Hydra++: Real-Time Hierarchical 3D Scene Graph Construction With Object-Level Shape Estimation

Hyungtae Lim[1], Nathan Hughes[1], Xihang Yu[1], Ruihan Xu[1],
Yun Chang[1], Jingnan Shi[1], Rajat Talak[2], and Luca Carlone[1]

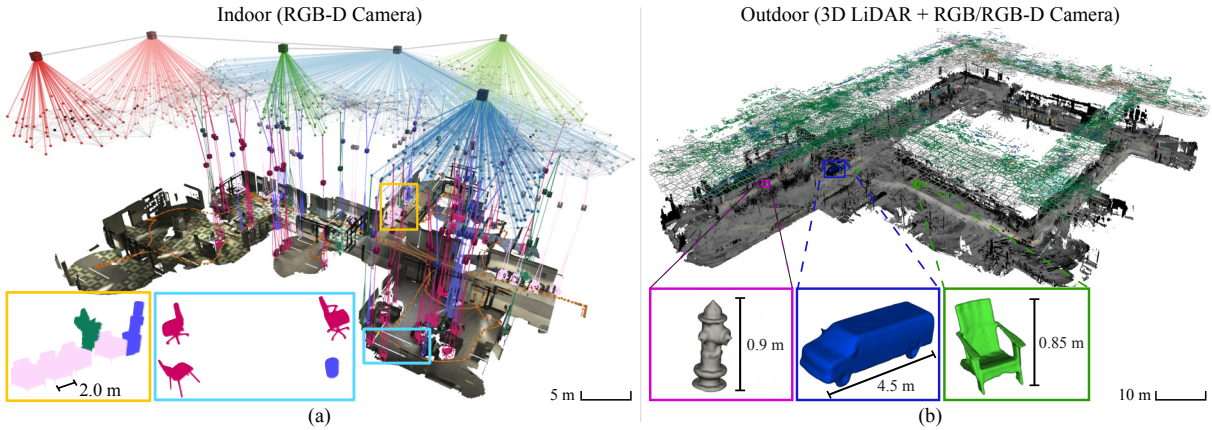


Fig. 1. 3D scene graphs with object-level shape reconstruction generated by *Hydra++*. (a) Indoor environment using the uHumans2 dataset [1]. Zoom-in views show full object shapes reconstructed from partial observations. Each color represents a semantic class (*e.g.,* couches in pink, plants in green, trash bins in purple, and chairs in red). The object shape estimation network in our framework captures intra-class variation, as illustrated by the reconstructed trash bins and chairs with different geometries. (b) Outdoor campus scene using the hybrid LiDAR-camera configuration. Our method reconstructs objects of various sizes, from a small fire hydrant to a large car, under realistic sensing conditions.

***Abstract*— 3D scene graphs provide a hierarchical abstraction of environments by encoding spatial entities (*e.g.,* objects, places) and their relationships. However, existing scene graph systems model object geometry coarsely, relying on partial point clouds or class-level CAD templates, which limits instance-specific shape detail. This paper presents *Hydra++*, a system-level investigation into how learning-based object shape estimators can be integrated into a hierarchical 3D scene graph pipeline. Hydra++ incorporates category-agnostic shape estimation and a reprojection-mask consistency check (RMCC) to reject degenerate predictions from partial observations or imprecise segmentation. In its default CRISP-based configuration, Hydra++ performs online scene graph construction; slower estimators such as SAM3D are evaluated as modular alternatives to demonstrate generalization-latency trade-offs. Furthermore, to address the challenges of sparse and noisy depth measurements in outdoor environments, Hydra++ supports a hybrid LiDAR-camera configuration for large-scale operation, improving scene-level reconstruction quality. Experiments in both simulation and real-world outdoor campus scenarios demonstrate that Hydra++ improves object- and scene-level reconstruction quality. Project page is available at `https://hydra-plusplus.github.io/`.**

[1]H. Lim, N. Hughes, X. Yu, R. Xu, Y. Chang, J. Shi, and L. Carlone are with the Laboratory for Information & Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, USA, {shapelim, na26933, jimmyyu, multyxu, yunchang, jnshi, lcarlone}@mit.edu

[2]R. Talak is with the Department of Electrical and Computer Engineering at the National University of Singapore, Singapore, {talak@nus.edu.sg}

This work was partially funded by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. RS-2024-00461409) and by the Department of the Air Force Artificial Intelligence Accelerator, accomplished under Cooperative Agreement Number FA8750-19-2-1000.

I. Introduction

Modeling 3D scenes is crucial for many robotics tasks, from navigation to manipulation [2]. With the rise of deep learning and large language models (LLMs), many studies have examined semantic mapping, which captures semantic information about the environment to support reasoning and interaction [3]–[6]. In particular, recent 3D scene graph approaches encode semantics by treating entities (*e.g.,* objects, rooms, and agents) as nodes, and relationships as edges (*e.g.,* adjacency, inclusion, and support), yielding a lightweight and efficient abstraction of the environment [7], [8].

However, most scene graph systems still treat object geometry as a coarse proxy, typically relying on centroids and bounding boxes with partial reconstructions from accumulated point clouds [2], [9]. This abstraction is often sufficient for semantic queries [10], but it becomes a bottleneck when downstream tasks require instance-specific geometry, such as contact-aware reasoning, model-based manipulation, or rearrangement. In outdoor environments, this limitation compounds with sparse and noisy depth measurements, making mesh reconstruction of both objects and background especially challenging. Together, these challenges motivate the need for higher-fidelity object and background reconstructions, especially in large-scale outdoor environments.

In this paper, we present *Hydra++*, a system-level study of how object shape estimators can be integrated into a hierarchical 3D scene graph pipeline. As shown in Fig. 1, Hydra++ extends the frameworks of Hydra [8] and Khronos [11]

with per-object mesh-level reconstruction by incorporating category-agnostic shape estimators (CRISP [12] or SAM3D [13]). To handle degenerate shape predictions arising from partial observations or imprecise segmentation, we introduce a reprojection-mask consistency check (RMCC). We further study a hybrid LiDAR-camera configuration that combines the metric-scale depth from a 3D LiDAR sensor with image-based shape estimation, improving reconstruction quality at both the object level and the scene level in outdoor settings. Throughout the paper, the real-time claim refers to the default CRISP-based instantiation of Hydra++. SAM3D is included as an alternative module to analyze the trade-off between out-of-domain generalization and inference latency.

With Hydra++, we claim the following contributions. We (i) introduce an enhanced 3D scene graph construction system that integrates learning-based object shape estimation models to provide high-fidelity object geometry while remaining robust to noisy object detections, (ii) present a study of trade-offs between state-of-the-art shape-estimation methods, and (iii) extend the system with a hybrid LiDAR-camera configuration to support large-scale outdoor environments. We support these claims with experiments in both simulated and real-world environments.

II. Related Work

Object-level SLAM. 3D scene understanding is a fundamental task in robotic mapping, with extensive research enabling robots to understand and represent environments at varying semantic granularities. Pioneered by SLAM++ [3], object-level SLAM has since incorporated semantic objects directly into the mapping and localization pipeline [4]–[6]. Representative systems such as QuadricSLAM [4] and CubeSLAM [5] model objects using parametric geometric primitives within factor graph formulations, enabling lightweight and interpretable object-level mapping. While effective, these approaches generally rely on predefined geometric models, which limits their ability to adapt to partially observed or irregular object geometries.

To address this limitation, subsequent works extend this line of research through robust object data association and optimization. Bowman *et al.* [14] tackle the data association problem directly by formulating a joint optimization over metric and semantic information with probabilistic object associations, enabling reliable object-level mapping in ambiguous environments. Building on this line of work, Atanasov *et al.* [15] provide a unifying view of geometry, semantics, and data association in SLAM, introducing structured object models of mid-level part features to generalize beyond purely geometric representations.

Learning-Based Object Shape Representations. Further advancing the expressiveness of object representations, Shan *et al.* [16] propose *ELLIPSDF*, which jointly optimizes object pose and shape using a bi-level model consisting of a coarse ellipsoid and a fine-grained neural SDF, enabling compact yet expressive object-level map inference from multi-view RGB-D observations. Shape estimation for objects has since been explored through a variety of paradigms, including parametric fitting [17], volumetric representations [18], learning-based implicit functions [12], [19], and diffusion-based approaches [20], collectively demonstrating the feasibility of generating detailed object geometries even from partial observations.

In parallel, the integration of shape priors for detailed object representations has been explored [21], while decentralized and collaborative formulations have been proposed to support multi-robot object-level mapping under communication constraints, exemplified by SlideSLAM [22].

Scene Representations With 3D Scene Graphs. These developments in object-level representation collectively point toward a growing interest in organizing rich object geometry and semantics within structured spatial frameworks. Scene graph-based methodologies have emerged as a promising paradigm for such hierarchical spatial understanding [1], [7], [23], enabling robots to reason about environments from low-level geometric primitives to high-level semantic concepts such as rooms and buildings. Hughes *et al.* [8] present Hydra, a real-time spatial perception system that incrementally constructs layered scene graphs, handling loop closures through hierarchical descriptors and deformation-based optimization. Bavle *et al.* [24], [25] also develop LiDAR-based hierarchical representations for indoor scenes, called *S-Graphs* and *S-Graphs+*. Werby *et al.* [26] extend this paradigm by incorporating open-vocabulary vision-language features, enabling language-conditioned queries in multi-story environments. Greve *et al.* [27] further push scene graphs toward outdoor autonomy, constructing collaborative dynamic scene graphs from multi-agent panoptic LiDAR data for large-scale urban environments.

Although object-level SLAM and scene graph representations are each advancing rapidly, there remains limited exploration of how detailed learned object shapes can be organized within hierarchical scene understanding frameworks. Existing scene graph systems excel at capturing semantic and topological relationships but typically abstract object geometry to simple primitives, while shape estimation methods produce detailed object models but often operate in isolation from broader spatial context. In this work, we investigate the integration of learned object mesh estimation within scene graph representations using CRISP [12] or SAM3D [13], and explore how such representations can be incorporated into hierarchical spatial understanding pipelines to enable applications that benefit from both semantic awareness and object-level geometric detail.

III. Methodology

A. System Overview

Hydra++ builds on the Hydra [8] and Khronos [11] frameworks for 3D scene graph construction, extending them with a learning-based shape estimation module [12], [13], a geometric consistency check for predicted object shapes, and support for a hybrid LiDAR-camera sensor configuration.

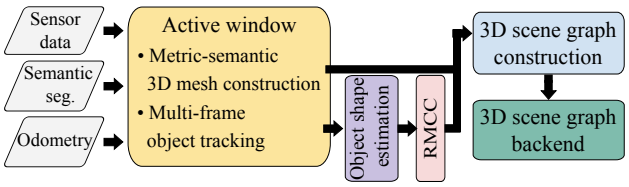


Fig. 2. System pipeline of *Hydra++*. Sensor data (RGB-D or 3D LiDAR + camera, semantic segmentation, and odometry) flows through an active window module that performs metric-semantic 3D mesh construction and object tracking. Inactive tracks are processed by a learning-based shape estimation module (instantiated as CRISP [12] or SAM3D [13]) for object pose and shape estimation, which is followed by our proposed reprojection-mask consistency check (RMCC) to validate the predictions. The 3D scene graph construction module instantiates nodes across multiple semantic layers, which are then optimized and updated asynchronously by the backend through deformation-graph-based optimization and incremental updates [1].

As illustrated in Fig. 2, the system receives sensor data (RGB-D or LiDAR + RGB/RGB-D camera), instance segmentation masks, and odometry estimates. From these inputs, an *active window* module maintains a volumetric map $\mathcal{M}_t$ that is incrementally updated via projective truncated signed distance field (TSDF) integration [28]. Meanwhile, object candidates are extracted in image space from the instance label image $\mathbf{I}_{\text{inst}}$, which encodes per-pixel category and instance IDs from an off-the-shelf segmentation model. Pixels sharing the same instance ID are grouped into clusters $\mathcal{C}_i^t$, each lifted to 3D by back-projecting the depth image through the camera intrinsics and depth information into a vertex map $\mathbf{V}$. Clusters are filtered by sensing range, pixel count, and 3D bounding-box volume. A max-IoU tracker [11] then associates clusters across frames using voxel-based overlap, forming temporal object tracks $\mathcal{T}_k$, where k indexes the currently active tracks.

When a track becomes inactive (*i.e.,* the object moves outside the active window), the system invokes a learning-based shape estimation module for 6-DoF pose and shape estimation (Sec. III-C), followed by a reprojection-mask consistency check (RMCC) to validate the prediction (Sec. III-D). Validated object nodes, together with place and region nodes, are assembled into a hierarchical 3D scene graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ using the Hydra backend [8], where nodes can span multiple semantic layers (*e.g.,* agents, objects, places, and regions) and edges encode spatial and semantic relationships.

In the remainder of this section, we describe the two primary contributions of Hydra++: (a) an adaptive integration strategy for improved ground reconstruction in the hybrid LiDAR-camera configuration (Sec. III-B), and (b) the object-level pipeline comprising a learning-based shape estimation module (Sec. III-C) and a geometric consistency check via RMCC (Sec. III-D).

B. Ground-Aware Adaptive Integration Strategy

In TSDF-based mapping, each voxel $\boldsymbol{u}$ within the active window stores a signed distance value $\phi(\boldsymbol{u})$: positive in free space (*i.e.,* in front of the surface), negative behind a surface, and zero at the surface itself. Surface meshes are extracted by finding the zero level-set $\{\boldsymbol{u} \mid \phi(\boldsymbol{u}) = 0\}$, which in practice reduces to detecting sign changes between neighboring voxels via marching cubes. For a sign change

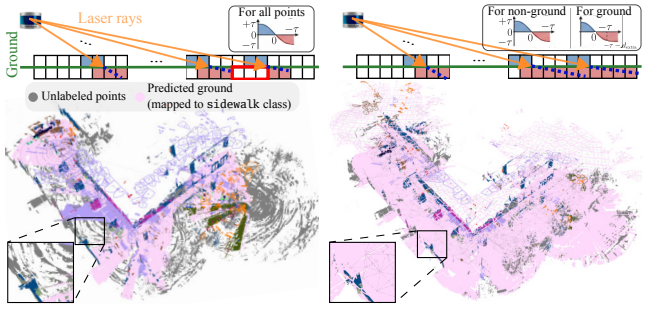


Fig. 3. Comparison of TSDF integration behavior and resulting ground mesh quality. Top: LiDAR laser rays hit the ground at high incidence angles, especially at long range. (a) Since standard integration drops measurements with signed distances beyond the truncation boundary $-\tau$, at long range the oblique ray overestimates the signed distance and the negative-side voxels receive no updates (white voxels in the red box), breaking the sign transitions. (b) Our ground-aware adaptive integration strategy allows measurements within an additional range d_{extra} beyond $-\tau$ for ground-labeled voxels while keeping the standard band for non-ground voxels, ensuring that negative-side voxels are updated even at large incidence angles. Bottom: the resulting mesh quality, where (a) standard integration produces fragmented ground geometry, while (b) our method recovers a complete and consistent ground mesh.

to exist, voxels on *both sides* of a surface must receive integration updates; however, voxels beyond the truncation distance τ behind a surface are never updated and carry no weight. This means that if the negative-side voxels of a surface are not reached by any measurement, no sign change is recorded and the surface simply goes missing (*i.e.,* red box in Fig. 3(a)).

In projective integration, the signed distance is estimated along the ray rather than the surface normal [11], which overestimates the true distance at high incidence angles and pushes measurements outside the truncation band, leaving negative-side voxels without updates. When using LiDAR point clouds, this effect is compounded by the sparsity of LiDAR returns on the ground plane at long range, causing inconsistent sign transitions and fragmented or missing ground meshes.

To address this, we introduce a ground-aware adaptive integration strategy. For ground-labeled voxels [29], instead of discarding measurements whose signed distance falls below the truncation distance $-\tau$, we allow measurements whose signed distance lies within an additional range d_{extra} beyond the truncation band (empirically, $d_{\text{extra}} = 3\,\text{m}$), as illustrated by the extended dotted lines in Fig. 3(b). Measurements falling within this extended band are integrated by clamping their observed distance to $-\tau$ with a small constant weight, rather than being discarded. This controlled negative support ensures that voxels behind the ground surface receive integration updates, restoring the sign transitions required for reliable zero level-set extraction.

C. Object Pose and Shape Estimation

As mentioned in Sec. III-A, when an object track $\mathcal{T}_k$ becomes inactive, the system invokes a learning-based pose and shape estimation module to predict the 6-DoF object pose and geometry from the observation in which the object

is most fully visible. To identify this observation, we heuristically select the frame with the maximum 3D bounding-box volume, which typically corresponds to the view with minimal occlusion:

$$(\mathcal{C}^*, t^*) = \underset{(\mathcal{C}^{t_r}, t_r) \in \mathcal{T}_k}{\arg\max} \ \mathrm{vol}\big(\mathrm{3DBBox}(\mathcal{C}^{t_r})\big)\,. \tag{1}$$

For the selected observation, we extract the RGB-D frame and binary mask $\mathbf{M}_{\text{obs}} \in \{0,1\}^{H\times W}$ to form the module input:

$$\mathbf{x} = (\mathbf{I}_{\text{RGB}},\, \mathbf{I}_{\text{depth}},\, \mathbf{M}_{\text{obs}},\, \mathbf{K})\,, \tag{2}$$

where $\mathbf{I}_{\text{RGB}} \in \mathbb{R}^{H\times W\times 3}$ is the RGB image, $\mathbf{I}_{\text{depth}} \in \mathbb{R}^{H\times W}$ is the depth image obtained from either an RGB-D camera or a LiDAR point cloud projected onto the image plane, and $\mathbf{K} \in \mathbb{R}^{3\times 3}$ is the camera intrinsic matrix. The module outputs an object pose ${}^{C}\mathbf{T}_O \in \mathrm{SE}(3)$, a metric scale $s \in \mathbb{R}_+$, and an object mesh $\mathcal{M}$. The object pose in the world frame ${}^{W}\mathbf{T}_O$ is:

$${}^{W}\mathbf{T}_O = {}^{W}\mathbf{T}_{C_{t^*}} \cdot {}^{C}\mathbf{T}_O, \tag{3}$$

where ${}^{W}\mathbf{T}_{C_{t^*}}$ is the camera pose at time t^* in (1). Each object node stores:

$$\mathcal{A}_{\text{obj}} = \{l,\ s,\ \mathcal{M},\ {}^{W}\mathbf{p}_O,\ {}^{W}\mathbf{R}_O\}, \tag{4}$$

where l is the semantic label, $s \in \mathbb{R}_+$ is the metric scale, $\mathcal{M}$ is the object mesh, and ${}^{W}\mathbf{p}_O \in \mathbb{R}^3$ and ${}^{W}\mathbf{R}_O \in \mathrm{SO}(3)$ are the translation and rotation of ${}^{W}\mathbf{T}_O$, respectively.

Next, we describe the integration with two different pose and shape estimators. Note that our framework is modular and agnostic to the underlying shape estimation architecture, making it readily extensible to other methods.

CRISP: CRISP [12] is a category-agnostic network that jointly estimates object mesh shape and 6-DoF pose and metric scale via normalized object coordinate space (NOCS) prediction. It employs a DINOv2 backbone [30] with parallel branches for shape reconstruction and NOCS prediction, respectively, yielding:

$${}^{C}\mathbf{T}_O \in \mathrm{SE}(3), \quad s \in \mathbb{R}_+, \quad \mathbf{z}_{\text{shape}} \in \mathbb{R}^{256}, \tag{5}$$

where ${}^{C}\mathbf{T}_O$ is the object pose relative to the NOCS canonical frame, s is the metric scale, and $\mathbf{z}_{\text{shape}}$ is the latent shape code. Because CRISP solves for rotation, translation, and scale jointly through pixel-wise correspondences between 3D points back-projected from $\mathbf{I}_{\text{depth}}$ and their predicted NOCS coordinates, all three quantities are recovered in a single forward pass without additional processing. The object mesh $\mathcal{M}$ is then obtained by decoding $\mathbf{z}_{\text{shape}}$ through an implicit SDF decoder, extracting the iso-surface via marching cubes and scaling by s.

SAM3D: SAM3D [13] outputs $\mathcal{M}$ directly as a dense mesh along with an object pose estimate, but the mesh is defined only up to an unknown scale factor. To recover metric scale, following the pixel-wise scale alignment used in VGGT-SLAM [31], we establish pixel-wise correspondences between the depth rendered by rasterizing $\mathcal{M}$ from the camera viewpoint and the sensor-measured depth $\mathbf{I}_{\text{depth}}$ over the object mask $\mathbf{M}_{\text{obs}}$, and estimate s as the median depth ratio across all valid correspondences.

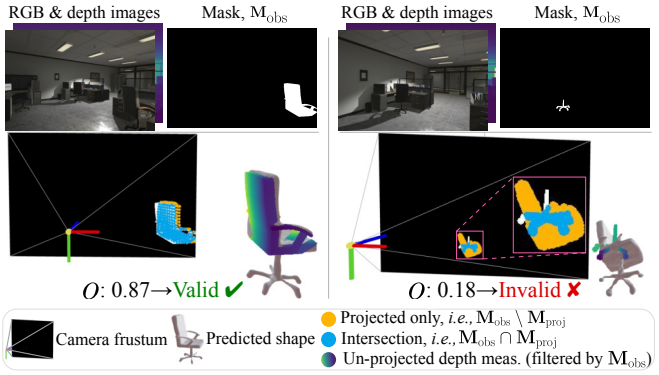


Fig. 4. Overview of the reprojection-mask consistency check (RMCC) for valid (left) and invalid (right) cases. From top to bottom: the RGB-D image with the observed object mask $\mathbf{M}_{\text{obs}}$ (input to the shape estimation module, Sec. III-C); and the camera frustum visualizing the observed mask $\mathbf{M}_{\text{obs}}$ alongside the reprojected mask $\mathbf{M}_{\text{proj}}$ generated from a point cloud sampled from the predicted shape. The validity of the predicted object shape is determined by the *overlap score* O between $\mathbf{M}_{\text{proj}}$ and $\mathbf{M}_{\text{obs}}$; see (8).

D. Reprojection-Mask Consistency Check (RMCC)

The shape estimation module can produce unreliable predictions from noisy or incomplete masks due to over-segmentation or mask imprecision. To filter out such degenerate predictions, we employ a reprojection-mask consistency check (RMCC) that verifies geometric alignment between the predicted shape and the observed mask (Fig. 4). RMCC samples surface points $\mathcal{P}_{\text{surf}}$ from the predicted mesh $\mathcal{M}$, scales them by s, transforms them via ${}^{C}\mathbf{T}_O$, and projects them onto the image plane:

$$\mathbf{u}_j = \pi\left(\mathbf{p}_{\text{cam},j}; \mathbf{K}\right), \tag{6}$$

$$\text{where } \pi([x,y,z]^\top; \mathbf{K}) = \left[\frac{f_x x}{z} + c_x,\ \frac{f_y y}{z} + c_y\right]^\top. \tag{7}$$

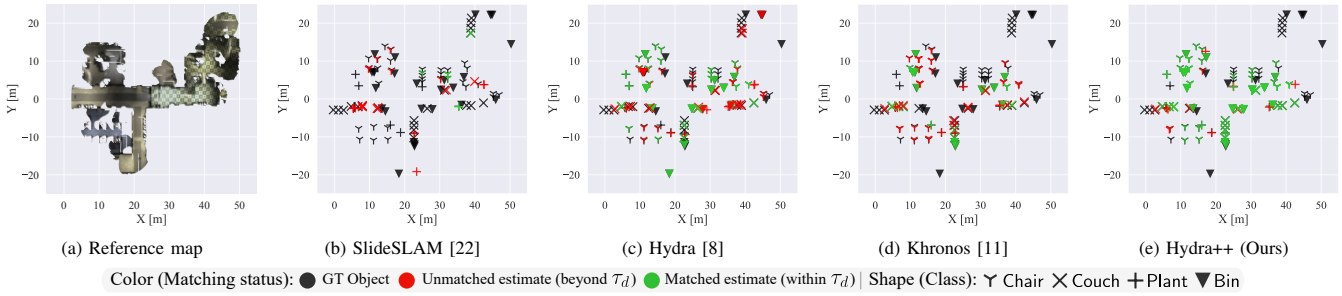
The projected coordinates form a mask $\mathbf{M}_{\text{proj}} \in \{0,1\}^{H\times W}$, and we compute the overlap score:

$$O = \frac{|\mathbf{M}_{\text{proj}} \cap \mathbf{M}_{\text{obs}}|}{|\mathbf{M}_{\text{proj}}|}. \tag{8}$$

A prediction is accepted if $O > \tau_O$, filtering pose ambiguities from occluded or truncated viewpoints.

IV. Experiments

We conduct experiments to substantiate the three key contributions. We (i) evaluate the integration of learning-based object shape estimation into a hierarchical 3D scene graph pipeline (Secs. IV-C and IV-D); (ii) analyze the practical trade-offs between shape estimator choices by comparing CRISP, our default estimator, with SAM3D as a slower alternative instantiation (Sec. IV-E), and study the effect of RMCC on prediction reliability (Sec. IV-G) and (iii) evaluate the hybrid LiDAR-camera configuration in outdoor environments, assessing improvements in both scene-level mesh continuity and object-level geometric reconstruction (Sec. IV-F).



(a) Reference map (b) SlideSLAM [22] (c) Hydra [8] (d) Khronos [11] (e) Hydra++ (Ours)

Fig. 5. (a) Reference point cloud map and qualitative comparison of object-level mapping results for (b) SlideSLAM [22], (c) Hydra [8], (d) Khronos [11], and (e) our Hydra++. An estimated object is marked as matched (green) if the closest ground-truth (GT) object of the same class lies within $\tau_d = 0.2\,\text{m}$, and unmatched (red) otherwise, *i.e.,* the centroid of the partially reconstructed object deviates substantially from the GT centroid.

A. Environment Setups

First, to compare our approach with state-of-the-art scene graph approaches [8], [11], we utilize the uHumans2 dataset [1], a photorealistic simulation environment that provides ground-truth object meshes with semantic annotations. We select Chair, Couch, Plant, and Bin as object categories of interest.

Importantly, in closed-set semantic segmentation, a finite label taxonomy forces many distinct real-world object instances to be mapped to the same category label, inevitably introducing substantial intra-class variation [32]. For example, in the office scene in uHumans2, the Chair class includes (i) office chairs with armrests and wheels, (ii) office chairs with wheels, and (iii) table chairs, while the Bin class includes (i) small bins, (ii) large bins, and (iii) water dispensers. Consequently, this office scene is well-suited to highlight the benefits of our category-agnostic shape prediction, which can accommodate large intra-class diversity without relying on category-specific shape priors.

For real-world deployment, we used a sequence from the Kimera-Multi dataset [33]. The dataset features a robot equipped with a RealSense D455 RGB-D camera and a Velodyne 3D LiDAR, enabling hybrid sensor configurations. Training CRISP on this real-world data requires two inputs for each instance: (a) a segmentation mask and (b) a ground-truth object mesh. For masks, we applied SAM [34] to automatically segment object instances from the RGB frames, with the goal of assessing whether CRISP can be trained using only off-the-shelf tools without manual annotation, thereby probing its feasibility for in-the-wild deployment. In total, approximately 50 keyframes were used across three object categories: fire hydrant, car, and bench.

For ground-truth geometry, we scanned target objects using an iPad Pro with a built-in LiDAR sensor and a 3D scanning application. The resulting meshes contained holes from limited scanning coverage and thin structures (*e.g.,* narrow object parts), which we manually repaired using Blender [35] to produce watertight meshes. This hole-filling step is crucial: CRISP relies on a well-defined signed distance field, and holes introduce sign ambiguities near open boundaries that can corrupt the learned shape codes.

TABLE I. Performance comparison of state-of-the-art approaches for a given distance threshold τ_d in the uHumans2 dataset [1]. Precision (P), recall (R), and F_1 are computed using the centroids of the ground-truth and estimated object bounding boxes. Note that Chamfer distance (Chamf.), B-IoU, and V-IoU are averaged only over valid object pairs, where the distance between the estimated and ground-truth centroids is below τ_d.

	Method	P↑	R↑	F_1↑	Chamf.↓	B-IoU↑	V-IoU↑
$\tau_d = 0.2\,\text{m}$	SlideSLAM	0.206	0.057	0.089	0.039	0.606	0.199
	Hydra	0.529	0.409	0.461	0.035	0.515	0.298
	Khronos	0.362	0.239	0.288	0.028	0.671	0.350
	Ours w/o RMCC	0.670	**0.602**	0.634	0.008	0.690	0.540
	Ours	**0.787**	0.591	**0.675**	**0.004**	**0.739**	**0.598**
$\tau_d = 0.5\,\text{m}$	SlideSLAM	0.794	0.159	0.265	0.051	0.436	0.134
	Hydra	0.897	**0.705**	**0.789**	0.072	0.412	0.246
	Khronos	**0.971**	0.546	0.699	0.091	0.358	0.239
	Ours w/o RMCC	0.839	0.682	0.752	0.032	0.582	0.451
	Ours	0.921	0.671	0.776	**0.020**	**0.655**	**0.528**

B. Evaluation Metrics

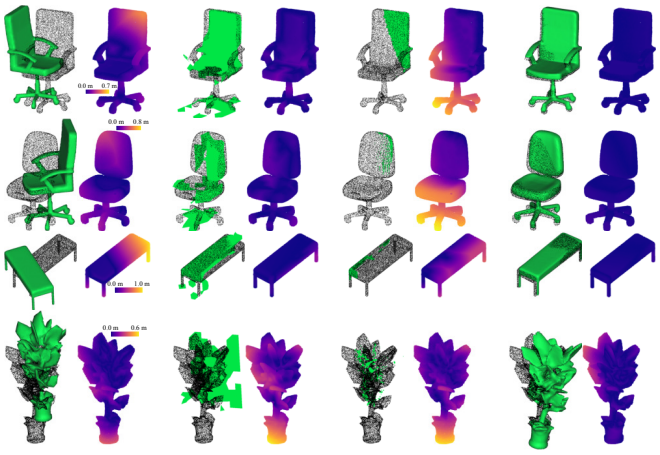
We evaluate object-level mapping in two stages: (i) detection performance and (ii) geometric reconstruction quality. For detection, we use centroid-based metrics. Given a distance threshold τ_d, an estimated object is counted as a true positive if there exists a ground-truth object of the same semantic class whose centroid lies within τ_d of the estimated centroid. Thus, unmatched estimates are false positives, and unmatched ground-truth objects are false negatives. We report precision (P), recall (R), and F_1 score.

For geometric reconstruction, we evaluate each matched object pair (centroid distance $< \tau_d$) using the following metrics:

- *Chamfer Distance*: Bidirectional point-to-point distance between point sets $\mathcal{P}_{\text{pred}}$ and $\mathcal{P}_{\text{GT}}$, sampled from predicted and ground-truth meshes, respectively.
- *Bounding-Box IoU (B-IoU)*: Intersection-over-union of the 3D axis-aligned bounding boxes of the matched objects:

$$\text{B-IoU}(\mathcal{C}_{\text{pred}}, \mathcal{C}_{\text{GT}}) = \frac{\text{vol}(\text{3DBBox}(\mathcal{C}_{\text{pred}}) \cap \text{3DBBox}(\mathcal{C}_{\text{GT}}))}{\text{vol}(\text{3DBBox}(\mathcal{C}_{\text{pred}}) \cup \text{3DBBox}(\mathcal{C}_{\text{GT}}))}, \tag{9}$$

  where $\mathcal{C}_{\text{pred}}$ and $\mathcal{C}_{\text{GT}}$ denote the predicted and ground-truth object clusters, respectively.
- *Volumetric IoU (V-IoU)*: Intersection-over-union between voxelized occupancies of the matched meshes, following the same form as (9) with $\mathcal{U}_{\text{pred}}$ and $\mathcal{U}_{\text{GT}}$ denoting the sets of occupied voxels for the predicted and ground-truth meshes, respectively.



(a) SlideSLAM [22] (b) Hydra [8] (c) Khronos [11] (d) Hydra++ (Ours)

Fig. 6. Qualitative comparison with other object representations. In each row, the left image shows the estimated mesh (green) overlaid with ground-truth surface points (black), while the right image shows the corresponding per-point error heatmap.

C. Object-Level Scene Graph Quality

First, we compare Hydra++ against existing scene graph SLAM approaches: Hydra [8], which clusters semantic meshes to extract objects, and Khronos [11], which reconstructs objects with instance segmentation, on the uHumans2 dataset.

As shown in Fig. 5 and Table I, Hydra++ achieves competitive performance across detection and shape metrics at the strict threshold ($\tau_d = 0.2\,\text{m}$). In particular, Fig. 5 shows that although the baselines [8], [11] successfully detect objects, they represent them solely by accumulating partial observations, which often results in centroid errors exceeding 0.2 m compared to the ground truth (the partial meshes can be seen in Fig. 6).

At the relaxed threshold ($\tau_d = 0.5\,\text{m}$), Hydra achieves the best F_1 score by achieving the highest recall, yet exhibits substantially higher Chamfer distance and lower B-IoU and V-IoU. This reflects a fundamental limitation of prior approaches: because they rely on partial observations for clustering [8] and instance tracking [11], they cannot infer geometry in unobserved regions, which ultimately degrades geometric accuracy in the scene graph. By contrast, Hydra++ intrinsically performs object shape completion, resulting in lower Chamfer distances and higher volumetric overlap.

D. Analysis on Intra-Class Shape Variation

Next, we emphasize the importance of category-agnostic, learning-based shape estimation in handling intra-class variation. CAD-template-based approaches such as SlideSLAM [22] rely on a single canonical shape per semantic class, which inherently limits their ability to represent diverse real-world instances. As shown in the first and second rows of Fig. 6, visually distinct objects, such as office chairs with and without armrests, are forced to share the same template, leading to systematic misalignment and increased Chamfer distance when the true geometry deviates from the canonical model. In contrast, Hydra++ leverages a learning-based shape estimator to reconstruct instance-specific geometry, capturing salient details such as armrests, seat curvature, and backrest shape.

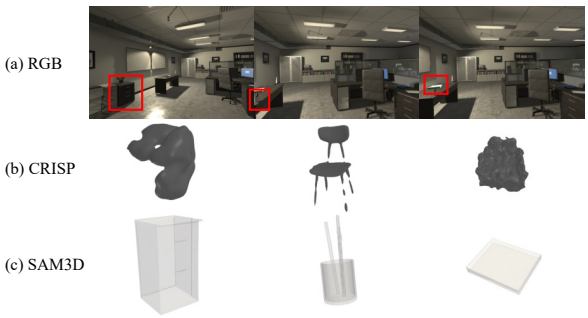


Fig. 7. Qualitative comparison of mesh reconstruction in an out-of-domain setting, where unseen object instances are given as inputs. (a) RGB image for reference, with target objects highlighted in red boxes; (b) mesh results produced by CRISP [12] and (c) those produced by SAM3D [13].

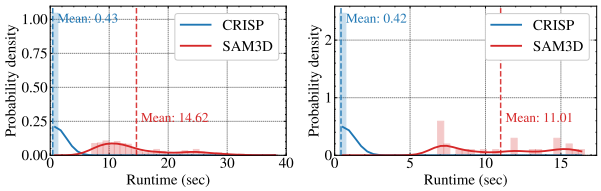


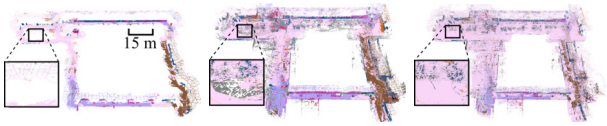
Fig. 8. Left to right: Timing comparison between SAM3D [13] and CRISP [12] in indoor (using the uHumans2 dataset [1]) and outdoor (using the Kimera-Multi dataset [33]) environments.

Overall, these results demonstrate that fixed per-class CAD templates are insufficient in diverse environments, whereas Hydra++ effectively captures intra-class variation through instance-level shape estimation.

E. Shape Estimation Model Comparison

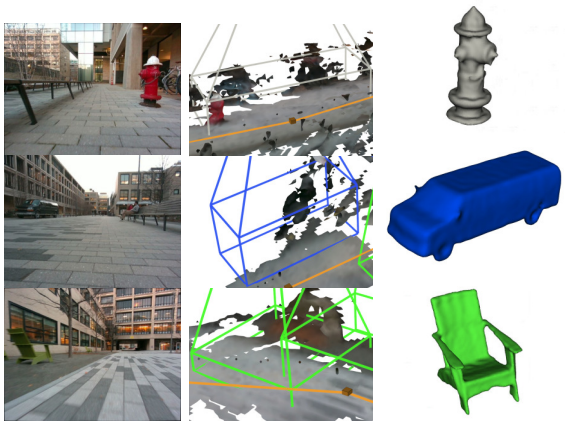
Although our pipeline is primarily built upon CRISP [12], we further analyze how the choice of shape estimation model influences overall system behavior by comparing CRISP and SAM3D [13]. This comparison does not imply that every supported estimator enables real-time operation: CRISP is the default estimator used for the real-time configuration, whereas SAM3D is evaluated to demonstrate the modularity of the interface and the computational cost associated with stronger out-of-domain generalization. The key distinction lies in the trade-off between out-of-domain generalization and inference speed. To test out-of-domain generalization, we introduce an additional computer class alongside our original four categories. In the uHumans2 dataset, various distinct office supplies are ambiguously grouped into this single computer label.

As a foundation model, SAM3D exhibits strong generalization capabilities, reliably reconstructing novel, out-of-domain geometries where CRISP struggles (see Figs. 7(b) and 7(c)). However, this robustness comes at the expense of significantly higher inference latency, as presented in Fig. 8; in our experiments, SAM3D requires roughly 11–14 s per object and is therefore unsuitable for real-time multi-object deployment in the current system. Conversely, while CRISP is not a foundation model and is limited to its training distribution, it boasts much faster inference times.



(a) (b) (c)

Fig. 9. Mesh quality comparison in outdoor scenes: (a) Meshes generated in RGB-D-only mode, corresponding to the original metric-semantic mesh generation in Hydra [8] or Khronos [11]. (b)–(c) Meshes from a hybrid LiDAR-camera mode without and with the adaptive integration strategy, respectively.



(a) RGB input (b) Khronos [11] (c) Hydra++ (Ours)

Fig. 10. Object-level close-up results from the outdoor scene graph. From top to bottom: fire hydrant, car, and bench. (a) RGB input image (*i.e.,* $\mathbf{I}_{\text{RGB}}$ in (2)). (b) Khronos [11], an RGB-D-based method relying on instance-aware voxel clustering, fails to generate reliable object meshes due to sparse depth measurements in outdoor scenes, leading to under-segmentation with partial meshes. (c) Hydra++ with hybrid LiDAR-camera fusion successfully reconstructs fine-grained geometric details of each object via CRISP [12].

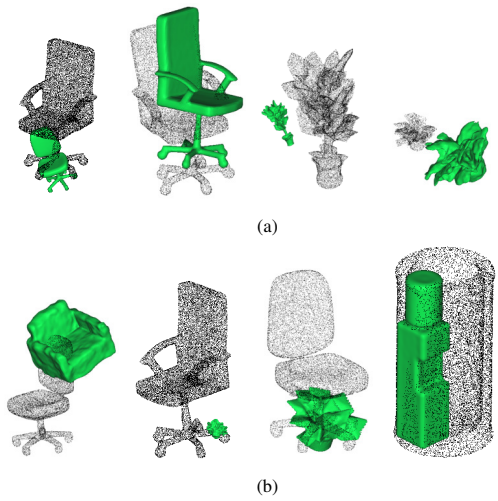
In summary, CRISP is optimized for real-time deployments over known object classes, whereas SAM3D can improve reconstruction of novel objects when the application can tolerate delayed object-level mesh updates.

F. Outdoor Deployment With Hybrid Sensor Fusion Mode

Lastly, we demonstrate that our hybrid LiDAR-camera mode improves both scene-level geometry and object-level reconstruction quality. To validate this capability beyond indoor RGB-D settings, we deploy Hydra++ in outdoor environments using a hybrid sensor configuration, where a 3D LiDAR point cloud provides metric scene structure while an RGB/RGB-D camera enables CRISP-based object shape estimation.

As shown in Fig. 9 and Fig. 10, existing methods, such as Hydra [8] or Khronos [11], suffer from degraded depth accuracy in outdoor scenes, leading to incomplete scene meshes (Fig. 9(a)) and under-segmented object reconstructions (Fig. 10(b)). In contrast, our hybrid mode, using the ground-aware adaptive integration strategy presented in Sec. III-B, reduces mesh discontinuities and spurious holes in the scene reconstruction (Fig. 9(c)), while simultaneously recovering fine-grained object details via learned shape priors (Fig. 10(c)).

This complementary integration enables consistent metric scene meshes together with detailed, instance-level object geometry, supporting our third claim on improved outdoor reconstruction quality.



(a)

(b)

Fig. 11. Examples of erroneous shape estimates rejected by RMCC. Green meshes show estimated shapes, and black points are sampled from ground truth. (a) Intra-class errors with insufficient geometric overlap. (b) Inter-class errors from class-agnostic CRISP predictions.

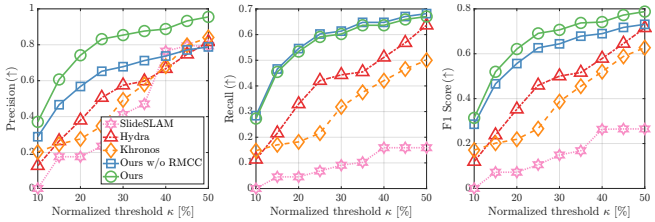


Fig. 12. Precision, recall, and F_1 score as functions of the normalized threshold κ, applied as $\tau_d \leftarrow \kappa \cdot d_{\text{diag}}$, where d_{diag} denotes the half-diagonal length of each object.

G. Quantitative and Qualitative Studies on RMCC

In addition, we provide quantitative and qualitative results on the effect of our RMCC. Object shape prediction models are inherently vulnerable to partial masks or degenerate viewpoints, which often lead to inaccurate shape estimates. Because CRISP operates as a class-agnostic shape estimator, such ambiguous inputs can result in severe intra-class misalignments (Fig. 11(a)) or even generate shapes belonging to entirely different classes (Fig. 11(b)). However, our RMCC module effectively filters out these false cases by rejecting predictions that exhibit insufficient image-space overlap according to the overlap score in (8).

Furthermore, since RMCC is driven by relative geometric proportions rather than absolute physical sizes, we evaluate its performance using a relative threshold $\tau_d \leftarrow \kappa \cdot d_{\text{diag}}$, where d_{diag} is the half-diagonal length of the ground-truth object mesh. As shown in Fig. 12, while RMCC causes a slight drop in recall by rejecting some valid but uncertain estimates, enabling RMCC substantially increases precision across all relative thresholds. Consequently, Hydra++ with RMCC achieves consistently higher F_1 scores across the entire threshold range, confirming its efficacy as a robust post-verification step.

V. Conclusion and Lessons Learned

We have presented *Hydra++*, a scene graph-based system that integrates learned object-level pose and shape estimation within a hierarchical 3D scene graph framework. In particular, we have demonstrated that a pose and shape estimation pipeline can be seamlessly integrated into the system, providing improved object-level geometric representations while maintaining semantic grounding. We have further demonstrated that fusing LiDAR and camera information improves scene-level mesh completeness in outdoor environments.

Our experience also highlights the lack of outdoor benchmarks for object-level mesh reconstruction. Existing public datasets commonly provide point-wise or pixel-wise annotations, but rarely offer scene-scale, instance-level mesh ground truth from a mapping perspective. This limits both training and evaluation of shape estimators such as CRISP in real outdoor settings. Future work will explore generative outdoor scene synthesis to create such benchmarks and improve generalization beyond controlled indoor environments.

References

[1] 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," *Int. J. Robot. Res.*, vol. 40, no. 12–14, pp. 1510–1546, 2021.

[2] N. Hughes, Y. Chang, and L. Carlone, "Hydra: a real-time spatial perception engine for 3D scene graph construction and optimization," in *Proc. Robot.: Sci. Syst.*, 2022.

[3] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison, "SLAM++: Simultaneous localisation and mapping at the level of objects," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2013.

[4] L. Nicholson, M. Milford, and N. Sünderhauf, "QuadricSLAM: Dual quadrics from object detections as landmarks in object-oriented SLAM," *IEEE Robot. Autom. Lett.*, vol. 4, pp. 1–8, 2018.

[5] S. Yang and S. Scherer, "CubeSLAM: Monocular 3-D object SLAM," *IEEE Trans. Robot.*, vol. 35, no. 4, pp. 925–938, 2019.

[6] C. Li, H. Xiao, K. Tateno, F. Tombari, N. Navab, and G. D. Hager, "Incremental scene understanding on dense SLAM," in *Proc. Int. Conf. Intell. Robots Syst.*, 2016, pp. 574–581.

[7] I. Armeni, Z. He, J. Gwak, A. Zamir, M. Fischer, J. Malik, and S. Savarese, "3D scene graph: A structure for unified semantics, 3D space, and camera," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 5664–5673.

[8] N. Hughes, Y. Chang, S. Hu, R. Talak, R. Abdulhai, J. Strader, and L. Carlone, "Foundations of spatial perception for robotics: Hierarchical representations and real-time systems," *Int. J. Robot. Res.*, 2024.

[9] S. Wu, J. Wald, K. Tateno, N. Navab, and F. Tombari, "SceneGraphFusion: Incremental 3D scene graph prediction from RGB-D sequences," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 7515–7525.

[10] D. Honerkamp, M. Büchner, F. Despinoy, T. Welschehold, and A. Valada, "Language-grounded dynamic scene graphs for interactive object search with mobile manipulation," *IEEE Robot. Autom. Lett.*, vol. 9, no. 10, pp. 8298–8305, 2024.

[11] L. Schmid, M. Abate, Y. Chang, and L. Carlone, "Khronos: A unified approach for spatio-temporal metric-semantic SLAM in dynamic environments," in *Proc. Robot.: Sci. Syst.*, 2024.

[12] J. Shi, R. Talak, H. Zhang, D. Jin, and L. Carlone, "CRISP: Object pose and shape estimation with test-time adaptation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2025.

[13] X. Chen, F.-J. Chu, P. Gleize, K. J. Liang, A. Sax, H. Tang, W. Wang, M. Guo, T. Hardin, X. Li *et al.*, "SAM3D: 3DFY anything in images," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2026, pp. 7220–7232.

[14] S. Bowman, N. Atanasov, K. Daniilidis, and G. Pappas, "Probabilistic data association for semantic SLAM," in *Proc. Int. Conf. Robot. Automat.*, 2017, pp. 1722–1729.

[15] N. Atanasov, S. L. Bowman, K. Daniilidis, and G. J. Pappas, "A unifying view of geometry, semantics, and data association in SLAM." in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 5204–5208.

[16] M. Shan, Q. Feng, Y.-Y. Jau, and N. Atanasov, "ELLIPSDF: joint object pose and shape optimization with a bi-level ellipsoid and signed distance function description," in *Proc. Int. Conf. Comput. Vis.*, 2021, pp. 5946–5955.

[17] X. Yu, R. Talak, J. Shi, U. Viereck, I. Gilitschenski, and L. Carlone, "Box pose and shape estimation and domain adaptation for large-scale warehouse automation," *arXiv preprint arXiv:2507.00984*, 2025.

[18] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. Davison, and S. Leutenegger, "MID-Fusion: Octree-based object-level multi-instance dynamic SLAM," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 5231–5237.

[19] J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2019.

[20] M. Liu, C. Xu, H. Jin, L. Chen, M. Varma T, Z. Xu, and H. Su, "One-2-3-45: Any single image to 3D mesh in 45 seconds without per-shape optimization," *Adv. Neural Inf. Process. Syst.*, vol. 36, 2024.

[21] J. Wang, M. Rünz, and L. Agapito, "DSP-SLAM: Object-oriented SLAM with deep shape priors," in *Proc. Int. Conf. 3D Vis.*, 2021, pp. 1362–1371.

[22] X. Liu, J. Lei, A. Prabhu, Y. Tao, I. Spasojevic, P. Chaudhari, N. Atanasov, and V. Kumar, "SlideSLAM: Sparse, lightweight, decentralized metric-semantic SLAM for multi-robot navigation," *IEEE Trans. Robot.*, vol. 41, pp. 6529–6548, 2025.

[23] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone, "3D dynamic scene graphs: Actionable spatial perception with places, objects, and humans," in *Proc. Robot.: Sci. Syst.*, 2020.

[24] H. Bavle, J. L. Sanchez-Lopez, M. Shaheer, J. Civera, and H. Voos, "Situational graphs for robot navigation in structured indoor environments," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 9107–9114, 2022.

[25] ——, "S-Graphs+: Real-time localization and mapping leveraging hierarchical representations," *arXiv preprint arXiv:2212.11770*, 2022.

[26] A. Werby, C. Huang, M. Büchner, A. Valada, and W. Burgard, "Hierarchical Open-Vocabulary 3D Scene Graphs for Language-Grounded Robot Navigation," in *Proc. Robot.: Sci. Syst.*, 2024.

[27] E. Greve, M. Büchner, N. Vödisch, W. Burgard, and A. Valada, "Collaborative dynamic 3D scene graphs for automated driving," in *Proc. Int. Conf. Robot. Automat.*, 2024, pp. 11 118–11 124.

[28] L. Schmid, J. Delmerico, J. L. Schönberger, J. Nieto, M. Pollefeys, R. Siegwart, and C. Cadena, "Panoptic Multi-TSDFs: a flexible representation for online multi-resolution volumetric mapping and long-term dynamic scene consistency," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 8018–8024.

[29] H. Lim, M. Oh, and H. Myung, "Patchwork: Concentric zone-based region-wise ground segmentation with ground likelihood estimation using a 3D LiDAR sensor," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 6458–6465, 2021.

[30] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, "DINOv2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.

[31] D. Maggio, H. Lim, and L. Carlone, "VGGT-SLAM: Dense RGB SLAM optimized on the SL(4) manifold," in *Adv. Neural Inf. Process. Syst.*, 2025.

[32] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ADE20K dataset," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5122–5130.

[33] Y. Tian, Y. Chang, L. Quang, A. Schang, C. Nieto-Granda, J. How, and L. Carlone, "Resilient and distributed multi-robot visual SLAM: Datasets, experiments, and lessons learned," in *Proc. Int. Conf. Intell. Robots Syst.*, 2023.

[34] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *Proc. Int. Conf. Comput. Vis.*, 2023, pp. 4015–4026.

[35] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org