MAP-ADAPT: Real-Time Quality-Adaptive Semantic 3D Maps

Jianhao Zheng[1], Daniel Barath[2], Marc Pollefeys[2,3], and Iro Armeni[1]

[1] Stanford University, Gradient Spaces Lab, USA
[2] ETH Zurich, Computer Vision and Geometry Group, Switzerland
[3] Microsoft Mixed Reality & AI Lab, Switzerland
{jianhao, iarmeni}@stanford.edu
{danielbela.barath, marc.pollefeys}@inf.ethz.ch

Abstract. Creating 3D semantic reconstructions of environments is fundamental to many applications, especially when related to autonomous agent operation (*e.g.*, goal-oriented navigation or object interaction and manipulation). Commonly, 3D semantic reconstruction systems capture the entire scene in the same level of detail. However, certain tasks (*e.g.*, object interaction) require a fine-grained and high-resolution map, particularly if the objects to interact are of small size or intricate geometry. In recent practice, this leads to the entire map being in the same high-quality resolution, which results in increased computational and storage costs. To address this challenge, we propose *MAP-ADAPT*, a real-time method for quality-adaptive semantic 3D reconstruction using RGBD frames. MAP-ADAPT is the first adaptive semantic 3D mapping algorithm that, unlike prior work, generates directly a *single* map with regions of different quality based on both the semantic information and the geometric complexity of the scene. Leveraging a semantic SLAM pipeline for pose and semantic estimation, we achieve comparable or superior results to state-of-the-art methods on synthetic and real-world data, while significantly reducing storage and computation requirements. Code is available at https://map-adapt.github.io/.

Keywords: 3D semantic reconstruction · Quality-adaptive mapping

1 Introduction

Advancements in 3D sensing devices (*e.g.*, Intel RealSense [13], Microsoft Kinect [27], and Orbbec Astra [34]) and semantic understanding [3, 18, 19] have enabled the reconstruction of an increasing number of semantic maps of environments in accuracy and detail. This is particularly useful for autonomous agents since they utilize such maps to perform tasks, *e.g.*, navigation [1,20] and object manipulation [2,41]. In recent practice, the common output of 3D reconstruction systems [33, 35, 39, 51] is a volumetric map of the environment that is uniform in the level of detail (single-resolution map). When the task requires a fine-grained and high-resolution reconstruction, *e.g.*, for interacting with objects of small size or intricate geometry, the resulting map can lead to substantial computation and storage demands, which can be crucial for the operation of agents.

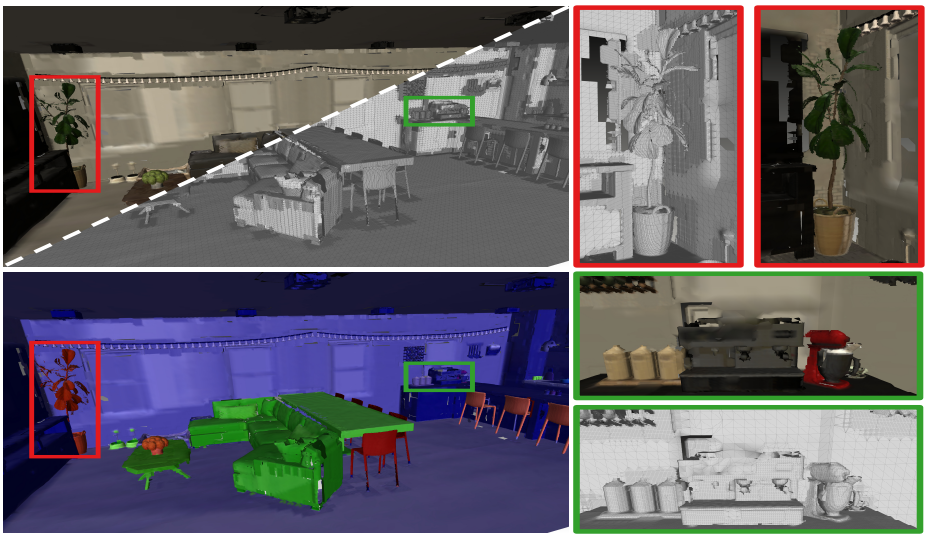


Fig. 1: MAP-ADAPT. Our method generates quality-adaptive semantic 3D maps of environments, where regions of different semantics and geometric complexity are reconstructed in different quality levels. An example map is shown here: 3D reconstructed mesh (top-left) and the semantic quality mask (bottom-left). Mask colors denote three quality levels, where red is *high*, green is *middle*, and blue is *coarse*. A plant reconstructed in high quality due to its semantic label is highlighted (top-right). Though the coffee machine based on its label should appear coarse, it is still mapped in fine resolution due to high geometric complexity (bottom-right).

We approach these shortcomings from the lens of not always needing 'everything in anything', *i.e.*, all information in the same level of detail, and address them by creating the 3D semantic maps in a quality-adaptive manner. Prior work has independently addressed building semantic maps [7,11,24,29,54] and multi-resolution geometric mapping [5, 6, 15, 45, 46, 50, 57] to achieve accurate and memory-efficient reconstructions. Except for [43], no other method has attempted to create quality-adaptive *semantic* 3D maps. This method employs semantics to represent individual object instances in *separate* 3D Truncated Signed Distance Field (TSDF) maps with different resolutions. However, since each map is created independently from the others and due to noisy semantic estimation, multiple maps may occupy the same spatial region without any mechanism to disambiguate across and merge them.

To address these limitations, we propose **MAP-ADAPT**, a real-time method for quality-adaptive semantic 3D reconstruction with RGBD frames. Our main contribution is the *first* adaptive semantic 3D mapping algorithm that generates directly a *single* map with regions of different quality. In comparison to prior work on multi-resolution maps where the resolution is determined by the distance to the camera [50, 57], the quality per region is defined by the semantic label of an object and/or its geometric complexity. Our method is less computationally and storage demanding than single-resolution methods [33] and it is faster and more accurate than the other semantic quality-adaptive

method [43]. Hence, it has practical applicability to autonomous agents due to their limitations on computing, power supply, and storage.

Given a lack of adaptive multi-resolution representations for semantic and 3D geometric data, we develop a new structure that can update reconstructed map regions and their quality level as new observations are received, building on top of an existing voxel hashing method [33]. We also propose a new approach to incrementally update the geometric complexity of the surface in each *single* voxel. Furthermore, we estimate the camera pose and semantics of RGBD frames with a SLAM and a semantic segmentation method respectively, and use this information to build the map in an online manner. We propose a modified mesh generation method based on [52] to create a mesh from our multi-resolution map. Last, we evaluate end-to-end the adaptive semantic reconstruction of MAP-ADAPT and baselines in simulated and real-world environments using two state-of-the-art 3D datasets. Our contributions are summarized as follows:

- A real-time framework that generates a single quality-adaptive map, where areas that belong to different semantic groups and regions with intricate geometric details are distinctly reconstructed.
- A multi-resolution map representation that encapsulates geometric and semantic information and can be incrementally updated with newly acquired observations.
- An adaptive mesh generation approach that can handle voxels and their neighbors in different resolutions.

2 Related Work

Adaptive 3D Semantic Mapping. Our focus is on methods that create real-time maps of the scene at different levels of quality. Prior work has mainly explored the creation of *geometrically* adaptive maps. In [5, 45], 3D information is kept at multiple resolutions simultaneously. The coarse information is then used to regularize the fine resolution levels. This creates a large amount of redundant information, especially when considering large-scale 3D scenes. In [46], the authors develop a SLAM approach that employs surfel-based 3D maps of incoming frames in different resolution levels, which are further associated per level to get the final 3D reconstruction of the scene. In [57], the authors fuse depth frames into a multi-resolution triangular mesh that is adaptively tessellated based on the distance of the camera from the observed surface. Similarly, [50] introduces an octree-based volumetric SLAM pipeline that integrates and renders depth images at an adaptive level of detail based on the camera distance. In [15], the authors use a voxel-hashing approach to bypass the time-consuming traversal of tree structures and generate adaptive maps based on the geometric complexity of the surface. In our work, we address the problem of creating 3D semantic maps that adapt the geometry based on both geometric and semantic information. Our TSDF voxel-based formulation incorporates camera distance to define geometric and semantic accuracy.

In Panoptic Multi-TSDF [43], similar to us, the authors use a TSDF voxel-based structure to acquire a semantic 3D map given RGBD frames. However, they represent each object instance in the scene in a separate TSDF voxel-based map that varies in terms of resolution depending on the semantic category of the object. Although this

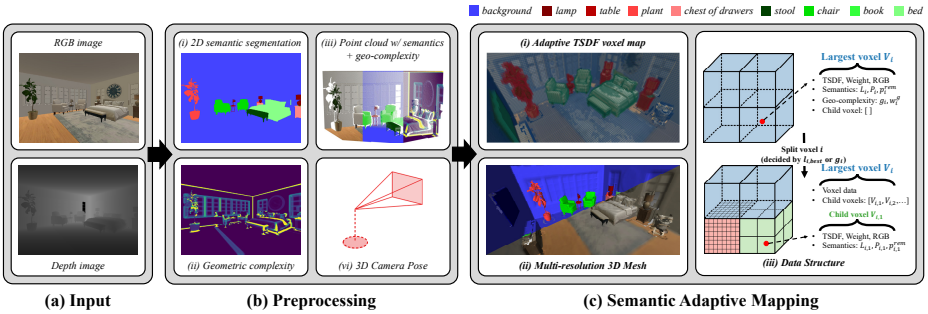


Fig. 2: Overview of MAP-ADAPT. (a) Given RGBD frames, we estimate (b-i) semantic segmentation and (b-iv) camera pose and compute (b-ii) geometric complexity. (c-i) We integrate geometric and semantic information (b-iii) on the TSDF voxel map. The geometric complexity and the semantic label will define the voxel size of that region of the map. (c-ii) shows the multi-resolution mesh output. The adaptive structure we use is shown in (c-iii).

work handles semantic mapping with different resolutions, dividing the scene into multiple maps has certain limitations. Imperfect semantic segmentation and camera pose estimation can lead to duplicate reconstructions of spatial regions in these maps. This occurs because semantic masks may overlap with adjacent categories when projected from 2D to 3D and individual maps are created in isolation without information exchange. This complicates merging the data into a single map due to the ambiguity in semantic interpretation. In contrast, we create a single map representation that handles regions of adaptive resolution as new data points are received and overcomes the above challenge because of the way it represents the scene.

3D Map Representations. There exist multiple ways of representing 3D scenes, ranging from the use of 3D point clouds, to surfels [26, 53], voxels [33, 39], 3D Gaussians [16], sparse representations [31, 55], and neural implicit ones [47, 56]. For generating real-time maps that can operate on autonomous agents and allow them to perform other downstream tasks (*e.g.*, navigation or object manipulation), voxel-based TSDF representations are commonly used. To further allow real-time generation, methods have focused on octrees [12] and voxel hashing [10, 32, 33]. In [33], voxel hashing was shown to be a more efficient method to query voxels compared to octrees [12]. Hence, we build on the Voxblox [33] voxel-hashing TSDF approach and contribute to it with a semantic adaptive structure and a fusion approach for generating and updating 3D semantic maps of adaptive resolution.

Semantic SLAM. Incorporating semantic information into SLAM-generated maps can be categorized into three types of methods: (i) *Object detection-based*: Methods implement object-level detection (*e.g.*, [21, 38]) on RGB images to output 2D bounding boxes. After further processing, they either use a parameterized way to represent the detected object, such as Quadrics [37] and the pose of a pre-modeled object [42], or further perform geometric segmentation on the depth map [9, 48]. (ii) *Semantic segmentation-based*: Methods process semantic segmentation on 2D RGB images and build 3D geometric maps separately. The two outputs are fused with a Bayesian update to generate the semantic map [26, 39]. (iii) *Instance segmentation-based*: Such methods are similar

to (ii). The main difference is that the RGB image is segmented to acquire object instances [25,40]. One exception is the method of Grinvald et al. [8], which first segments a depth image and then utilizes the instance segmentation on an RGB image to refine the previous segments. We follow a semantic segmentation approach that is based on panoptic understanding [26,39], but the proposed method can easily adapt to instances.
Mesh Generation. Marching Cubes [22] is widely used to extract mesh from a voxel-based map. Although it is effective for fixed-size voxel maps [32,33,39], modifications are required for multi-resolution ones. To generate a mesh for a query voxel, a $2 \times 2 \times 2$ cube is formed with its 7 neighbors. [22] requires the latter in the same size, which is not possible at the boundary of different resolutions. [50] proposes to use the coarsest resolution at the voxel boundary to ensure that all 8 voxels exist. However, this ignores fine-level voxels. In contrast, we adapt the idea of [52] on iso-surface extraction to our specific data structure. Such a method leverages information from the voxels at all levels. Although [6] also claims to follow [52]'s approach for mesh generation from their multi-resolution voxel map, no explanation of the implementation is provided.

3 MAP-ADAPT

Given a set of RGBD frames, we use the RGB and depth images to estimate camera pose C_k and predict semantic segmentation map S_k using the RGB images only, where $k = 1, 2, ..., K$ and K is the total number of frames. We employ this information to create a quality-adaptive map in an online manner. Hereafter, $N \in \mathbb{N}^+$ is the total number of semantic labels that the semantic segmentation method can predict, l is a semantic label from this set, and $l_{i,\text{best}}$ is the label with the highest probability in the voxel V_i. Let us consider that the adaptive map has three resolution levels: fine, middle, and coarse.[4] Each semantic label l is associated with a level of the targeted reconstruction quality (*e.g.*, fine) based on user preference. Per map region, the level of resolution is decided based on its semantic label and can also incorporate the geometric complexity of the observed surface. Regarding the latter, thresholds are noted as θ_r where $r \in \{fine, middle, coarse\}$. In the rest of this section, we describe the adaptive mapping process and map representation in detail. An overview of the pipeline is in Fig. 2.
Adaptive Map Representation. Our map representation, as in Voxblox [33], uses a TSDF voxel grid V to implicitly store geometric information, from which the 3D mesh of the mapped scene will be extracted with the use of Marching Cubes [22]. This two forms of maps are shown as (c-i) and (c-ii) in Fig. 2. In addition to the truncated distance, its weight, and color [33], each voxel V_i in our map stores its geometric complexity g_i, a weight w_i^g representing the confidence in g_i, a vector L_i of semantic labels that have been associated with this voxel, a vector P_i of the probabilities corresponding to these semantics, and the probability p_i^{rem} that corresponds to any non-associated semantic labels. We assume a uniform probability distribution for all non-associated labels so that we can store their probabilities in a single scalar p_i^{rem}. Each voxel is initialized with an empty vector for L_i and P_i and the probability $p_i^{\text{rem}} = 1/N$. As new RGB-D frames are processed, the probability of a semantic label l may be updated for that voxel (see

[4] Even though we describe the map assuming three levels of hierarchy, its depth in our implementation can be chosen arbitrarily, depending on the application at hand.

below for an explanation of the update process). If l was previously associated with V_i, only its probability in P_i is updated. Otherwise, l and its probability will be added to the L_i and P_i vectors, respectively. Compared to allocating a single fixed-size vector per V_i for the probabilities of all N semantic classes, even if not associated with this voxel [26, 39], our approach consumes less memory, especially when N is large.

So far, the described map representation is not adaptive. We introduce adaptivity by creating a hierarchy of parent-child voxels from the coarsest resolution (parent) to the finest (child). A given voxel V_i in the voxel grid is initialized in the coarsest resolution when first created, *i.e.*, it is initialized in the largest voxel size. If either the most likely semantic label $l_{i,\text{best}}$ of V_i corresponds to a finer resolution level r or the geometric complexity reaches the threshold $g_i \geq \theta_r$, this voxel will be subdivided by generating a vector of child voxels with the corresponding size of r. Furthermore, if V_i already contains child voxels but both $l_{i,\text{best}}$ and g_i get updated to one of the coarser resolutions, the child voxels will be removed from V_i so that the voxel degrades back to a coarse representation. To avoid the loss of geometric information when the $l_{i,\text{best}}$ is uncertain, child voxels are removed only when $l_{i,\text{best}} \geq 0.95$. Please note that division and merging operations are defined based on the $l_{i,\text{best}}$ and g_i of the voxel in the coarser resolution level. This adaptive resolution structure is shown in Fig. 2 (c-iii).

Incorporating RGB-D Frames. With the depth map, RGB image, pose C_k, and semantic map S_k at frame k, we create a semantically labeled 3D point cloud PC_k in the world coordinate system (Fig. 2 (b-iii)). To avoid losing semantic information, especially when considering the noisy nature of predictions, instead of using a segmentation map that contains per pixel only the l with the highest confidence score [39], we provide at most the four top-scoring semantic labels that have confidence score greater than the threshold $t = 0.1$. These semantic labels and their confidence scores are raycasted to voxels in V per 3D point pc_j in PC_k. Similar to [33], a ray that connects the camera center of frame k with pc_j is used to find those voxels whose absolute value of the truncated signed distance is smaller than their size. This saves computational effort by only updating semantic information on voxels near the surface. We modify the raycasting in [33] for adaptive resolution as described below.

Adaptive Raycasting. We use a modified version of the fast bundled raycasting in [33] but extend it to the resolution-adaptive setting. Before casting a ray on V, we need to decide which points from PC_k may be redundant and hence can be skipped with a minimum loss of information. For a non-adaptive geometric map, a hash 3D grid with resolution $v_{grid} = \alpha v$, where α is a subsampling factor with default value 0.5 and v is the voxel size of the TSDF map, keeps track of points in PC_k that will be used to update V. Specifically, a point in PC_k is discarded if the grid cell it falls into is already occupied by another 3D point originating from the same frame k. Since MAP-ADAPT has multiple resolutions (three in the described scenario), we initialize three grids with $v_{r,grid} = \alpha v_r$, where v_r is the voxel size of quality level r in the TSDF map. Each virtual grid is used to determine whether the point will be utilized to update voxels of the corresponding size. Every point in PC_k will be inserted into all three grids. If the position in $v_{r,grid}$ has already been occupied, this point will not be used to update voxels whose resolutions are level r. However, the same point might integrate information into voxels of another size r' as long as the position in $v_{r',grid}$ is free.

Updating Voxel Probabilities. Assume that M_j is the set of four (or fewer) top-scoring semantic labels for point pc_j and $P_j(l|S_k)$ is the probability of the label of point pc_j to be semantic label l. When assigning semantic information from pc_j in PC_k to a voxel V_i, we use the probabilities that are already associated with pc_j for the top-scoring semantic labels in M_j. For all other labels, we assume a uniform probability distribution. To avoid exceedingly fast convergence to a specific label for V_i, we empirically define a lower bound $\xi = 0.01$. Specifically, $\forall l \notin M_j$, its probability is given by:

$$P_j(l \mid S_k) = \max\left(\xi, \frac{1 - \sum_{m \in M_j} P_j(l_m \mid S_k)}{N - \text{sizeof}(M_j)}\right). \tag{1}$$

Similar to [26,39], when new frames are incorporated in V, a Bayesian update is utilized to update the semantic probabilities of voxel V_i. Given the 3D point pc_j, the probability of a voxel V_i to be semantic label l after k frames $P_i(l \mid S_{1,\ldots,k})$ is updated by the following rules:

$$P_i(l \mid S_{1,\ldots,k}) = \frac{1}{Z} P_i\left(l \mid S_{1,\ldots,k-1}\right) \left[P_j(l \mid S_k)\right]^{w_j}, \tag{2}$$

where Z is a normalization term for the probabilities so that they will sum to 1 and $w_j = 1/z_j^2$ is a weight function that depends on the depth measurement z_j of point pc_j in the depth frame k.

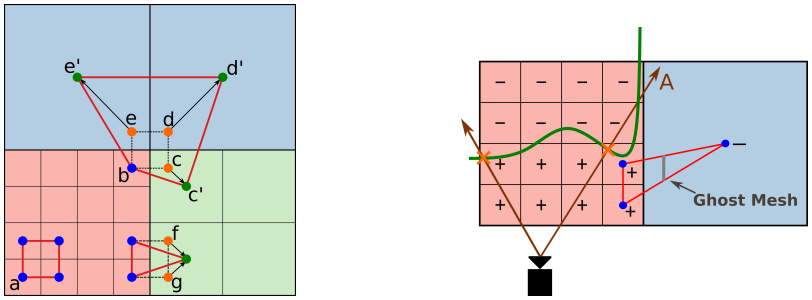
Estimating Geometric Complexity. As certain tasks require increased precision in understanding the geometric details of objects beyond semantic distinctions, we employ a voxel-wise geometric complexity measurement to determine the reconstruction quality level. This involves assessing the change of curvature [36, 41] on the projected points at each frame k and incrementally updating this value in V. For a point pc_j in PC_k, the eigenvalues $\lambda_1^j, \lambda_2^j, \lambda_3^j$ of the respective 3D structure tensor [14] are extracted, with $\lambda_1^j \geq \lambda_2^j \geq \lambda_3^j \geq 0$. The change of curvature at this point CC_j is calculated as $CC_j = \lambda_3^j/(\lambda_1^j + \lambda_2^j + \lambda_3^j)$. For a voxel V_i passed by the ray of pc_j, its geometric complexity g_i and weight w_i^g are updated as follows:

$$\begin{aligned} g_i &\leftarrow \frac{w_i^g g_i + w_j CC_j}{w_i^g + w_j} \\ w_i^g &\leftarrow \min\left(w_i^g + w_j, W_{max}\right) \end{aligned} \tag{3}$$

W_{max} is the same upper bound as in updating TSDF values.

Multi-resolution Mesh Generation. We generate the final 3D mesh with Marching Cubes [22] in a bottom-up fashion. When generating the mesh, we traverse all coarse-resolution voxels V_{coarse}. If a voxel in V_{coarse} has children – *i.e.* is split into a finer resolution, the mesh will be generated on its child voxels. To mesh a voxel with [22], we need the TSDF values and coordinates of its 7 neighbors to form a cube. However, [22] requires all 8 voxels to be in the same resolution, which is not always feasible in our multi-resolution map.

To construct the 8-voxel cube, we initiate the process with voxels at the finest resolution. If any of the 8 voxels is absent at this level, it is substituted by its corresponding



(a) Mesh generation. **(b)** An example of ghost mesh.

Fig. 3: Illustration of forming a cube to generate a mesh from our multi-resolution map. (a) When a neighboring voxel of the queried resolution (***orange node***) does not exist, the corresponding coarser-resolution one (***green node***) will be used instead. (b) A ghost mesh is generated at the boundary of resolutions.

voxel at a coarser resolution. This process is illustrated in Figure 3a using a 2D grid for simplification. Voxel ***a*** shows the typical mesh generation approach in fixed-size maps, where ***a*** and its neighbors all belong to the finest resolution. When attempting to mesh ***b***, which is at the finest level, its neighbors (***c***, ***d***, and ***e***) are not available there. Consequently, we substitute them with their coarse counterparts ***c'***, ***d'***, and ***e'***. This substitution may result in the formation of different geometric structures, such as triangles or lines, instead of hexahedra. For instance, when multiple fine-resolution voxels like ***f*** and ***g*** are substituted by the same coarse-resolution voxel, it leads to collapsed edges where endpoints coincide. As noted in [52], [22] can still process these geometries effectively as if they were regular hexahedra; no mesh is generated along the collapsed edges since both endpoints have the same TSDF value.

The other challenging issue is that non-existent meshes (ghost meshes) may be generated near the surface of objects that occupy voxels in finer resolution. This primarily occurs because the adjacent voxels in free space are in the coarsest resolution, leading to reduced accuracy in their TSDF values. An example is in Figure 3b, where we split a voxel to the finest resolution because it contains a surface with high geometric complexity, while its right neighbor remains in the coarse resolution. When ray ***A*** is integrated into the map, the blue voxel, which is supposed to be empty, will also be updated since the ray passes through a small part of it. As a result, the voxel will be assigned a negative TSDF value. Since two of its neighbors have a positive TSDF value, a ghost mesh will be generated there. To mitigate this problem, when a voxel is split to a finer resolution, we also split all neighboring voxels to the same one. Though it will lead to higher quality reconstruction on regions which should have coarser resolution, it significantly improves the quality of the generated mesh for fine-level semantics.

4 Experiments

We evaluate MAP-ADAPT's performance on creating accurate and complete geometric and semantic 3D maps with adaptive resolution, and compare with the fixed voxel size

Table 1: Evaluation per quality level on HSSD [17]. @XXcm represents the evaluation on the regions of semantics corresponding to the resolution level of XX (cm). Best values per evaluation level are in **bold**.

	Method	Reconstruction Quality (cm)	Completion Error (cm) ↓	Compl. <5cm Ratio (%) ↑	Geometric Error (cm) ↓	Semantic Accuracy (%) ↑	Semantic mIoU (%) ↑
@1cm	Voxblox [33] (fixed)	Fine [1]	**2.49 ± 2.80**	**88.74**	4.14 ± 4.49	12.96	6.62
	Multi-TSDFs [43]	Multi-level [1-4-8]	2.74 ± 4.00	85.59	**4.10 ± 6.53**	8.58	4.86
	MAP-ADAPT-S	**Adaptive [1-4-8]**	2.54 ± 2.92	88.15	4.18 ± 4.62	**13.12**	**6.74**
	MAP-ADAPT-SG	**Adaptive [1-4-8]**	2.53 ± 2.84	88.34	4.19 ± 4.57	**13.12**	**6.74**
@4cm	Voxblox [33] (fixed)	Middle [4]	3.06 ± 3.50	84.39	4.10 ± 4.16	**40.00**	16.01
	Multi-TSDFs [43]	Multi-level [1-4-8]	3.09 ± 3.83	83.29	4.18 ± 6.46	10.57	6.41
	MAP-ADAPT-S	**Adaptive [1-4-8]**	2.89 ± 3.45	86.12	4.05 ± 4.19	39.69	**16.26**
	MAP-ADAPT-SG	**Adaptive [1-4-8]**	**2.67 ± 3.25**	**88.04**	**3.85 ± 4.13**	39.88	16.21
@8cm	Voxblox [33] (fixed)	Coarse [8]	3.59 ± 3.59	77.93	4.57 ± 6.11	**60.38**	**21.46**
	Multi-TSDFs [43]	Multi-level [1-4-8]	3.42 ± 3.79	79.86	**4.05 ± 5.89**	49.59	8.85
	MAP-ADAPT-S	**Adaptive [1-4-8]**	3.43 ± 3.47	79.94	4.53 ± 5.95	**60.38**	21.18
	MAP-ADAPT-SG	**Adaptive [1-4-8]**	**3.10 ± 3.27**	**83.56**	4.53 ± 5.89	**60.38**	21.17

Voxblox [33] at different resolution levels, as well as with Panoptic Multi-TSDFs [43]. We choose the following three levels of quality (voxel size): fine (1 cm), middle (4 cm), and coarse (8 cm). We use all three in the adaptive methods (ours and [43]), whereas for the fixed-size one, we compare to three different instantiations of it, one per resolution. Results from two versions of MAP-ADAPT are provided; **MAP-ADAPT-S** decides to divide a voxel only based on its semantic label, whereas **MAP-ADAPT-SG** decides based on semantic label and/or geometric complexity.

We report results on the Habitat Synthetic Scene Dataset (HSSD) [17] and on the real-world ScanNet [4] dataset. The threshold of geometric complexity is chosen as $\theta_{middle} = 0.05, \theta_{fine} = 0.1$. Since the motivation of the system is task-driven, giving users the freedom to choose which categories to reconstruct in fine quality and which are unimportant, in our experiments we randomly allocate semantic categories per level of quality and we repeat this 5 times; results are averaged over them. For HSSD, we randomly assign the 28 semantic categories provided into the three levels of quality. For the 40 NYUv2 [44] labels used in ScanNet, we allocate those corresponding to the HSSD categories to the same quality level and randomly assign the rest. In the supplementary material, we provide results with allocating semantics per quality level based on their physical size.

We employ the commonly used ORB-SLAM2 [28] as the visual SLAM module for its robust and real-time behavior; any other SLAM approach could also be used. We employ the Light-weight Refinenet [30] as the segmentation module[5], for allowing real-time processing while providing good segmentation results on unseen data. Similarly, other segmentation methods could be used, especially if processing time is not a concern. We sample training and validation data from the 125 HSSD scenes in the

[5] Even though we demonstrate MAP-ADAPT with object categories, other semantic information can be used, *e.g.*, material, function, change.

train split to train a Lightweight RefineNet model for our experiments on this dataset. Provided by [30], a pre-trained model on the NYUv2 dataset is used for ScanNet.

Metrics. For geometric evaluation, we report: (i) completion error (cm), *i.e.*, the mean Euclidean distance of all ground truth (GT) 3D points from the closest reconstructed ones; (ii) completion ratio for all GT points that have less than 5 cm distance from the closest reconstructed point; and (iii) geometric error (cm), *i.e.*, the mean Euclidean distance of all reconstructed points from the closest GT ones. The reconstructed 3D points are sampled from the generated mesh. The GT points are the aggregated projections from all depth frames, using GT camera pose. The geometric metrics are calculated separately for 3 different quality levels. Each GT point will be classified as 1cm, 4cm, or 8cm based on its GT semantic label. For each point P_i sampled from the reconstructed mesh, we identify the nearest point P_{gt} in the GT map. We then evaluate P_i based on the level corresponding to the semantic label of that closest P_{gt} regardless of the predicted semantic label of P_i. For each of the three semantic levels, evaluation is between the sampled points and GT points based on the latter's quality level. For semantic evaluation, we follow standard approaches [39] and report the overall portion of correctly labeled voxels (Accuracy) and the mean Intersection over Union (mIoU). We report map size in megabytes (MB) and runtime in milliseconds (ms).

Experimental Setup. All experiments are performed on an AMD Ryzen 7 5800H CPU. The only component that requires GPU is the 2D semantic segmentation, which takes 37ms per frame on a GeForce RTX 2080 GPU for Light-weight Refinenet [30].

HSSD Dataset. The HSSD dataset [17] consists of high-quality 3D scenes on the scale of an entire residence with fully human-authored 3D interiors. To generate sequences of frames for SLAM-based reconstruction, the dataset is commonly used within the Habitat [23, 49] simulation environment, which can render RGBD frames from the underlying 3D model given arbitrary 3D camera poses. We manually record camera trajectories in the scenes and use the rendered RGBD frames in our experiments. We develop our method on the training scenes and evaluate on the *open* validation scenes without parameter tuning. We create 43 subscenes from the validation split and ensure they contain at least one semantic category per quality level in each subscene. Statistics on these scenes are in the supplementary material.

Results on geometric and semantic evaluation are shown in Table 1. We employ colors to differentiate the evaluation of regions from different quality levels and report only directly comparable methods; *e.g.*, at *Eval. @8cm*, MAP-ADAPT and [43] are directly comparable with the Voxblox fixed-size on 8cm. Results for the fixed-size methods on other resolutions are included in the supplementary material. We compute metrics per quality level *only* based on the GT semantic regions that correspond to this level.

As shown in Table 1, both versions of MAP-ADAPT achieve performance similar to fixed (1cm) reconstruction in regions where semantics are at the finest level. In regions of semantics belonging to middle and coarse quality, MAP-ADAPT-S performs slightly better than the corresponding fixed size [33] since we split the neighboring voxels of fine-quality semantic voxels, thus these regions will be closer to ground truth points. MAP-ADAPT-SG outperforms all methods in terms of completion error in the middle and coarse semantic regions as it generates finer resolution voxels in regions with high geometric complexity even if their semantics are not allocated in the fine-quality

set. The geometric error of MAP-ADAPT-SG does not show an advantage and is even higher than [43] in the coarsest regions due to errors in the estimated camera pose. A detailed reconstruction will lead to even higher geometric error if it is reconstructed in the wrong position (see Section 4.1 for GT camera pose results). For the map reconstructed using the fixed size method [33] with the finest quality (1 cm), the geometric error in the *coarse* semantic regions is also large ($5.11 \pm 6.37cm$). This result is reported in the supplementary material. In contrast, [43] generates a relatively incomplete reconstruction (higher completion error) in these challenging regions. This means there are fewer points from which to compute the geometric error, which partially explains the lower values. The other issue of [43] is that it generates overlapping mesh regions across the individual semantic maps due to the noisy semantic estimation, as explained in the related work. As a result, [43] performs significantly worse than MAP-ADAPT in all semantic evaluation metrics.

An example of the generated map is shown in Figure 4 (top). The fixed-size Voxblox versions provide less accurate geometry on the overall map as the quality level goes from fine to coarse. This is visible with the increasingly brighter colored regions for completion and geometric errors. For Muti-TSDFs [43] and MAP-ADAPT, we can observe the adaptive reconstruction from the various errors per semantic quality level. The red regions (fine quality) in the GT semantic map have darker colors in the visualizations of completion and geometric error, and the blue regions (coarse quality) are brighter in the error map. Comparing ours with [43], the completion error maps of MAP-ADAPT-S and MAP-ADAPT-SG are darker throughout the scene. On the geometric error map, we generally have a better result. However, objects with high error (*e.g.* chandelier) are not reconstructed in [43], explaining why it has less geometric error on average, as stated above. In the geometric complexity color map, we observe that MAP-ADAPT-SG manages to capture regions having rich geometric information and those regions have much less error compared to MAP-ADAPT-S. We highlighted regions with high geometric complexity (guitar) and with high-quality semantics (plant), where MAP-ADAPT-SG generates the most detailed and sharp reconstruction.

We also report the map size and runtime per method in Table 2. Compared to Voxblox (1cm), MAP-ADAPT-S occupies 3.5 times less memory and is also faster in updating the TSDF values. MAP-ADAPT-SG needs more storage and time since it reconstructs more high-quality regions but still consumes less than Voxblox (1cm). In contrast, [43] takes substantially more time to perform TSDF updates since [43] generates multiple TSDF maps per instance and requires an additional process to track. Both versions of MAP-ADAPT need more time to generate the mesh due to the complex generation of meshes on the border of different voxel resolutions. However, mesh generation is only executed once at the end of reconstruction. We provide statistics and analysis on voxel percentage per quality level in the supplementary material.

ScanNet Dataset. To understand the behavior of MAP-ADAPT given real-world RGBD frames, we evaluate on the ScanNet dataset [4]. It consists of 3D scenes on the scale of a room and includes 3D mesh reconstructions, as well as the sequences of RGBD frames that were used for the reconstruction. We evaluate our approach on 38 randomly selected scenes from the *open* validation set.[6] The results are in Table 3. We can observe

[6] We employ the validation set since the test set does not have publicly available annotations.

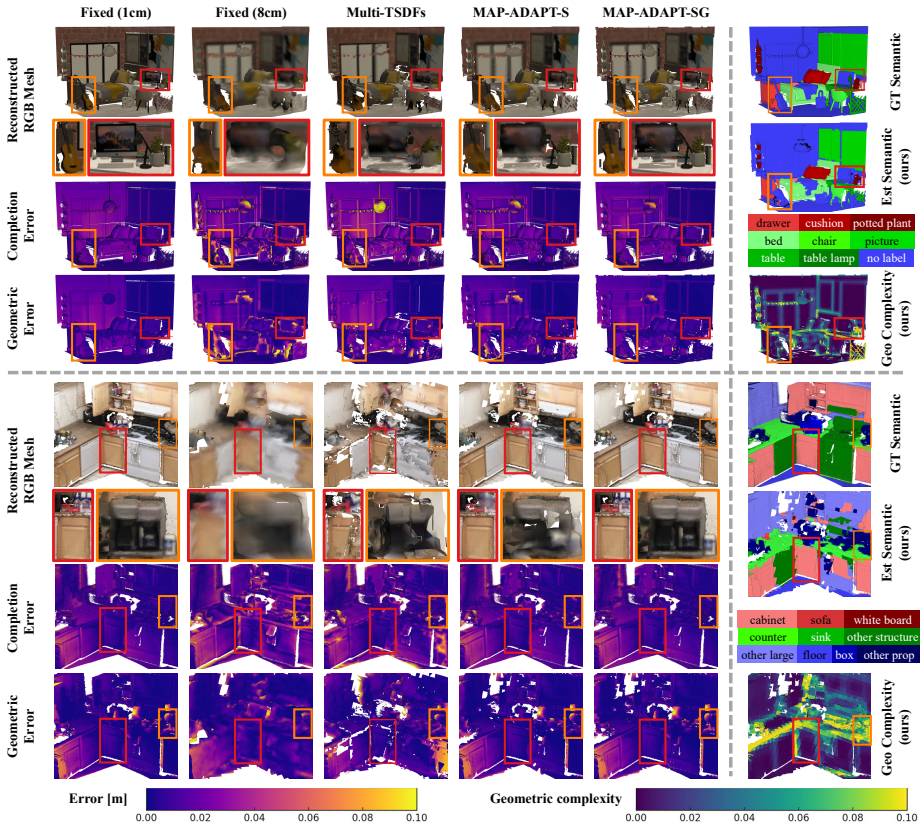


Fig. 4: Reconstruction results per method. Top example is on HSSD and bottom one on ScanNet datasets. Geometric and completion errors are shown as heatmaps; the darker the color, the closer to the GT geometry. For semantic map, results are colorized per quality level; different semantics in the same quality level range from brighter to darker. Another heatmap is used to show the estimated geometric complexity. We highlight regions that are classified into high-quality semantics (*red block*) or have large geometric variance (*orange block*). *Best viewed on screen.*

that all methods perform less well on this real-world dataset, given the blurriness in the frames and noisy sensors. Despite this, MAP-ADAPT achieves comparable results to fix-size (1cm) on fine-quality regions and performs better in semantic accuracy and completion error on coarser regions. Results of [43] are similar to the HSSD dataset. An example of the generated maps is in Figure 4 (bottom). Fixed-size Voxblox has a similar behavior as on HSSD, and so does ours – *e.g.*, error maps are comparable per quality level. MAP-ADAPT-S and MAP-ADAPT-SG still provide lower completion error over all quality levels vs. [43]. The map reconstructed by [43] exhibits a more irregular structure with more holes in the cabinet and several ghost meshes, indicating that it is more affected by the noisy pose estimation and depth data.

Table 2: Evaluation on map size and runtime. *Best* values are **bold**. *Best* of multi-resolution methods are in **underlined bold**. Note that update TSDF is processed at each frame, whereas mesh generation only needs to be executed once at the end.

Method	Map Size (MB) ↓	Runtime (ms) ↓ Update TSDF	Generate Mesh
Voxblox [33] (1cm)	1225.30	89.61 ± 14.16	631.66 ± 307.50
Voxblox [33] (4cm)	60.39	46.02 ± 7.47	31.76 ± 11.15
Voxblox [33] (8cm)	**12.53**	**39.86** ± **6.40**	**8.83** ± **2.90**
Multi-TSDFs [43]	266.91	201.45 ± 212.02	**203.81** ± **155.16**
MAP-ADAPT-S	**265.21**	**54.62** ± **10.99**	638.55 ± 384.82
MAP-ADAPT-SG	469.85	71.79 ± 11.87	1252.54 ± 606.16

Table 3: Evaluation per quality level on Scannet [4]. @XXcm represents the evaluation on the regions of semantics corresponding to the resolution level of XX (cm). Best values per evaluation level are in **bold**.

	Method	Reconstruction Quality (cm)	Completion Error (cm) ↓	Compl. <5cm Ratio (%) ↑	Geometric Error (cm) ↓	Semantic Accuracy (%) ↑	Semantic mIoU (%) ↑
@1cm	Voxblox [33] (fixed)	Fine [1]	**3.21** ± **4.92**	**82.61**	7.08 ± 13.38	10.36	**6.60**
	Multi-TSDFs [43]	Multi-level [1-4-8]	3.75 ± 5.69	77.42	**5.53** ± **10.47**	6.55	4.41
	MAP-ADAPT-S	**Adaptive [1-4-8]**	3.36 ± 5.20	81.57	6.31 ± 11.51	**10.40**	**6.60**
	MAP-ADAPT-SG	**Adaptive [1-4-8]**	3.27 ± 5.03	82.27	6.84 ± 12.99	10.36	6.59
@4cm	Voxblox [33] (fixed)	Middle [4]	4.93 ± 6.52	69.52	7.90 ± 14.80	**9.07**	**5.76**
	Multi-TSDFs [43]	Multi-level [1-4-8]	4.43 ± 6.80	74.94	**6.95** ± **13.28**	4.47	3.23
	MAP-ADAPT-S	**Adaptive [1-4-8]**	4.24 ± 6.22	75.62	8.02 ± 14.84	8.89	5.71
	MAP-ADAPT-SG	**Adaptive [1-4-8]**	**3.91** ± **6.02**	**78.82**	8.58 ± 16.20	9.00	5.73
@8cm	Voxblox [33] (fixed)	Coarse [8]	6.48 ± 7.30	55.36	11.43 ± 17.92	**19.05**	**8.94**
	Multi-TSDFs [43]	Multi-level [1-4-8]	5.23 ± 7.02	67.00	**9.02** ± **15.02**	14.10	5.28
	MAP-ADAPT-S	**Adaptive [1-4-8]**	5.01 ± 6.64	67.77	9.94 ± 16.07	**19.05**	**8.94**
	MAP-ADAPT-SG	**Adaptive [1-4-8]**	**4.27** ± **6.19**	**74.51**	9.48 ± 15.82	**19.05**	**8.94**

4.1 Ablation Studies

In this section, to further evaluate our design choices, we provide additional experiments on all 43 scenes from the HSSD dataset with 1 random semantic quality allocation.

GT pose and semantics: In Table 4, we further evaluate the geometric metrics of all methods when using GT camera pose and semantic information as input. A full table with semantic evaluation is in the supplementary material. As with estimated input, MAP-ADAPT-S has similar results to the corresponding fixed-size Voxblox on regions of different quality. Although the results of all methods are significantly improved, MAP-ADAPT-S and MAP-ADAPT-SG outperform Multi-TSDF [43] on both geometric and completion errors in the fine quality. Without the noise of estimated poses, objects will not be reconstructed in wrong positions. Therefore, [43] cannot benefit from an incomplete reconstruction when computing the geometric error. In the coarser region, multi-TSDFs [43] achieve less completion and geometric error due to a more accurate TSDF estimation in large voxels. Nevertheless, this region requires less focus, since the objective is to maintain rough reconstruction on them and build a higher quality map for other semantics.

Table 4: Ablation Study. Results on HSSD with GT camera pose and 2D semantic segmentation. We also investigate the impact of adaptive raycasting and neighborhood split. Best values per evaluation level are in **bold**, second best in **underlined bold**.

	Method	Reconstruction Quality (cm)	Completion Error (cm) ↓	Compl. <5cm Ratio (%) ↑	Geometric Error (cm) ↓
@1cm	Voxblox [33] (fixed)	Fine [1]	**0.29 ± 0.22**	**99.99%**	**0.36 ± 0.37**
	Multi-TSDFs [43]	Multi-level [1-4-8]	0.34 ± 0.56	99.71%	0.79 ± 1.62
	MAP-ADAPT-S	**Adaptive [1-4-8]**	**0.27 ± 0.22**	**99.99%**	0.37 ± 0.42
	MAP-ADAPT-SG	**Adaptive [1-4-8]**	0.29 ± 0.23	**99.99%**	**0.37 ± 0.41**
	w/o adaptive raycasting	**Adaptive [1-4-8]**	0.40 ± 0.31	99.98%	0.38 ± 0.35
	w/o neighbor splitting	**Adaptive [1-4-8]**	0.29 ± 0.27	**99.98%**	0.67 ± 2.01
@4cm	Voxblox [33] (fixed)	Middle [4]	0.92 ± 1.15	98.55%	1.96 ± 1.96
	Multi-TSDFs [43]	Multi-level [1-4-8]	0.99 ± 2.39	96.85%	1.57 ± 2.20
	MAP-ADAPT-S	**Adaptive [1-4-8]**	0.85 ± 1.13	98.66%	1.83 ± 1.93
	MAP-ADAPT-SG	**Adaptive [1-4-8]**	**0.49 ± 0.64**	**99.83%**	**0.90 ± 1.33**
	w/o adaptive raycasting	**Adaptive [1-4-8]**	**0.47 ± 0.54**	**99.89%**	**0.84 ± 1.30**
	w/o neighbor splitting	**Adaptive [1-4-8]**	0.72 ± 0.88	99.59%	1.99 ± 2.57
@8cm	Voxblox [33] (fixed)	Coarse [8]	1.43 ± 1.77	95.54%	1.42 ± 2.10
	Multi-TSDFs [43]	Multi-level [1-4-8]	0.87 ± 1.92	98.13%	0.75 ± 1.47
	MAP-ADAPT-S	**Adaptive [1-4-8]**	1.32 ± 1.64	96.33%	1.24 ± 1.90
	MAP-ADAPT-SG	**Adaptive [1-4-8]**	**0.85 ± 1.13**	**98.90%**	**0.73 ± 1.18**
	w/o adaptive raycasting	**Adaptive [1-4-8]**	**0.86 ± 1.14**	**98.87%**	**0.72 ± 1.19**
	w/o neighbor splitting	**Adaptive [1-4-8]**	1.19 ± 1.36	97.61%	1.24 ± 1.87

Adaptive raycasting: Table 4 shows results on MAP-ADAPT-SG without adaptive raycasting, *i.e.*, using a single virtual grid for coarse level (8 cm) to decide if a point should be updated to voxels of all resolutions. Compared to using adaptive raycasting, results in coarser regions are not affected. However, completion error increases in fine regions where many holes appear. Visualization is in supplementary material.
Neighbor splitting: We provide results of MAP-ADAPT-SG without splitting neighboring voxels to the same resolution of a query voxel when that gets split to a finer resolution. In fine regions, although MAP-ADAPT-SG without split achieves a similar completion error, it has a significantly higher geometric error due to ghost meshes generated at the boundaries of voxels in different resolutions.

5 Conclusion

We present MAP-ADAPT, the first real-time quality-adaptive semantic 3D reconstruction method that creates a single map with regions of different quality levels. We showcase its performance in an end-to-end reconstruction pipeline on a simulated and a real-world dataset. When compared to baselines, it provides a lightweight semantic 3D map that is comparable or superior in geometric and semantic accuracy to using a fixed-sized map. Compared to the only other method that creates maps of different resolutions leveraging semantic information [43] – albeit individual object-instance-based ones, our method generates more detailed and complete reconstructions without duplicate information across resolutions.

Acknowledgement. This project was supported by the ETH RobotX research grant.

References

1. Aotani, Y., Ienaga, T., Machinaka, N., Sadakuni, Y., Yamazaki, R., Hosoda, Y., Sawahashi, R., Kuroda, Y.: Development of autonomous navigation system using 3d map with geometric and semantic information. Journal of Robotics and Mechatronics **29**(4), 639–648 (2017)
2. Breyer, M., Chung, J.J., Ott, L., Siegwart, R., Nieto, J.: Volumetric grasping network: Real-time 6 dof grasp detection in clutter. In: Conference on Robot Learning. pp. 1602–1611. PMLR (2021)
3. Cai, Y., Chen, X., Zhang, C., Lin, K.Y., Wang, X., Li, H.: Semantic scene completion via integrating instances and scene in-the-loop. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 324–333 (2021)
4. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5828–5839 (2017)
5. Fuhrmann, S., Goesele, M.: Fusion of depth maps with multiple scales. ACM Transactions on Graphics (TOG) **30**(6), 1–8 (2011)
6. Funk, N., Tarrio, J., Papatheodorou, S., Popović, M., Alcantarilla, P.F., Leutenegger, S.: Multi-resolution 3d mapping with explicit free space representation for fast and accurate mobile robot motion planning. IEEE Robotics and Automation Letters **6**(2), 3553–3560 (2021)
7. Grinvald, M., Furrer, F., Novkovic, T., Chung, J.J., Cadena, C., Siegwart, R., Nieto, J.: Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery. IEEE Robotics and Automation Letters **4**(3), 3037–3044 (July 2019). https://doi.org/10.1109/LRA.2019.2923960
8. Grinvald, M., Furrer, F., Novkovic, T., Chung, J.J., Cadena, C., Siegwart, R., Nieto, J.: Volumetric instance-aware semantic mapping and 3d object discovery. IEEE Robotics and Automation Letters **4**(3), 3037–3044 (2019)
9. Hachiuma, R., Pirchheim, C., Schmalstieg, D., Saito, H.: Detectfusion: Detecting and segmenting both known and unknown dynamic objects in real-time slam. arXiv preprint arXiv:1907.09127 (2019)
10. Han, L., Gao, F., Zhou, B., Shen, S.: Fiesta: Fast incremental euclidean distance fields for online motion planning of aerial robots. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4423–4430. IEEE (2019)
11. Han, M., Zhang, Z., Jiao, Z., Xie, X., Zhu, Y., Zhu, S.C., Liu, H.: Reconstructing interactive 3d scenes by panoptic mapping and cad model alignments pp. 12199–12206 (2021). https://doi.org/10.1109/ICRA48506.2021.9561546
12. Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W.: Octomap: An efficient probabilistic 3d mapping framework based on octrees. Autonomous robots **34**, 189–206 (2013)
13. Intel: Realsense depth camera d435i. https://www.intelrealsense.com/depthcamera-d435i/
14. Jutzi, B., Gross, H.: Nearest neighbour classification on laser point clouds to gain object structures from buildings. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences **38**(Part 1), 4–7 (2009)
15. Kähler, O., Prisacariu, V., Valentin, J., Murray, D.: Hierarchical voxel block hashing for efficient integration of depth images. IEEE Robotics and Automation Letters **1**(1), 192–197 (2015)

16. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (July 2023), https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/
17. Khanna*, M., Mao*, Y., Jiang, H., Haresh, S., Shacklett, B., Batra, D., Clegg, A., Undersander, E., Chang, A.X., Savva, M.: Habitat Synthetic Scenes Dataset (HSSD-200): An Analysis of 3D Scene Scale and Realism Tradeoffs for ObjectGoal Navigation. arXiv preprint (2023)
18. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. arXiv preprint arXiv:2304.02643 (2023)
19. Lai, X., Chen, Y., Lu, F., Liu, J., Jia, J.: Spherical transformer for lidar-based 3d recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17545–17555 (2023)
20. Lin, Y., Gao, F., Qin, T., Gao, W., Liu, T., Wu, W., Yang, Z., Shen, S.: Autonomous aerial navigation using monocular visual-inertial fusion. Journal of Field Robotics **35**(1), 23–51 (2018)
21. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. pp. 21–37. Springer (2016)
22. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. ACM siggraph computer graphics **21**(4), 163–169 (1987)
23. Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., Parikh, D., Batra, D.: Habitat: A Platform for Embodied AI Research. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2019)
24. Mascaro, R., Teixeira, L., Chli, M.: Volumetric instance-level semantic mapping via multi-view 2d-to-3d label diffusion. IEEE Robotics and Automation Letters **7**(2), 3531–3538 (2022)
25. McCormac, J., Clark, R., Bloesch, M., Davison, A., Leutenegger, S.: Fusion++: Volumetric object-level slam. In: 2018 international conference on 3D vision (3DV). pp. 32–41. IEEE (2018)
26. McCormac, J., Handa, A., Davison, A., Leutenegger, S.: Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In: 2017 IEEE International Conference on Robotics and automation (ICRA). pp. 4628–4635. IEEE (2017)
27. Microsoft: Azure kinect dk. https://azure.microsoft.com/en-us/products/kinect-dk
28. Mur-Artal, R., Tardós, J.D.: Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE transactions on robotics **33**(5), 1255–1262 (2017)
29. Narita, G., Seno, T., Ishikawa, T., Kaji, Y.: Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4205–4212 (2019). https://doi.org/10.1109/IROS40897.2019.8967890
30. Nekrasov, V., Shen, C., Reid, I.: Light-weight refinenet for real-time semantic segmentation. arXiv preprint arXiv:1810.03272 (2018)
31. Nicholson, L., Milford, M., Sünderhauf, N.: Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam. IEEE Robotics and Automation Letters **4**(1), 1–8 (2018)
32. Nießner, M., Zollhöfer, M., Izadi, S., Stamminger, M.: Real-time 3d reconstruction at scale using voxel hashing. ACM Transactions on Graphics (ToG) **32**(6), 1–11 (2013)

33. Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., Nieto, J.: Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2017)
34. Orbbec: Astra series. https://www.orbbec.com/products/structured-light-camera/astra-series/
35. Pan, Y., Kompis, Y., Bartolomei, L., Mascaro, R., Stachniss, C., Chli, M.: Voxfield: Non-projective signed distance fields for online planning and 3d reconstruction. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 5331–5338. IEEE (2022)
36. Pauly, M., Gross, M., Kobbelt, L.P.: Efficient simplification of point-sampled surfaces. In: IEEE Visualization, 2002. VIS 2002. pp. 163–170. IEEE (2002)
37. Qian, Z., Patath, K., Fu, J., Xiao, J.: Semantic slam with autonomous object-level data association. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). pp. 11203–11209. IEEE (2021)
38. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)
39. Rosinol, A., Abate, M., Chang, Y., Carlone, L.: Kimera: an open-source library for real-time metric-semantic localization and mapping. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 1689–1696. IEEE (2020)
40. Runz, M., Buffier, M., Agapito, L.: Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In: 2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR). pp. 10–20. IEEE (2018)
41. Rusu, R.B.: Semantic 3d object maps for everyday manipulation in human living environments. KI-Künstliche Intelligenz **24**, 345–348 (2010)
42. Salas-Moreno, R.F., Newcombe, R.A., Strasdat, H., Kelly, P.H., Davison, A.J.: Slam++: Simultaneous localisation and mapping at the level of objects. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1352–1359 (2013)
43. Schmid, L., Delmerico, J., Schönberger, J., Nieto, J., Pollefeys, M., Siegwart, R., Cadena, C.: Panoptic multi-tsdfs: a flexible representation for online multi-resolution volumetric mapping and long-term dynamic scene consistency. In: 2022 IEEE International Conference on Robotics and Automation (ICRA). pp. 8018–8024 (2022). https://doi.org/10.1109/ICRA46639.2022.9811877
44. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12. pp. 746–760. Springer (2012)
45. Steinbrücker, F., Sturm, J., Cremers, D.: Volumetric 3d mapping in real-time on a cpu. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). pp. 2021–2028. IEEE (2014)
46. Stückler, J., Behnke, S.: Multi-resolution surfel maps for efficient dense 3d modeling and tracking. Journal of Visual Communication and Image Representation **25**(1), 137–147 (2014)
47. Sucar, E., Liu, S., Ortiz, J., Davison, A.J.: imap: Implicit mapping and positioning in real-time. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6229–6238 (2021)
48. Sünderhauf, N., Pham, T.T., Latif, Y., Milford, M., Reid, I.: Meaningful maps with object-oriented semantic mapping. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 5079–5085. IEEE (2017)
49. Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D., Maksymets, O., Gokaslan, A., Vondrus, V., Dharur, S., Meier,

F., Galuba, W., Chang, A., Kira, Z., Koltun, V., Malik, J., Savva, M., Batra, D.: Habitat 2.0: Training home assistants to rearrange their habitat. In: Advances in Neural Information Processing Systems (NeurIPS) (2021)
50. Vespa, E., Funk, N., Kelly, P.H., Leutenegger, S.: Adaptive-resolution octree-based volumetric slam. In: 2019 International Conference on 3D Vision (3DV). pp. 654–662. IEEE (2019)
51. Vespa, E., Nikolov, N., Grimm, M., Nardi, L., Kelly, P.H., Leutenegger, S.: Efficient octree-based volumetric slam supporting signed-distance and occupancy mapping. IEEE Robotics and Automation Letters **3**(2), 1144–1151 (2018)
52. Wald, I.: A simple, general, and gpu friendly method for computing dual mesh and iso-surfaces of adaptive mesh refinement (amr) data. arXiv preprint arXiv:2004.08475 (2020)
53. Whelan, T., Leutenegger, S., Salas-Moreno, R., Glocker, B., Davison, A.: Elasticfusion: Dense slam without a pose graph. Robotics: Science and Systems (2015)
54. Wu, S.C., Wald, J., Tateno, K., Navab, N., Tombari, F.: Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences. In: CVPR. pp. 7515–7525 (June 2021)
55. Yang, S., Scherer, S.: Cubeslam: Monocular 3-d object slam. IEEE Transactions on Robotics **35**(4), 925–938 (2019)
56. Zhu, Z., Peng, S., Larsson, V., Xu, W., Bao, H., Cui, Z., Oswald, M.R., Pollefeys, M.: Nice-slam: Neural implicit scalable encoding for slam. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12786–12796 (2022)
57. Zienkiewicz, J., Tsiotsios, A., Davison, A., Leutenegger, S.: Monocular, real-time surface reconstruction using dynamic level of detail. In: 2016 Fourth International Conference on 3D Vision (3DV). pp. 37–46. IEEE (2016)