or repetitive object layouts. Semantic aliasing may also occur when different objects are represented by the same primitive, increasing dependence on accurate semantic labeling. Misclassifications in such cases can cause errors. We are interested in investigating better map representations that can model generic objects in detail while still maintaining relative sparsity. Secondly, the current inter-robot loop closure strategy does not follow a conventional “filtering”

approach, where multiple loop closures are first established and then a consensus is formed to choose the best loop closure result and filter out false positives. Currently, we only use a sufficient inlier threshold to filter out false positives and establish loop closure in a “one-shot” manner. This may not always be effective in certain situations where the environment is cluttered with the same semantic landmarks and similar pairwise distances (e.g., two conference rooms with similar table and chair arrangements, or dense forests). Therefore, we would like to employ a filtering-based approach to our current loop closure strategy, where the best loop closure candidate is chosen based on a maximum consensus of potential loop closure options. Lastly, our current place recognition and mapping formulation doesn’t account for dynamic objects. Such objects (e.g., a car changing its parking spot over time) violate the pairwise distance consistency assumption required for robust place recognition and can therefore induce faulty loop closures. In the future, we would like to leverage semantic knowledge of objects, combined with intelligent agents such as Large Language Models (LLMs), to better handle dynamic objects. Using the semantic knowledge, an LLM can infer which objects can be dynamic in nature and automatically filter them out during place recognition.

## VII. CONCLUSION

In this work, we introduce a real-time decentralized metric-semantic SLAM framework, along with its integration into autonomous exploration and navigation systems for aerial and ground robots. This integrated system enables teams of heterogeneous robots to autonomously explore and collaboratively construct maps with both geometric and open-vocabulary semantic information across a variety of indoor and outdoor environments. The robots opportunistically leverage communication to exchange sparse, lightweight semantic measurements, which are used for inter-robot localization and map merging. Through a comprehensive set of real-world experiments involving three types of aerial and ground robots, as well as benchmarks on publicly available datasets and comparisons against existing methods, we demonstrate the capabilities, effectiveness, and robustness of our system. Our runtime and communication analysis also show the importance of using sparse semantic map representations, especially for large-scale (over 1 km) metric-semantic mapping tasks in challenging and dense environments, such as forests with over 1000 object landmarks.

## VIII. ACKNOWLEDGMENT

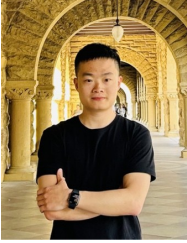
We thank Jie Mei for his help with experiments, Guilherme V. Nardari, Ian D. Miller, Kashish Garg, Jeremy Wang, and Alex Zhou for the software and hardware support.

## REFERENCES

- [1] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, “Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam,” *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.
- [2] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, “Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping,” in *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2020, pp. 5135–5142.
- [3] Y. Tao, X. Liu, I. Spasojevic, S. Agarwal, and V. Kumar, “3d active metric-semantic slam,” *IEEE Robotics and Automation Letters*, 2024.
- [4] Y. Chang, N. Hughes, A. Ray, and L. Carlone, “Hydra-multi: Collaborative online construction of 3d scene graphs with multi-robot teams,” *arXiv preprint arXiv:2304.13487*, 2023.
- [5] X. Liu, A. Prabhu, F. Cladera, I. D. Miller, L. Zhou, C. J. Taylor, and V. Kumar, “Active metric-semantic mapping by multiple aerial robots,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 3282–3288.
- [6] Y. Wu, Y. Zhang, D. Zhu, Z. Deng, W. Sun, X. Chen, and J. Zhang, “An object slam framework for association, mapping, and high-level tasks,” *IEEE Transactions on Robotics*, 2023.
- [7] Y. Tian, Y. Chang, F. Herrera Arias, C. Nieto-Granda, J. P. How, and L. Carlone, “Kimera-multi: Robust, distributed, dense metric-semantic slam for multi-robot systems,” *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2022–2038, 2022.
- [8] X. Liu, G. V. Nardari, F. C. Ojeda, Y. Tao, A. Zhou, T. Donnelly, C. Qu, S. W. Chen, R. A. Romero, C. J. Taylor, *et al.*, “Large-scale autonomous flight with real-time semantic SLAM under dense forest canopy,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5512–5519, 2022.
- [9] M. Shan, Q. Feng, and N. Atanasov, “Orevio: Object residual constrained visual-inertial odometry,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5104–5111.
- [10] S. Yang and S. Scherer, “Cubeslam: Monocular 3-d object slam,” *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 925–938, 2019.
- [11] L. Nicholson, M. Milford, and N. Sünderhauf, “Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam,” *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 1–8, 2019.
- [12] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. I. Christensen, and F. Dellaert, “Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models,” 2017.
- [13] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, “Probabilistic data association for semantic slam,” in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 1722–1729.
- [14] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, “Slam++: Simultaneous localisation and mapping at the level of objects,” in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1352–1359.
- [15] A. Cunningham, M. Paluri, and F. Dellaert, “Ddf-sam: Fully distributed slam using constrained factor graphs,” in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 3025–3030.
- [16] A. Cunningham, V. Indelman, and F. Dellaert, “Ddf-sam 2.0: Consistent distributed smoothing and mapping,” in *2013 IEEE international conference on robotics and automation*. IEEE, 2013, pp. 5220–5227.
- [17] 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*, vol. 40, no. 12-14, pp. 1510–1546, 2021.
- [18] Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, B. Sen, A. Agarwal, C. Rivera, W. Paul, K. Ellis, R. Chellappa, *et al.*, “Concept-graphs: Open-vocabulary 3d scene graphs for perception and planning,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 5021–5028.
- [19] “ModalAI VOXL® Flight Deck.” [Online]. Available: <https://www.modalai.com/products/voxl-flight-deck>
- [20] C. Bai, T. Xiao, Y. Chen, H. Wang, F. Zhang, and X. Gao, “Faster-lio: Lightweight tightly coupled lidar-inertial odometry using parallel sparse incremental voxels,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4861–4868, 2022.
- [21] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age,” *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.
- [22] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto, “Volumetric instance-aware semantic mapping and 3d object discovery,” *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3037–3044, 2019.
- [23] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger, “Fusion++: Volumetric object-level slam,” in *2018 International Conference on 3D Vision (3DV)*, 2018, pp. 32–41.

- [24] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "Suma++: Efficient lidar-based semantic slam," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4530–4537.
- [25] D. Maturana, P.-W. Chou, M. Uenoyama, and S. Scherer, "Real-time semantic mapping for autonomous off-road navigation," in *Field and Service Robotics*, M. Hutter and R. Siegwart, Eds. Cham: Springer International Publishing, 2018, pp. 335–350.
- [26] I. D. Miller, A. Cowley, R. Konkimalla, S. S. Shivakumar, T. Nguyen, T. Smith, C. J. Taylor, and V. Kumar, "Any way you look at it: Semantic crossview localization and mapping with lidar," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2397–2404, 2021.
- [27] I. D. Miller, F. Cladera, T. Smith, C. J. Taylor, and V. Kumar, "Stronger together: Air-ground robotic collaboration using semantics," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9643–9650, 2022.
- [28] S. Chen, G. Nardari, E. Lee, C. Qu, X. Liu, R. Romero, and V. Kumar, "Sloam: Semantic lidar odometry and mapping for forest inventory," *IEEE Robotics and Automation Letters*, vol. 5, pp. 1–1, 01 2020.
- [29] N. Atanasov, S. L. Bowman, K. Daniilidis, and G. J. Pappas, "A unifying view of geometry, semantics, and data association in slam," in *IJCAI*, 2018, pp. 5204–5208.
- [30] Q. Feng, Y. Meng, M. Shan, and N. Atanasov, "Localization and mapping using instance-specific mesh models," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 4985–4991.
- [31] J. Fu, Y. Du, K. Singh, J. B. Tenenbaum, and J. J. Leonard, "Neuse: Neural se (3)-equivariant embedding for consistent spatial understanding with objects," *arXiv preprint arXiv:2303.07308*, 2023.
- [32] 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*, 2021, pp. 6229–6238.
- [33] 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*, 2022, pp. 12786–12796.
- [34] J. G. Mangelson, D. Dominic, R. M. Eustice, and R. Vasudevan, "Pairwise consistent measurement set maximization for robust multi-robot map merging," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 2916–2923.
- [35] H. Yang, J. Shi, and L. Carlone, "Teaser: Fast and certifiable point cloud registration," *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 314–333, 2020.
- [36] P. C. Lusk, K. Fathian, and J. P. How, "CLIPPER: A graph-theoretic framework for robust data association," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13828–13834.
- [37] M. Leordeanu and M. Hebert, "A spectral technique for correspondence problems using pairwise constraints," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2. IEEE, 2005, pp. 1482–1489.
- [38] X. Xu, S. Lu, J. Wu, H. Lu, Q. Zhu, Y. Liao, R. Xiong, and Y. Wang, "Ring++: Roto-translation invariant gram for global localization on a sparse scan map," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4616–4635, 2023.
- [39] D. Cattaneo, M. Vaghi, and A. Valada, "Lcdnet: Deep loop closure detection and point cloud registration for lidar slam," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2074–2093, 2022.
- [40] A. Gawel, C. D. Don, R. Y. Siegwart, J. I. Nieto, and C. Cadena, "X-view: Graph-based semantic multiview localization," *IEEE Robotics and Automation Letters*, vol. 3, pp. 1687–1694, 2017. [Online]. Available: <https://api.semanticscholar.org/CorpusID:3337551>
- [41] X. Guo, J. Hu, J. Chen, D. Fuqin, and T. L. Lam, "Semantic histogram based graph matching for real-time multi-robot global localization in large scale environment," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, 02 2021.
- [42] G. V. Nardari, A. Cohen, S. W. Chen, X. Liu, V. Arcot, R. A. Romero, and V. Kumar, "Place recognition in forests with urquhart tessellations," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 279–286, 2020.
- [43] D. Galvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.
- [44] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A real-time spatial perception system for 3d scene graph construction and optimization," 2022.
- [45] R. Dubé, A. Gawel, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena, "An online multi-robot slam system for 3d lidars," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 1004–1011.
- [46] Y. Chang, K. Ebadi, C. Denniston, M. Ginting, A. Rosinol, A. Reinke, M. Palieri, J. Shi, A. Chatterjee, B. Morrell, A.-a. Agha-mohammadi, and L. Carlone, "Lamp 2.0: A robust multi-robot slam system for operation in challenging large-scale underground environments," *IEEE Robotics and Automation Letters*, vol. 7, pp. 1–8, 10 2022.
- [47] P.-Y. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "Door-slam: Distributed, online, and outlier resilient slam for robotic teams," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, 01 2020.
- [48] Y. Huang, T. Shan, F. Chen, and B. Englot, "Disco-slam: Distributed scan context-enabled multi-robot lidar slam with two-stage global-local graph optimization," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1150–1157, 2022.
- [49] P.-Y. Lajoie and G. Beltrame, "Swarm-slam: Sparse decentralized collaborative simultaneous localization and mapping framework for multi-robot systems," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 475–482, 2024.
- [50] T. Cieslewski, S. Choudhary, and D. Scaramuzza, "Data-efficient decentralized visual slam," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2466–2473.
- [51] Y. Tian, K. Khosoussi, D. M. Rosen, and J. P. How, "Distributed certifiably correct pose-graph optimization," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 2137–2156, 2021.
- [52] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, Z. Liu, H. I. Christensen, and F. Dellaert, "Multi robot object-based slam," in *2016 International Symposium on Experimental Robotics*. Springer, 2017, pp. 729–741.
- [53] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. I. Christensen, and F. Dellaert, "Distributed trajectory estimation with privacy and communication constraints: A two-stage distributed gauss-seidel approach," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 5261–5268.
- [54] I. Deusch, M. Liu, and R. Siegwart, "A framework for multi-robot pose graph slam," in *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, 2016, pp. 567–572.
- [55] G. Georgakis, B. Bucher, K. Schmeckpeper, S. Singh, and K. Daniilidis, "Learning to map for active semantic goal navigation," *arXiv preprint arXiv:2106.15648*, 2021.
- [56] L. Zheng, C. Zhu, J. Zhang, H. Zhao, H. Huang, M. Niessner, and K. Xu, "Active scene understanding via online semantic reconstruction," in *Computer Graphics Forum*, vol. 38, no. 7. Wiley Online Library, 2019, pp. 103–114.
- [57] J. A. Placed, J. Strader, H. Carrillo, N. Atanasov, V. Indelman, L. Carlone, and J. A. Castellanos, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Transactions on Robotics*, 2023.
- [58] A. Asgharivaskasi and N. Atanasov, "Active bayesian multi-class mapping from range and semantic segmentation observations," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 1–7.
- [59] —, "Semantic octree mapping and shannon mutual information computation for robot exploration," *IEEE Transactions on Robotics*, vol. 39, no. 3, pp. 1910–1928, 2023.
- [60] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "isam2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.
- [61] F. Dellaert and GTSAM Contributors, "borglab/gtsam," May 2022. [Online]. Available: <https://github.com/borglab/gtsam>
- [62] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "Rangenet++: Fast and accurate lidar semantic segmentation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4213–4220.
- [63] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," Jan. 2023. [Online]. Available: <https://github.com/ultralytics/ultralytics>
- [64] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, "Yolo-world: Real-time open-vocabulary object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16901–16911.
- [65] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

## IX. AUTHOR'S BIOGRAPHY SECTION



**Xu Liu** received the M.S.E. degree in Robotics and the Ph.D. degree in Mechanical Engineering and Applied Mechanics from the General Robotics, Automation, Sensing, and Perception (GRASP) Laboratory, University of Pennsylvania, Philadelphia, PA, USA, in 2019 and 2024, respectively. He was a Postdoctoral Scholar at Stanford University, Stanford, CA, USA. He is currently a Senior Applied Scientist at Microsoft, Redmond, WA, USA. His research interests include metric-semantic SLAM, autonomous UAVs, and foundation models for robotics.



**Jiuzhou Lei** received his bachelor's degree in Mechanical Engineering from Sichuan University, Chengdu, China in 2021 and the M.S.E. degree in Robotics from the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA, USA in 2023. He is currently a Ph.D. student in Mechanical Engineering at Texas A&M University, College Station, Texas, USA. He works on robot autonomy, general-purpose robotic manipulation, and navigation.



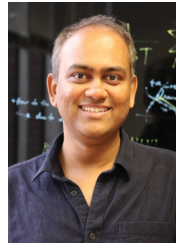
**Ankit Prabhu** received the B.Tech. degree in Computer Science and Engineering from SRM University, Chennai, India, in 2021, and the M.S.E. degree in Robotics from the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA, USA, in 2023. He is currently a Research Scientist at the Kumar-Robotics lab, and his research focuses on metric-semantic SLAM and robot autonomy.



**Yuezhan Tao** received the B.Eng. (Hons.) degree in computer science and engineering from The Chinese University of Hong Kong, Shenzhen, Guangdong, China, in 2019, the M.S.E degree in computer and information science from the University of Pennsylvania, Philadelphia, PA, USA in 2021. He is currently a Ph.D student in computer and information science with the University of Pennsylvania, Philadelphia, PA, USA. His research focuses on active perception, with an emphasis on mapping and motion planning for autonomous robotic systems.



**Igor Spasojevic** is an Assistant Professor at the University of California, Riverside. Before joining UCR, he was a postdoctoral scholar in the GRASP Laboratory at the University of Pennsylvania, Philadelphia, PA. He received his Ph.D. from the Massachusetts Institute of Technology, and his master's and bachelor of arts degrees from the University of Cambridge. His interests focus on optimization algorithms in robotics, including topics such as motion planning, active sensing, and autonomous exploration.



**Pratik Chaudhari** is an Assistant Professor in Electrical and Systems Engineering and Computer and Information Science at the University of Pennsylvania. He is a core member of the GRASP Laboratory. From 2018-19, he was a Senior Applied Scientist at Amazon Web Services and a Postdoctoral Scholar in Computing and Mathematical Sciences at the California Institute of Technology, Pasadena, CA, USA. He received his PhD in Computer Science from University of California, Los Angeles, CA, USA, in 2018, and his Master's and Engineer's degrees in Aeronautics and Astronautics from Massachusetts Institute of Technology, Cambridge, MA, USA, in 2012 and 2014, respectively. He was a part of NuTonomy Inc. (now Hyundai-Aptiv Motionial) from 2014-16. He is the recipient of the Amazon Machine Learning Research Award in 2020, National Science Foundation CAREER award in 2022 and the Intel Rising Star Faculty Award in 2022.



**Nikolay Atanasov** (Senior Member, IEEE) received a B.S. degree in Electrical Engineering from Trinity College, Hartford, CT, USA, in 2008, and M.S. and Ph.D. degrees in Electrical and Systems Engineering from the University of Pennsylvania, Philadelphia, PA, USA in 2012 and 2015, respectively. He is an Associate Professor of Electrical and Computer Engineering with the University of California San Diego, La Jolla, CA, USA. His research focuses on probabilistic models for simultaneous localization and mapping (SLAM) and on optimal control and reinforcement learning algorithms for minimizing model uncertainty. Dr. Atanasov was the recipient of the Joseph and Rosaline Wolf award for the best Ph.D. dissertation in Electrical and Systems Engineering at the University of Pennsylvania in 2015, the Best Conference Paper Award at the IEEE International Conference on Robotics and Automation (ICRA) in 2017, the NSF CAREER Award in 2021, and the IEEE RAS Early Academic Career Award in Robotics and Automation in 2023.



**Vijay Kumar** (Fellow, IEEE) received the Ph.D. degree in mechanical engineering from the Ohio State University, Columbus, OH, USA, in 1987. He is currently the Nemirovsky Family Dean of Penn Engineering (with appointments) with the Department of Mechanical Engineering and Applied Mechanics, the Department of Computer and Information Science, and the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA. From 2012 to 2014, he was the Assistant Director of Robotics and Cyber Physical Systems with the White House Office of Science and Technology Policy. has served on the editorial boards of the IEEE Transactions on Robotics and Automation, IEEE Transactions on Automation Science and Engineering, ASME Journal of Mechanical Design, and the Springer Tract in Advanced Robotics (STAR), and was the chief editor for the ASME Journal of Mechanisms and Robotics. He has won best paper awards at DARS 2002, ICRA 2004, ICRA 2011, RSS 2011, RSS 2013, ICRA 2014, BICT 2015, and MARSS 2016 and has advised doctoral students who have won Best Student Paper Awards at ICRA 2008, RSS 2009, and DARS 2010. He was recognized with the RSS Time of Test Award in 2025. Additionally, he is the recipient of the 2013 Popular Mechanics Breakthrough Award, a 2014 Engelberger Robotics Award, the 2017 IEEE Robotics and Automation Society George Saridis Leadership Award, the 2018 IEEE Robotics and Automation Pioneer Award, and the 2020 IEEE Robotics and Automation Field Award. He was elected to the National Academy of Engineering in 2013, the American Philosophical Society in 2018, and the American Academy of Arts and Sciences and the National Academy of Inventors in 2022.