SceneGraphFusion: Incremental 3D Scene Graph Prediction from RGB-D Sequences

Shun-Cheng Wu[1] Johanna Wald[1] Keisuke Tateno[2] Nassir Navab[1] Federico Tombari[1,2]

[1]Technische Universität München [2]Google

shunchengwu.github.io/SceneGraphFusion

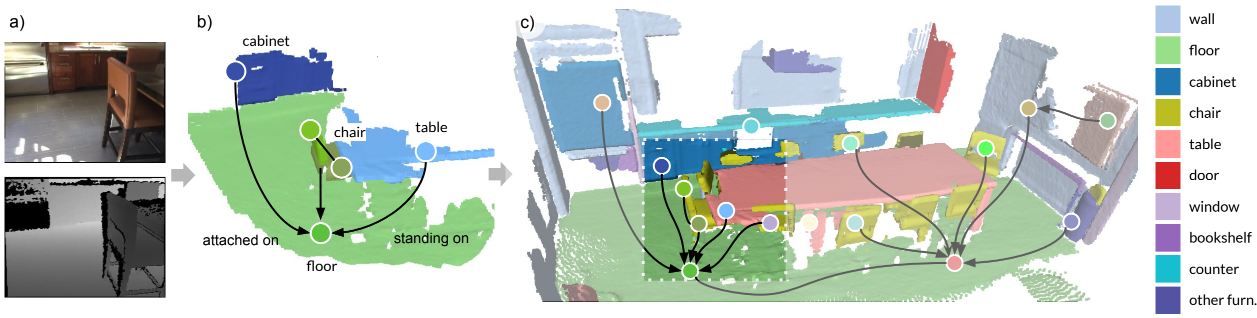

Figure 1. We create a globally consistent 3D scene graph b) by fusing predictions of a graph neural network (GNN) from an incremental geometric segmentation created from an RGB-D sequence a). Our method merges nodes on the same object instance and naturally grows and improves over time when new segments and surfaces are discovered, see c). As a by-product, our method produces accurate panoptic segmentation of large-scale 3D scans. The nodes represent the different object segments.

Abstract

Scene graphs are a compact and explicit representation successfully used in a variety of 2D scene understanding tasks. This work proposes a method to incrementally build up semantic scene graphs from a 3D environment given a sequence of RGB-D frames. To this end, we aggregate PointNet features from primitive scene components by means of a graph neural network. We also propose a novel attention mechanism well suited for partial and missing graph data present in such an incremental reconstruction scenario. Although our proposed method is designed to run on submaps of the scene, we show it also transfers to entire 3D scenes. Experiments show that our approach outperforms 3D scene graph prediction methods by a large margin and its accuracy is on par with other 3D semantic and panoptic segmentation methods while running at 35Hz.

1. Introduction

High-level scene understanding is a fundamental task in computer vision required for many applications in fields such as robotics and augmented or mixed reality. Boosted by the availability of inexpensive depth sensors, real-time dense SLAM algorithms [33, 21, 35, 56] and large scale 3D datasets [5, 53], the research focus has shifted from reconstructing the 3D scene geometry to enhancing the 3D maps with semantic information about scene components. Several methods have deployed a neural network to process a complete 3D scan of a scene [5, 10, 41, 39, 24, 18, 17, 16, 9]. However, these all require 3D geometry as prior information and they typically operate in an offline fashion, *i.e.* without satisfying real-time requirements, which are fundamental for many real-world applications. Real-time scene understanding that incrementally built 3D scans poses important challenges such as handling partial, incomplete, and ambiguous scene geometry where object shapes may change dramatically over time. Learning a robust 3D feature that can cope with this variability is difficult. Furthermore, fusing multiple, potentially contradictory network predictions to ensure consistency in the global map, is also challenging. Recently, in the image domain, semantic scene graphs have been used to derive relationships among scene entities [26, 57, 34, 59, 15]. Scene graphs demonstrated to be a powerful abstract representation for scene understanding. Being compact and explicit, they are beneficial for complex tasks such as image captioning [58, 20], generation [19], manipulation [7] or visual questioning and answering [49]. For this reason, recent works have explored scene graph prediction from entire 3D scans in an offline manner [54, 1]. Furthermore, building up semantic graph maps *online* is a major challenge, requiring not only to efficiently detect semantic instances in the scene but also to robustly estimate predicates between them, while dealing with partial and incomplete 3D geometry.

In this work, we propose a real-time method to incrementally build, in parallel to 3D mapping, a globally consistent semantic scene graph, as shown in Fig. 1. Our approach relies on a geometric segmentation method [47] and a novel inductive graph network, which handles missing edges and nodes in partial 3D point clouds. Our scene nodes are geometric segments of primitive shapes. Their 3D features are propagated in a graph network that aggregates features of neighborhood segments. Our method predicts scene semantics and identifies object instances by learning relationships among clusters of over-segmented regions. Towards this end, we propose to learn additional relationships, referred to as `same part` in an end-to-end manner.

The main contributions of this work can be summarized as follows: (1) We propose the first online 3D scene graph prediction, *i.e*. incrementally fusing predictions from currently observed sub-maps into a globally consistent semantic graph model. (2) Due to a new relationship type, nodes are merged into 3D instances, resembling panoptic segmentation.(3) We introduce a novel attention method that can handle partial and incomplete 3D data, as well as highly dynamic edges, which is required for incremental scene graph prediction. Our experiments show that we outperform 3D scene graph prediction and achieve on par performance on 3D semantic and instance segmentation benchmarks while running in 35Hz.

2. Related Work

2.1. Semantic SLAM

Several 3D scene understanding methods leverage deep learning to perform either semantic segmentation [5, 10, 41, 39, 18], or instance segmentation/object detection [17, 37, 24, 9] from the complete 3D volume or point cloud of the scene. Conversely, incremental semantic SLAM approaches do not assume a full 3D scan to be available, instead directly operate on the incoming frames of RGB(-D) sequences [29, 50, 42]. Such methods simultaneously carry out a 3D reconstruction of the scene, while extracting the corresponding semantics of the currently observed surface. To this end, some incremental methods transfer image predictions from a convolutional neural network (CNN) to 3D, passing the data from the image to the 3D reconstruction [29]. [46] propose a monocular approach that constructs the 3D geometry from a depth prediction, rather than a depth image. These incremental approaches often require a sophisticated fusion and/or a regularization method to deal with multiple, potentially contradictory, predictions, and to handle spatial and temporal consistency [29, 31, 46]. Other approaches fuse the 2D image and 3D reconstruction [60]. These semantic SLAM methods such as SemanticFusion [29], ProgressiveFusion [36] or FusionAware [60] are able to reconstruct 3D semantic scene maps in real-time, but are not able to differentiate between individual object instances. Object-level SLAM approaches focus on object instances while sometimes requiring prior knowledge of the scene such as an object database or semantic class annotations [44, 48, 28, 11, 14].

Segmentation techniques are often used to reduce data complexity and meet the required runtime on limited resources. Several methods [48, 25, 30, 55, 14] incorporate the efficient incremental segmentation method proposed in [47] to perform online scene understanding.

Semantic SLAM methods achieve great performance in computing a semantic or object-level representation but neither focus on semantic scene graphs nor on semantic relationships between object instances.

2.2. Scene Graphs for Images and 3D Data

Graph Neural Networks (GNNs) have recently emerged as a popular inference tool for many challenging tasks [51, 40, 27, 52, 3, 45]. In particular, GNNs have been proposed to infer scene graphs from images [59, 40], where scene entities are the nodes of the graph, *e.g*. object instances. Scene graph prediction goes beyond instance segmentation by adding relationships between instances. Although scene graphs are adapted from computer graphics, their semantic extension has become an important research area in computer vision. Since the introduction of a large scale 2D scene graph dataset [23], several graph prediction methods have been proposed focusing on message passing with recurrent neural networks [57], iterative statistical optimization [6] or methods to handle limited data [2, 8]. Furthermore, recent datasets with 3D semantic scene graph annotations have been proposed [12, 1, 54], alongside with 3D graph estimation methods. [54] predict semantic scene graphs from a ground truth class-agnostic segmentation of the 3D scene. [12] use an object detector on a sequence of images to construct 3D quadrics – their object representation of choice. The geometric and visual features are then processed with a recurrent neural network. [1] use mask predictions and a multi-view regularization technique on sampled images to compute relationships derived from detected object instances. They construct a 3D scene graph of a building that includes object semantics, rooms and cameras, as well as the relationships between these entities. [43]

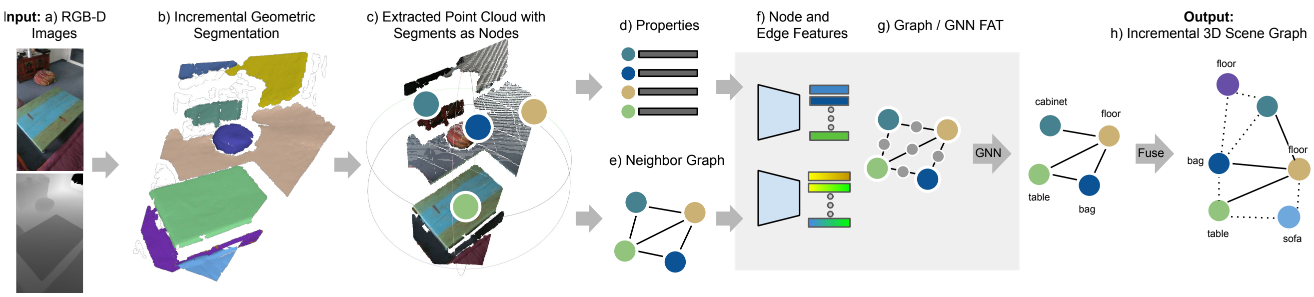

Figure 2. Overview of the proposed SceneGraphFusion framework. Our method takes a stream of RGB-D images a) as input to create an incremental geometric segmentation b). Then, the properties of each segment and a neighbor graph between segments are constructed. The properties d) and neighbor graph e) of the segments that have been updated in the current frame c) are used as the inputs to compute node and edge features f) and to predict a 3D scene graph g). Finally, the predictions are h) fused back into a globally consistent 3D graph.

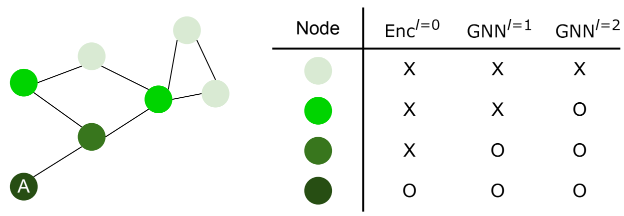

Figure 3. A representation of our efficient graph update strategy. Given a network with a basic encoder (Enc$^{l=0}$) and two message passing layers (GNN$^{l=1}$, GNN$^{l=2}$), by storing different layers of features separately. When node A is updated, we can reuse the lower-layer features from other nodes without recomputing them. We visualize the operations needed at each node with color.

extended this model to include dynamic scene entities, e.g. humans. Nevertheless, all of these methods work offline and expect the reconstructed 3D scene as an input. Similarly to [54], we predict graphs of semantic nature, but in contrast to [54], our graph prediction does not require any prior scene knowledge and is able to segment instances and their semantic information, as well as their relationships, in real-time, while the scene is being reconstructed.

3. Incremental 3D Scene Graph Framework

Fig. 2 illustrates the pipeline of our SceneGraphFusion framework. Our system consists of two separate cores: a reconstruction and segmentation pipeline adapted from [47] (Sec. 3.1), and a scene graph prediction network (SPN) (Sec. 4). Our system takes a sequence of RGB-D frames with associated poses as input to reconstruct a segmented map of the scene, while estimating a neighbor graph and properties of each segment. Then, a subset of the neighbor graph and the properties of the segments that have been recently observed are fed into our graph network to predict node and edge semantics. Finally, the predictions are fused into the globally consistent 3D scene graph. To maintain real-time performance of our system, we separate the scene graph prediction process into a different thread. The 3D scene graph is asynchronously predicted and fused from the reconstruction pipeline. Our semantic scene graph $\mathcal{G}$ consists of a set of tuples $(\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V}$ and edges $\mathcal{E}$. Nodes represent segments with their object categories, and edges represent the semantic relationships (predicates) between nodes, such as `standing on` and `attached to`.

3.1. Scene Reconstruction with Property Building

An incremental and computationally efficient method of estimating instances is required to enable construction of a scene graph in real-time. We use the incremental geometrical segmentation method in [47] to build a globally consistent segmentation map, and incorporate it with our online property update and neighbor graph building.

Geometric Segmentation and Reconstruction. Given input RGB-D frames and associated poses, the incremental segmentation algorithm generates a global 3D segmentation map, shown in Fig. 2b, by performing incremental segmentation on top of a dense reconstruction algorithm. The 3D segmentation map consists of a set of segments $\mathbf{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_n\}$. Each segment stores a set of 3D points $\mathbf{P}_i$ where each point has a 3D coordinate a normal and a color. Our map is updated at every new frame, by adding new segments and merging or removing old ones.

Segment Properties. In addition to segment reconstruction, we compute segment properties (see Fig. 2d) to describe a segment shape, *i.e.* centroid $\overline{\mathbf{p}}_i \in \mathbb{R}^3$, standard deviation of the position of points $\boldsymbol{\sigma}_i \in \mathbb{R}^3$, size of the axis-aligned bounding box $\mathbf{b}_i = (b_x, b_y, b_z) \in \mathbb{R}^3$, maximum length $l_i = \max(b_x, b_y, b_z) \in \mathbb{R}$ and bounding box volume $\nu_i = b_x \cdot b_y \cdot b_z \in \mathbb{R}$. Reconstructing the segments in this incremental manner allows us to update the properties of each node efficiently. These properties are updated by checking every modification of the points in the segment.

Neighbor Graph. Additionally, we construct a neighbor graph having nodes as segments and edges as the connection between the segments, as depicted in Fig. 2e. To find the adjacent segments, we compute the distances between all the combinations of the bounding box of the segments. The segment pairs where the distance is less than a certain threshold are added as edges (we use 0.5 meters as a proximity threshold in our experiments).

3.2. Prediction with Graph Structure

Next, we feed the segments properties and the neighbor graph to our graph network to predict the segment label and predicate on each segment and edge, shown in Fig. 2f-g. The detailed description of our graph network architecture can be found in Sec. 4. Since our segment reconstruction process is incremental, only those segments that are currently observed in the input frame are updated. Therefore, we only feed a subset of the segments and the neighbor graph which consists of the segments that have been updated in recent frames, this improving scalability and efficiency. To identify the newly updated segments, we store the segment size and timestamp whenever the segment is fed into the network. If the segment size changes more than 10%, or the segment has not been updated for 60 frames, they are flagged and fed into the network. Segments are continuously observed and outdated segments and their neighbors are extracted and processed with our graph neural network. For the sake of efficiency we store all features computed from our SPN in our neighbor graph. According to the message passing process in GNN [13], when a lower-layer feature of a node is updated, only the higher-layer features of this node, its direct neighbors, and their edge features are affected. This allows us to re-use previously computed features, as shown in Fig. 3, and greatly improve prediction efficiency and scalability.

3.3. Temporal Scene Graph Fusion

Finally, the predicted semantics of nodes and edges in the neighbor graph are fused into a globally consistent semantic scene graph, depicted in Fig. 2h. Due to the incremental nature of our method, as described in Sec. 3.2, the semantics of each segment and edge are predicted multiple times, resulting in potentially contradictory outcomes. To handle this, we apply a running average approach [4] to fuse the predictions of the same segment or edge. For each segment and edge in our neighbor graph, we store a weight w and a probability μ for each class or predicate prediction. Given a new prediction with probability μ^t at time t, we update the previously stored weight w^{t-1} and probability μ^{t-1} as

$$\mu^t = \frac{\mu^t \cdot w^t + \mu^{t-1} \cdot w^{t-1}}{w^t + w^{t-1}}, \tag{1}$$

$$w^t = \min\left(w_{max}, w^t + w^{t-1}\right), \tag{2}$$

where $w_{max} = 100$ is the maximum weight value. Importantly, since our framework predicts semantics at segment level, we are able to store and preserve the whole label probability distribution using a much smaller memory footprint compared to point-level methods [29].

4. Scene Graph Prediction

The use of segments obtained by the geometric segmentation method requires the design of a robust feature, since the shape of each segment is usually incomplete and relatively simple, and changes overtime during reconstruction. The feature of each segment can be enhanced with neighbor information by using a GNN. However, the number of neighbors of each segment changes over time, posing a serious challenge for the training process.

Dealing with dynamic nodes and edges in a GNN is known as inductive learning. Existing methods focus mainly on how to spread attention across all the neighbors [51, 52], or estimate the attention between nodes [3]. However, in either case, a missing edge still affects all the aggregated messages. To deal with this problem, we propose a novel feature-wise-attention (FAT), that re-weights individual latent features at each target node. By applying a max function on this re-weighted embedding, this strategy yields aggregated features that are less affected by missing neighboring points.

4.1. Network Architecture

The network architecture of our framework is shown in Fig. 2f-g (grey box). Our architecture is inspired from [54], with modification of some major components. Given a) a set of segments, b) the properties of each segment, and c) a neighbor graph, our network outputs a semantic scene graph by predicting class and predicate for each segment and edge respectively.

Node Feature. The point cloud $\mathbf{P}_i$ of each segment $\mathbf{s}_i$ is encoded with a PointNet [38] $f_p(\cdot)$ into a latent feature that represents the primitive shape of each segment. We concatenate the spatial invariant properties described in Sec. 3.1, *i.e.* standard deviation $\boldsymbol{\sigma}_i$, log of bounding box size $\mathbf{b}_i$, length l_i, and volume ν_i, with $f_p(\mathbf{P}_i)$ to handle the scale insensitive limitation caused by normalization of the input points on the unit sphere such that

$$\mathbf{v}_i = [f_p\left(\mathbf{P}_i\right), \boldsymbol{\sigma}_i, \ln\left(\mathbf{b}_i\right), \ln\left(\nu_i\right) \ln\left(l_i\right)], \tag{3}$$

where $[\cdot]$ denotes a concatenation function.

Edge Feature. The visual features of the edges are computed with the properties of the connected segments. Given an edge between a source node i and a target node j where $j \neq i$, the edge visual feature $\mathbf{e}_{ij}$ is computed such that

$$\mathbf{r}_{ij} = [\overline{\mathbf{p}}_i - \overline{\mathbf{p}}_j, \boldsymbol{\sigma}_i - \boldsymbol{\sigma}_j, \mathbf{b}_i - \mathbf{b}_j, \ln\left(\frac{l_i}{l_j}\right), \ln\left(\frac{\nu_i}{\nu_j}\right)], \quad (4)$$

$$\mathbf{e}_{ij} = g_s\left(\mathbf{r}_{ij}\right), \quad (5)$$

where $g_s(\cdot)$ is a multi-layer perception (MLP) projecting the paired segment properties into a latent space.

GNN Feature. After the initial feature embedding on nodes and edges, we propagate the features using a GNN with 2 message passing layers to enhance the features by enclosing the neighborhood information. Our GNN updates both node and edge features in each message passing layer ℓ. In each layer, the node and $\mathbf{v}_i^{\ell}$ and edge features $\mathbf{e}_{ij}^{\ell}$ are updated as follows:

$$\mathbf{v}_i^{\ell+1} = g_v\left([\mathbf{v}_i^{\ell}, \max_{j \in \mathcal{N}(i)}\left(\text{FAN}\left(\mathbf{v}_i^{\ell}, \mathbf{e}_{ij}^{\ell}, \mathbf{v}_j^{\ell}\right)\right)]\right), \quad (6)$$

$$\mathbf{e}_{ij}^{\ell+1} = g_e\left([\mathbf{v}_i^{\ell}, \mathbf{e}_{ij}^{\ell}, \mathbf{v}_j^{\ell}]\right), \quad (7)$$

where $g_v(\cdot)$ and $g_e(\cdot)$ are MLPs, $\mathcal{N}(i)$ is the set of neighbors indices of node i, and FAN$(\cdot)$ is the proposed feature-wise attention network, which is detailed in Sec. 4.2.

Class Prediction and Losses. Finally, the node class and the edge predicate are predicted by means of two MLP classifiers. Similarly to [54], our network can be trained end-to-end with a joint cross entropy loss, for both, object $\mathcal{L}_{obj}$ and predicates $\mathcal{L}_{pred}$.

4.2. Feature-wise Attention

Our feature-wise attention (FAT) module takes as input a query $\mathbf{Q}$ of dimensions d_q and targets $\mathbf{T}$ of dimensions d_τ. It estimates a weight distribution of dimensions d_τ by using a MLP $g_a(\cdot)$ with a softmax operation to normalize and distribute the weight. Then, the attention is calculated by element-wise multiplication of the weight matrix and the target $\mathbf{T}$,

$$\text{FAT}(\mathbf{Q}, \mathbf{T}) = \text{softmax}\left(g_a(\mathbf{Q})\right) \odot \mathbf{T}, \quad (8)$$

where $\odot$ denotes element-wise multiplication.

The use of softmax across the entire target dimension d_τ gives us a single weight matrix across all feature dimensions. We employ a multi-head approach as in [51, 45] to allow a more flexible attention distribution. The input feature dimension of $\mathbf{Q}$ and $\mathbf{T}$ are divided into h heads $\mathbf{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_h]$ and $\mathbf{T} = [\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_h]$ with $\mathbf{q}_i \in \mathbb{R}^{d_q/h}$ and $\boldsymbol{\tau}_i \in \mathbb{R}^{d_\tau/h}$. For each head, the same attention function as in equation (8) is applied, then the values from each head are concatenated back to the dimensions d_τ to obtain the multi-head attention:

$$\text{MFAT}(\mathbf{Q}, \mathbf{T}) = \left[\text{FAT}(\mathbf{q}_i, \boldsymbol{\tau}_i)\right]_{i=1}^{h}. \quad (9)$$

Unlike scaled dot-product attention [51], our approach does not distribute across edges. Instead, it learns to spread the attention across the feature dimensions of each target node. Towards this end, we design a feature-wise attention network (FAN). Given a source node feature $\mathbf{v}_i$, an edge feature $\mathbf{e}_{ij}$, and a target node feature $\mathbf{v}_j$, we compute the weighted message as

$$\begin{aligned}\text{FAN}\left(\mathbf{v}_i, \mathbf{e}_{ij}, \mathbf{v}_j\right) = \\ \text{MFAT}\left(\left[\hat{g}_q\left(\mathbf{v}_i\right), \hat{g}_e\left(\mathbf{e}_{ij}\right)\right], \hat{g}_\tau\left(\mathbf{v}_j\right)\right),\end{aligned} \quad (10)$$

where $\hat{g}_q(\cdot), \hat{g}_e(\cdot), \hat{g}_\tau(\cdot)$ are single layer perceptrons to map $\mathbf{v}_i, \mathbf{e}_{ij}, \mathbf{v}_j$ to dimensions $\frac{d_q}{2}, \frac{d_q}{2}, d_\tau$ respectively.

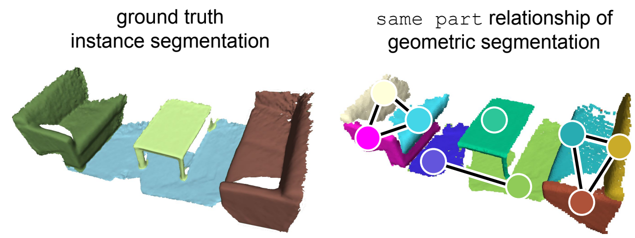

Figure 4. The `same part` relationship is generated between the segments corresponding to the same object instance.

5. Data Generation

We introduce a `same part` relationship to allow our network to cluster segments from the same object, enabling instance-level object segmentation on the over-segmented map. We generate training and testing data using the estimated segmentation and the ground truth instance annotations. Given a scene segmented by a geometrical segmentation method and its corresponding ground truth provided as object instance annotations, we find the best match between each estimated segment and the ground truth objects via nearest neighbor search. In particular, the best match is obtained by maximizing the area of intersection between the given segment and the ground truth objects. We reject matches where the area of intersection is less than 50% of the segment surface. In addition, we consider a valid match only if its corresponding segment does not cover any other ground truth objects by more than 10% of their area. In the case of multiple segments corresponding to the same object instance, we add the `same part` relationship between all of them, as shown in Fig. 4. Finally, if ground truth relationships exist on that object instance, they are inherited by all the segments.

	Relationship		Object		Predicate	
	R@50	R@100	R@5	R@10	R@3	R@5
Baseline [54]	0.39	0.45	0.66	0.77	0.62	0.88
3DSSG [54]	0.40	0.66	0.68	0.78	0.89	0.93
Ours	**0.85**	**0.87**	**0.70**	**0.80**	**0.97**	**0.99**

Table 1. Evaluation of the scene graph prediction task on 3RScan/3DSSG [54] with 160 objects and 26 predicate classes. The experiments were conducted on the complete 3D data.

6. Evaluation

In Sec. 6.1 we evaluate our scene graph prediction on 3DSSG [54]. The performance of our method is reported on full scenes given ground truth instances and second with geometric segments. We then show how relationships/graphs help with object prediction. In Sec. 6.2 we focus on the by-product of our method, panoptic segmentation by reporting segmentation scores on ScanNet [5]. Finally, in Sec. 6.3 we provide a runtime analysis of our method compared to other incremental semantic segmentation approaches.

6.1. Semantic Scene Graph Prediction

Following the evaluation scheme in [54], we separately report relationship, object, and predicate prediction accuracy with a top-n evaluation metric. Following [59], the relationship score is the multiplication of the object, subject, and predicate probability. Object and predicate metrics are calculated directly with the respective classification scores.

Ground Truth Instances. In Tbl. 1 we report the 3D scene graph prediction accuracy independently from the segmentation quality. The evaluation was conducted on the full 3D scene with the class-agnostic ground truth segmentation, as carried out in [54]. We followed the data split proposed in 3DSSG with 160 object classes and 26 different relationships. Our method outperforms [54] with a significant margin of +0.45 / +0.21 (R@50 / R@100) for relationship prediction due to small improvements in predicate and object classification. Note that our method can run offline on the pre-computed 3D data – as done here – but is also able to handle partial and incomplete shapes in an incremental online setup which is analyzed in following paragraph.

Geometric Segments. In Tbl. 2 we compare the performance of incremental ⑥-⑦ and full scene graph prediction ⑤ based on our geometric segmentation. ⑥ is slightly worse than ⑤ but generates predictions on the fly. Our proposed fusion ⑦ improves the performance further. Tbl. 2 additionally shows that we outperform 3DSSG [54] with a small margin without any attention method and with a large margin when using our proposed feature-wise attention, FAT ⑤. FAT ⑤ also outperforms other attention mechanisms GAT [52] ③ and SDPA [51] ④ for 3D semantic scene graph prediction. The input of the methods is either the full 3D scene ①-⑥, processed offline (f) or a stream of RGB-D images processed incrementally (i), ⑥, ⑦. For these experiments, we first acquired the geometric segmentation [46] from the RGB-D sequences of 3RScan [53]. The final training data was generated with the pipeline described in Sec. 5. We trained the networks with 20 NYUv2 [32] object classes used on the ScanNet [5] benchmark. Furthermore, only support predicates are used and relationships with too few occurrences are ignored. This leads to 8 predicates, including the `same part` relationship which we added in the data generation process. More details on the training setup and chosen hyper-parameters used in this experiment can be found in the supplementary material. A qualitative result of our graph prediction is shown in Fig. 5, more examples can also be found in the supplementary.

Method (Attention)		Relationship		Object		Predicate	
		R@1	R@3	R@1	R@3	R@1	R@2
① 3DSSG (none)	(f)	0.38	0.59	0.61	0.85	0.83	0.98
② Ours (none)	(f)	0.41	0.62	0.62	0.88	0.84	0.98
③ Ours (GAT)	(f)	0.12	0.22	0.25	0.64	0.85	0.98
④ Ours (SDPA)	(f)	0.39	0.62	0.62	0.87	0.85	0.98
⑤ Ours (FAT)	(f)	**0.55**	**0.78**	0.75	0.93	**0.86**	0.98
⑥ Ours (FAT)	(i)	0.51	0.67	0.78	**0.94**	0.77	0.98
⑦ Ours Fusion (FAT)	(i)	0.52	0.70	**0.79**	**0.94**	0.78	0.98

Table 2. Evaluation of the semantic scene graph prediction on geometric segments of 3RScan/3DSSG [54] with 20 objects and 8 predicate classes. (f) indicates a prediction on the full 3D scene while (i) is the incremental result from the RGB-D sequence.

	Relationship		Object		Predicate	
	R@1	R@3	R@1	R@3	R@1	R@2
Ours without $\mathcal{L}_{pred}$	0.26	0.36	0.62	0.87	0.59	0.75
Ours with $\mathcal{L}_{pred}$	**0.55**	**0.78**	**0.75**	**0.93**	**0.86**	**0.98**

Table 3. Ablation Study: Comparison of training with and without predicate loss $\mathcal{L}_{pred}$ on 3RScan/3DSSG [54] with 20 object and 8 predicate classes. Note that the comparison is based on graphs computed from the full 3D scene (f).

Predicate Influence on Object Classification. To verify if learning inter-instance relationship improves object classification, we train our network without the predicate loss. Tbl. 3 shows that object classification indeed benefits from joint relationship prediction.

6.2. 3D Panoptic/Semantic Segmentation

To evaluate the quality of the semantic/panoptic segmentation of our method, we trained the network with ScanNet [5]. We follow the ScanNet benchmark and evaluate with the IoU metric. Since InSeg [47] reconstructs and segments the scene with a different reconstruction algorithm and excludes small and unstable geometric segments, some

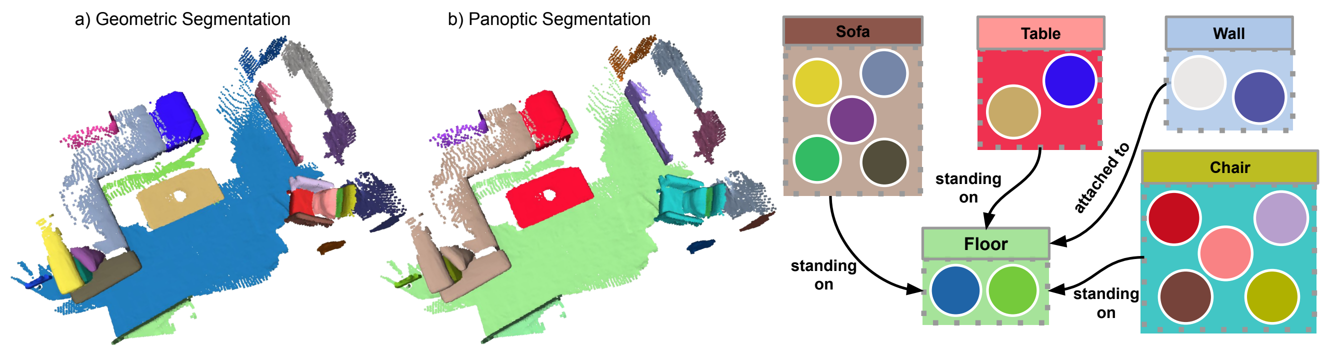

Figure 5. Qualitative evaluation of our incremental graph prediction. Node (circle) colors represent geometric segments shown on the left, the corresponding predicted semantics (panoptic segmentation) is visualized in the center, corresponding to the boxes in the right. For visualization purposes, we only show the biggest segments and filter out small ones.

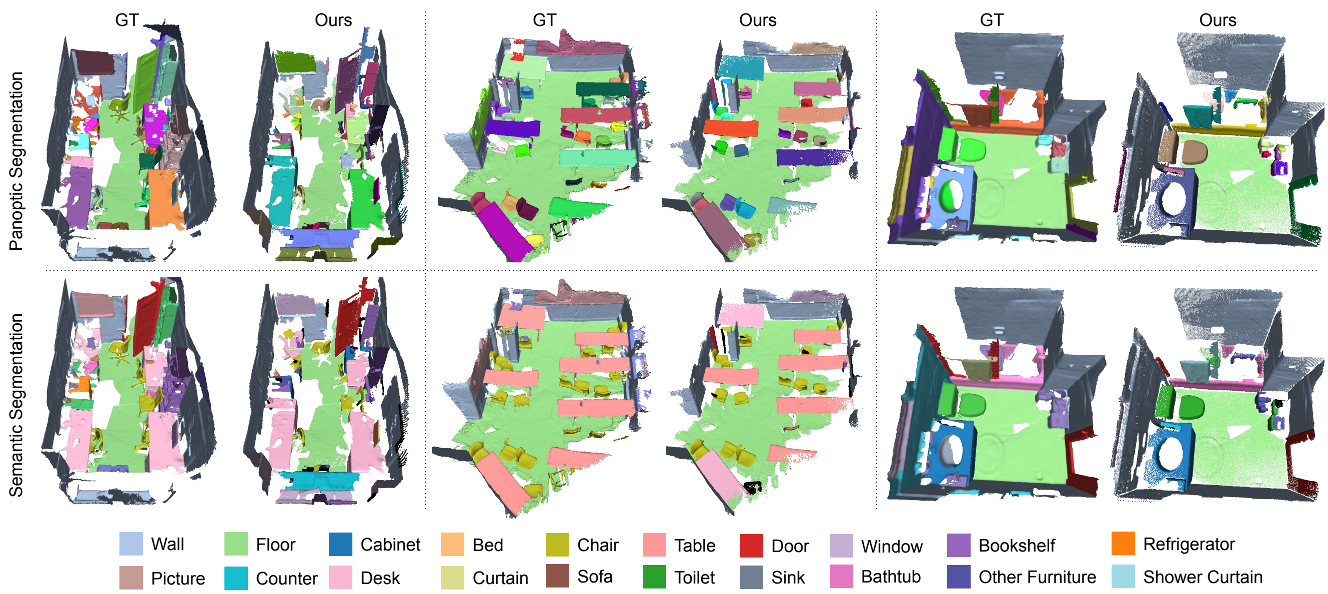

Figure 6. Qualitative semantic and panoptic segmentation results of SceneGraphFusion. Note that for 3D panoptic segmentation (top row), random colors are used for object instances, while walls are ▢ and the floor is ▢.

points might be missing in our 3D map. When evaluating, we address this issue by either a) mapping the points in our reconstruction to the nearest neighbor (NN) of the ScanNet ground truth 3D model or b) ignoring points where no corresponding 3D geometry was reconstructed.

3D Semantic Segmentation. In Tbl. 5, we compare our method against other incremental semantic segmentation methods, specifically SemanticFusion [29], ProgressiveFusion [36] and FusionAware [60] using the mean average precision (mAP). Our method has the second best mAP while running at 35Hz on a CPU, as detailed in Sec. 6.3 and the supplementary material. Qualitative results of our semantic segmentation are shown in the bottom row of Fig. 6.

3D Panoptic Segmentation. To evaluate panoptic segmentation we use the metrics proposed in [22], namely panoptic quality (PQ), segmentation quality (SQ), and recognition quality (RQ). In Tbl. 4 we compare our method against PanopticFusion [31]. Due to the missing scene geometry on which our approach relies, PanopticFusion outperforms our method with respect to the computed RQ. Nevertheless, SQ and PQ are on par or slightly worse. A comparison of only valid scene regions – by skipping unreconstructed parts – often results in a better performance. We provide an ablation study in Tbl. 6 to validate the effectiveness of the `same part` relationship and our proposed fusion mechanism. Finally, the qualitative results of our panoptic segmentation are shown in the top row of Fig. 6.

	Metric	All	Things	Stuff
PanopticFusion [31]	PQ	**33.5**	**30.8**	**58.4**
Ours (NN mapping)	PQ	31.5	30.2	43.4
Ours (skipped missing)	PQ	36.3	34.7	51.0
PanopticFusion [31]	SQ	**73.0**	**73.3**	70.7
Ours (NN mapping)	SQ	72.9	73.0	**72.6**
Ours (skipped missing)	SQ	76.1	75.9	77.9
PanopticFusion [31]	RQ	**45.3**	**41.3**	**80.9**
Ours (NN mapping)	RQ	42.2	40.3	59.3
Ours (skipped missing)	RQ	46.8	44.8	64.7

Table 4. 3D panoptic segmentation results on the ScanNet v2 open test set. We report the numbers of PanopticFusion [31] and our fusion method either by a NN mapping or by skipping those missing regions. *Our (NN mapping)* outperforms PanopticFusion in 7 classes, more information can be found in the supplementary material. Note that *Our (skipped missing)* is not considered when highlighting the best score since its not directly comparable.

	Hardware	Runtime [Hz]	mAP
SemanticFusion [29]	GPU + CPU	25	51.8
ProgressiveFusion [36]	GPU + CPU	10 – 15	56.6
Fusionaware [60]	-	10	**76.4**
Ours	CPU	**35**	63.7

Table 5. Comparison of incremental semantic segmentation methods on the open test set of ScanNet [5]. Runtime have been taken from the respective papers with potentially different hardware setups, therefore not directly comparable.

	PQ	SQ	RQ
① Ours without Fusion	35.4	76.3	35.4
② Ours without `same part`	10.9	59.1	16.0
③ Ours	36.3	76.1	46.8

Table 6. Ablation study: Analysis of the effect of our fusion mechanism and the `same part` relationship on the panoptic segmentation task evaluated on ScanNet [5]. PQ stands for panoptic, SQ for segmentation, and RQ for recognition quality.

Robustness against Missing Information. In this experiment, we evaluate the robustness of different attention methods against noisy data in form of missing edges. We train our network without attention, using GAT [52], SDPA [51], and our proposed FAT. The experimental setup is shared with Tbl. 2. In Tbl. 7 we compare the performance of the different attention methods on the full scene (f) and all edges (@1.0, left column) and with a random edge drop of 50% (@0.5, right column), see Tbl. 7.

The reference metric is the intersection over union (IoU). Our proposed attention mechanism, FAT, consistently outperforms the other approaches in full-edge and drop-edge scenarios. For the sake of space, a more detailed per-class evaluation is available in the supplementary, interestingly showing that some classes rely on the messages from neighbors more than others such as *e.g.* bathtub, shower curtain, and windows.

	avg. IoU (@1.0)	avg. IoU (@0.5)
Ours w/o attention	33.5	29.5
Ours SDPA [51]	33.0	29.7
Ours GAT [52]	11.5	12.5
Ours FAT	**49.3**	**41.9**

Table 7. Ablation study: Segment classification of InSeg [47] on 3RScan [53] reporting avg. IoU on segment-level. The complete per-class evaluation can be found in the supplementary material.

	Segmentation	Node	Edge	GNN
Mean [ms]	28	8	17	108

Table 8. Runtime [ms] of the different components of our method.

6.3. Runtime Analysis

We measured the runtime of our system on the ScanNet sequence `scene0645_01`. Our machine is equipped with an Intel Core i7-8700 CPU 3.2GHz CPU with 12 threads. Notably, our method only uses 2 threads: one for the scene reconstruction and the other one for 3D scene graph prediction. The scene reconstruction requires 28 ms on average while the graph prediction sums up to 133ms running the GNN and fusing the results.

7. Conclusion

In this work, we presented SceneGraphFusion, a 3D scene graph method that incrementally fuses partial graph predictions from a geometric segmentation into a globally consistent semantic map. Our network outperforms other 3D scene graph prediction methods; FAT works better than any other attention mechanism in handling missing graph information and the semantic/panoptic segmentation – the by-product of our method – achieves performance on par with other incremental methods while running in 35Hz. Due to this efficiency, incremental semantic scene graphs could be beneficial in future work, when retrieving camera poses or detecting loop-closures in a SLAM framework.

Acknowledgment

The authors thank Anees B. Kazi and Georam Vivar for fruitfull discussions. This work is supported by the *German Research Foundation* (DFG, project number 407378162) and the Bavarian State Ministry of Education, Science and the Arts in the framework of the Centre Digitisation Bavaria (ZD.B).

References

[1] Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3D SceneGgraph: A Structure for Unified Semantics, 3D Space, and Camera. In *IEEE International Conference Computer Vision (ICCV)*, 2019. 2

[2] Vincent S. Chen, Paroma Varma, Ranjay Krishna, Michael Bernstein, Christopher Re, and Li Fei-Fei. Scene Graph Prediction With Limited Labels. In *IEEE International Conference Computer Vision (ICCV)*, October 2019. 2

[3] Luca Cosmo, Anees Kazi, Seyed-Ahmad Ahmadi, Nassir Navab, and Michael Bronstein. Latent-Graph Learning for Disease Prediction. In *International Conference Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer, 2020. 2, 4

[4] Brian Curless and Marc Levoy. A Volumetric Method for Building Complex Models from Range Images. In *Proceedings Conference on Computer Graphics and Interactive Techniques*, 1996. 4

[5] Angela Dai, Angel Xuan Chang, Manolis Savva, Maciej Halber, Tom Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 6, 8

[6] Bo Dai, Yuqi Zhang, and D. Lin. Detecting Visual Relationships with Deep Relational Networks. *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[7] Helisa Dhamo, Azade Farshad, Iro Laina, Nassir Navab, Gregory D Hager, Federico Tombari, and Christian Rupprecht. Semantic Image Manipulation Using Scene Graphs. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[8] Apoorva Dornadula, Austin Narcomey, Ranjay Krishna, Michael Bernstein, and Fei-Fei Li. Visual Relationships as Functions: Enabling Few-Shot Scene Graph Prediction. In *IEEE International Conference Computer Vision Workshops (ICCVW)*, 2019. 2

[9] Francis Engelmann, Martin Bokeloh, Alireza Fathi, Bastian Leibe, and Matthias Nießner. 3D-MPA: Multi Proposal Aggregation for 3D Semantic Instance Segmentation. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2

[10] Francis Engelmann, Theodora Kontogianni, Alexander Hermans, and Bastian Leibe. Exploring Spatial Context for 3D Semantic Segmentation of Point Clouds. In *IEEE International Conference Computer Vision Workshops (ICCVW)*, 2017. 1, 2

[11] Fadri Furrer, Tonci Novkovic, Marius Fehr, Abel Gawel, Margarita Grinvald, Torsten Sattler, Roland Siegwart, and Juan I. Nieto. Incremental Object Database: Building 3D Models from Multiple Partial Observations. In *IEEE Conference Intelligent Robots and Systems (IROS)*, 2018. 2

[12] Paul Gay, James Stuart, and Alessio Del Bue. Visual Graphs from Motion (VGfM): Scene Understanding with Object Geometry Reasoning. In *Asian Conference on Computer Vision (ACCV)*. Springer, 2018. 2

[13] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural Message Passing for Quantum Chemistry. In *International Conference Machine Learning (ICML)*, 2017. 4

[14] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto. Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery. *IEEE Robotics and Automation Letters (RAL)*, 2019. 2

[15] Jiuxiang Gu, Handong Zhao, Zhe Lin, Sheng Li, Jianfei Cai, and Mingyang Ling. Scene Graph Generation with External Knowledge and Image Reconstruction. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[16] Lei Han, Tian Zheng, Lan Xu, and Lu Fang. OccuSeg: Occupancy-aware 3D Instance Segmentation. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[17] Ji Hou, Angela Dai, and Matthias Nießner. 3D-SIS: 3D Semantic Instance Segmentation of RGB-D Scans. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2

[18] Jingwei Huang, Haotian Zhang, Li Yi, A. Thomas Funkhouser, Matthias Niener, and J. Leonidas Guibas. TextureNet: Consistent Local Parametrizations for Learning from High-Resolution Signals on Meshes. *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2

[19] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image Generation from Scene Graphs. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[20] Andrej Karpathy and Li Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[21] Maik Keller, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb. Real-Time 3D Reconstruction in Dynamic Scenes Using Point-Based Fusion. In *International Conference on 3D Vision (3DV)*, 2013. 1

[22] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollar. Panoptic Segmentation. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, June 2019. 7

[23] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *International Journal of Computer Vision (IJCV)*, 2017. 2

[24] J. Lahoud, B. Ghanem, M. R. Oswald, and M. Pollefeys. 3D Instance Segmentation via Multi-Task Metric Learning. In *IEEE International Conference Computer Vision (ICCV)*, 2019. 1, 2

[25] Chi Li, Han Xiao, Keisuke Tateno, Federico Tombari, Nassir Navab, and Gregory D. Hager. Incremental Scene Understanding on Dense SLAM. In *IEEE Conference Intelligent Robots and Systems (IROS)*, 2016. 2

[26] Yikang Li, Wanli Ouyang, Bolei Zhou, Kun Wang, and Xiaogang Wang. Scene Graph Generation from Objects, Phrases

and Region Captions. In *IEEE International Conference Computer Vision (ICCV)*, 2017. 1

[27] Franco Manessi, Alessandro Rozza, and Mario Manzo. Dynamic Graph Convolutional Networks. *Pattern Recognition*, 2020. 2

[28] John McCormac, Ronald Clark, Michael Bloesch, Stefan Leutenegger, and Andrew Davison. Fusion++: Volumetric Object-Level SLAM. In *International Conference on 3D Vision (3DV)*, 2018. 2

[29] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. SemanticFusion: Dense 3D Semantic Mapping with Convolutional Neural Networks. In *International Conference Robotics and Automation (ICRA)*, 2017. 2, 4, 7, 8

[30] Yoshikatsu Nakajima, Keisuke Tateno, Federico Tombari, and Hideo Saito. Fast and Accurate Semantic Mapping through Geometric-based Incremental Segmentation. In *IEEE Conference Intelligent Robots and Systems (IROS)*, 2018. 2

[31] Gaku Narita, Takashi Seno, Tomoya Ishikawa, and Yohsuke Kaji. PanopticFusion: Online Volumetric Semantic Mapping at the Level of Stuff and Things. In *IEEE Conference Intelligent Robots and Systems (IROS)*, 2019. 2, 7, 8

[32] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor Segmentation and Support Inference from RGBD Images. In *Springer European Conference Computer Vision (ECCV)*, 2012. 6

[33] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. KinectFusion: Real-Time Dense Surface Mapping and Tracking. In *IEEE International Symposium on Mixed and Augmented Reality*, 2011. 1

[34] Alejandro Newell and Jia Deng. Pixels to Graphs by Associative Embedding. In *International Conference on Neural Information Processing Systems (NeurIPS)*, 2017. 1

[35] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3D Reconstruction at Scale using Voxel Hashing. In *ACM Transactions on Graphics*, 2013. 1

[36] Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Real-time Progressive 3D Semantic Segmentation for Indoor Scene. In *IEEE Winter Conference Applications Computer Vision (WACV)*, 2019. 2, 7, 8

[37] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep Hough Voting for 3D Object Detection in Point Clouds. In *IEEE International Conference Computer Vision (ICCV)*, 2019. 2

[38] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2017. 4

[39] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *International Conference on Neural Information Processing Systems (NeurIPS)*, 2017. 1, 2

[40] Mengshi Qi, Weijian Li, Zhengyuan Yang, Yunhong Wang, and Jiebo Luo. Attentive Relational Networks for Mapping Images to Scene Graphs. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[41] Dario Rethage, Johanna Wald, Jürgen Sturm, Nassir Navab, and Federico Tombari. Fully-Convolutional Point Networks for Large-Scale Point Clouds. In *Springer European Conference Computer Vision (ECCV)*, 2018. 1, 2

[42] A. Rosinol, M. Abate, Y. Chang, and L. Carlone. Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping. In *International Conference Robotics and Automation (ICRA)*, 2020. 2

[43] Antoni Rosinol, Arjun Gupta, Marcus Abate, Jingnan Shi, and Luca Carlone. 3D Dynamic Scene Graphs: Actionable Spatial Perception with Places, Objects, and Humans. *Computing Research Repository*, 2020. 2

[44] Renato F. Salas-Moreno, Richard A. Newcombe, Hauke Strasdat, Paul H. J. Kelly, and Andrew J. Davison. SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2013. 2

[45] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning Feature Matching with Graph Neural Networks. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 5

[46] K. Tateno, F. Tombari, I. Laina, and N. Navab. CNN-SLAM: Real-Time Dense Monocular SLAM with Learned Depth Prediction. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 6

[47] Keisuke Tateno, Federico Tombari, and Nassir Navab. Real-Time and Scalable Incremental Segmentation on Dense SLAM. In *IEEE Conference Intelligent Robots and Systems (IROS)*, 2015. 2, 3, 6, 8

[48] Keisuke Tateno, Federico Tombari, and Nassir Navab. When 2.5D is not enough: Simultaneous Reconstruction, Segmentation and Recognition on dense SLAM. In *International Conference Robotics and Automation (ICRA)*, 2016. 2

[49] D. Teney, L. Liu, and A. Van Den Hengel. Graph-Structured Representations for Visual Question Answering. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[50] Julien P. C. Valentin, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nießner, Antonio Criminisi, Shahram Izadi, and Philip H. S. Torr. SemanticPaint: Interactive 3D Labeling and Learning at your Fingertips. *ACM Transactions on Graphics*, 2015. 2

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *International Conference on Neural Information Processing Systems (NeurIPS)*, 2017. 2, 4, 5, 6, 8

[52] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *International Conference on Learning Representations (ICLR)*, 2018. 2, 4, 6, 8

[53] Johanna Wald, Armen Avetisyan, Nassir Navab, Federico Tombari, and Matthias Nießner. RIO: 3D Object Instance

Re-Localization in Changing Indoor Environments. In *IEEE International Conference Computer Vision (ICCV)*, 2019. 1, 6, 8

[54] Johanna Wald, Helisa Dhamo, Nassir Navab, and Federico Tombari. Learning 3D Semantic Scene Graphs from 3D Indoor Reconstructions. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3, 4, 5, 6

[55] Johanna Wald, Keisuke Tateno, Jürgen Sturm, Nassir Navab, and Federico Tombari. Real-Time Fully Incremental Scene Understanding on Mobile Platforms. *IEEE Robotics and Automation Letters (RAL)*, 2018. 2

[56] Thomas Whelan, Stefan Leutenegger, Renato Salas Moreno, Ben Glocker, and Andrew Davison. ElasticFusion: Dense SLAM Without A Pose Graph. In *Robotics: Science and Systems*, 2015. 1

[57] Danfei Xu, Yuke Zhu, Christopher Choy, and Li Fei-Fei. Scene Graph Generation by Iterative Message Passing. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2

[58] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *International Conference Machine Learning (ICML)*, 2015. 2

[59] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph R-CNN for Scene Graph Generation. In *Springer European Conference Computer Vision (ECCV)*, 2018. 1, 2, 6

[60] Jiazhao Zhang, Chenyang Zhu, Lintao Zheng, and Kai Xu. Fusion-Aware Point Convolution for Online Semantic 3D Scene Segmentation. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 7, 8