SAM-guided Graph Cut for 3D Instance Segmentation

Haoyu Guo[1*] He Zhu[2*] Sida Peng[1] Yuang Wang[1]
Yujun Shen[3] Ruizhen Hu[4†] Xiaowei Zhou[1†]

[1]Zhejiang University [2]Beijing Normal Univeristy [3]Ant Group [4]Shenzhen Univeristy

Abstract. This paper addresses the challenge of 3D instance segmentation by simultaneously leveraging 3D geometric and multi-view image information. Many previous works have applied deep learning techniques to 3D point clouds for instance segmentation. However, these methods often failed to generalize to various types of scenes due to the scarcity and low-diversity of labeled 3D point cloud data. Some recent works have attempted to lift 2D instance segmentations to 3D within a bottom-up framework. The inconsistency in 2D instance segmentations among views can substantially degrade the performance of 3D segmentation. In this work, we introduce a novel 3D-to-2D query framework to effectively exploit 2D segmentation models for 3D instance segmentation. Specifically, we pre-segment the scene into several superpoints in 3D, and formulate the task into a graph cut problem. The superpoint graph is constructed based on 2D segmentation models, enabling great segmentation performance on various types of scenes. We employ a GNN to further improve the robustness, which can be trained using pseudo 3D labels generated from 2D segmentation models. Experimental results on the ScanNet200, ScanNet++ and KITTI-360 datasets demonstrate that our method achieves state-of-the-art segmentation performance. Code will be made publicly available for reproducibility.

Keywords: 3D Instance Segmentation · 3D Scene Understanding · Graph Neural Network

1 Introduction

Instance segmentation of 3D scenes is a cornerstone of many applications, such as augmented and virtual reality, robot navigation, and autonomous driving. One typical pipeline for 3D scene segmentation is using deep neural networks [29, 37, 38, 52] to process the point cloud of the target scene to predict segmentation results. These methods [11, 12, 19, 23, 28, 46, 54, 56, 62] generally require annotated point clouds for training. However, annotating point clouds is costly, and thus there is a lack of datasets with the large scale and diversity similar to 2D image

* Equal contribution
† Corresponding authors

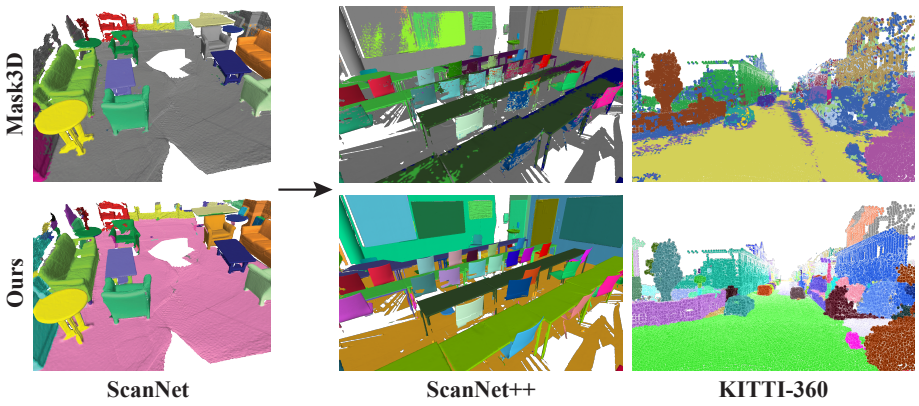

Fig. 1: Thanks to the inductive bias introduced by SAM-annotated superpoint graph, our method achieves good segmentation performance and generalization capabilities. After training solely on ScanNet200, our model can effectively generalize to data collected with different devices (ScanNet++) and even to entirely different types of scenes (KITTI-360).

datasets. As a result, these methods are often limited to specific types of scenes and struggle to generalize to in-the-wild scenes.

Compared to point clouds, the acquisition and annotation of images are much less costly. In recent years, with the emergence of large-scale labeled 2D datasets and improvement in model architecture and capacity, state-of-the-art 2D instance segmentation models [4, 24, 25, 39] with strong generalization capabilities have been developed. Consequently, a natural approach of 3D segmentation is to lift multi-view 2D segmentations to 3D, leveraging them to achieve segmentation in arbitrary 3D scenes.

Recently, some 2D-to-3D lifting methods [1, 26, 48, 57, 58, 61, 63] have used a bottom-up framework, which first runs 2D segmentation on each view to obtain several masks, and then attempts to establish correspondences among masks across different views. These masks are then merged in 3D to obtain the 3D segmentation results. However, such a bottom-up approach has a significant issue: 2D segmentation masks from different views may be inconsistent. For example, some instances may be segmented in some views but missing in others, thus severely degrading the performance of 3D segmentation.

In this paper, we propose a novel 3D segmentation approach based on a 3D-to-2D query framework, which effectively utilizes 2D segmentation models. Unlike previous methods that generate multiple masks from multi-view images first, our approach begins with a pre-segmentation of the 3D scene into several superpoints. Then, we construct a superpoint graph of the target scene and transform the problem into graph cut. The edge weights of the graph are obtained by projecting graph nodes onto multiple views, using the prompt mechanism of SAM [25] to predict multi-view masks, and calculating the intersection of

corresponding masks. The node features are obtained by aggregating multi-view image features. Finally, we use a graph neural network to refine the edge affinity for the graph partition [50]. The SAM-based graph enables great segmentation performance on various types of scenes. In addition, we develop a scheme to generate pseudo 3D labels from a 2D segmentation network and design a training strategy to effectively leverages these pseudo labels, allowing us to train our model without any manual 3D annotations.

We conduct experiments on the ScanNet200, ScanNet++ and KITTI-360 datasets. The experimental results indicate that, with the guidance of SAM in the construction of the graph, our method achieves good segmentation results on various types of scenes. Moreover, the GNN module in our method has good generalization capabilities, which is able to generalize well to other datasets with significant differences after trained on one dataset with pseudo-labels.

In summary, our contributions are as follows:

- We propose a novel 3D-to-2D-query framework that leverages SAM to construct node features and edge weights of the superpoint graph, enabling great segmentation performance on various types of scenes.
- We employ GNN to improve the robustness and develop a scheme to generate pseudo 3D labels from a 2D segmentation network, enabling our model to be trained without any manual 3D annotations.
- We demonstrate state-of-the-art performance on ScanNet200 [43], ScanNet++ [64] and KITTI-360 [30] datasets.

2 Related work

3D scene segmentation. The goal of 3D scene segmentation is to group the scene point cloud into semantically meaningful regions or distinct objects. Previous works leverage large-scale 3D labeled datasets to accomplish this objective in a supervised manner. They first train a neural network to extract per-point features, and then assign one predicted label to each point based on the extracted features. [16,21,29,32,38,51,52,55,59] achieve semantic segmentation on point cloud, and [11,12,19,23,28,54,56,62] further distinguish between different objects with the same semantics, thus getting 3D instance segmentation results. Recently, Mask3D [46] leverage Transformer [53] to construct the segmentation network, attaining superior instance segmentation quality on 3D point clouds. 3D-SIS [20] performs 3D instance segmentation on RGB-D scan data, fusing image features extracted from 2D convolution networks with 3D scan geometry features, allowing accurate inference for object bounding boxes, class labels, and instance masks.

Some works exploit 2D large vision-language models to achieve open-vocabulary segmentation in 3D space. OpenScene [36] back-project per-pixel image features extracted by large vision language models from multi-view posed images to form a feature point cloud endowed with open-vocabularity abilities for various downstream scene understanding tasks. PLA [10] constructs multi-scale 3D-text pairs

and uses contrastive learning to enable the model to learn language-aware embeddings for 3D semantic and instance segmentation.

Partitioning 3D point cloud into a collection of small, geometrically homogeneous regions, coined superpoints, can yield a decent rough prediction and effectively simplify the process of scene segmentation. [49] and [22] propose to represent each 3D scene by constructing a superpoint graph, where superpoints serve as graph nodes, and then based on this graph, 3D instance segmentation is performed by learning inter-superpoint affinity and clustering superpoint nodes into 3D objects. [41] propose an efficient semantic segmentation for large-scale scenes by partitioning point clouds into a hierarchical superpoint structure, and [42] further extend it for panoptic segmentation by scalable graph clustering.

2D-to-3D lifting. Due to the lower cost of acquiring and annotating 2D images, the scale and diversity of 2D annotated datasets [5, 9, 13, 18, 27, 31], are much larger compared to 3D datasets, facilitating the emergence of many highly effective 2D segmentation methods [4, 24, 25, 39] in recent years, making the use of 2D segmentation for 3D tasks a new and promising approach. Semantic-NeRF [67], based on the NeRF framework, utilizes outputs from a 2D semantic segmentation network at each view to train a 3D semantic field. Since it can fuse information from multiple viewpoints, this method is robust to inaccuracies and noise in individual view segmentations, yielding better 3D semantic segmentation results. However, extending this approach directly to instance segmentation is challenging, which is more complex since instance IDs in multi-view image segmentation results could inconsistent, making it necessary to design an effective label matching mechanisms in order to lift multi-view 2D instance segmentation results to 3D space.

To address this issue, [48] solves a linear assignment for instance identifiers across views with machine generated semantic and instance labels as supervision, [1] proposes a scalable slow-fast clustering objective function to fuse 2D predictions into a unified 3D scene segmentation results represented with a neural field. SAD [2] uses SAM to segment both images and depth maps, combining the advantages of both for improved results. SAM3D [63] proposes a method for point cloud fusion. It segments each frame and gradually merges the segmentation results of all frames together. For scenes with known geometry, it can achieve segmentation very efficiently. SAI3D [65] partitions a 3D scene into geometric primitives, which are then progressively merged into 3D instance segmentations that are consistent with the multi-view SAM masks. MaskClustering [60] proposes a novel metric called view consensus to better exploit multi-view observation for 3D instance segmentation. Some works integrate SAM [25] into the Neural Radiance Fields (NeRF) [33] framework for 3D segmentation. For example, [3] merges SAM features into Instant NGP [34] in 3D, allowing users to segment an object from 3D space through multiple clicks. OR-NeRF [66] enables users to segment an object by clicking and then remove it from the scene.

Additionally, some methods combine segmentation and reconstruction, enabling separate reconstruction of each object. [61] propose to decompose a scene by learning an object-compositional neural radiance field, with each standalone

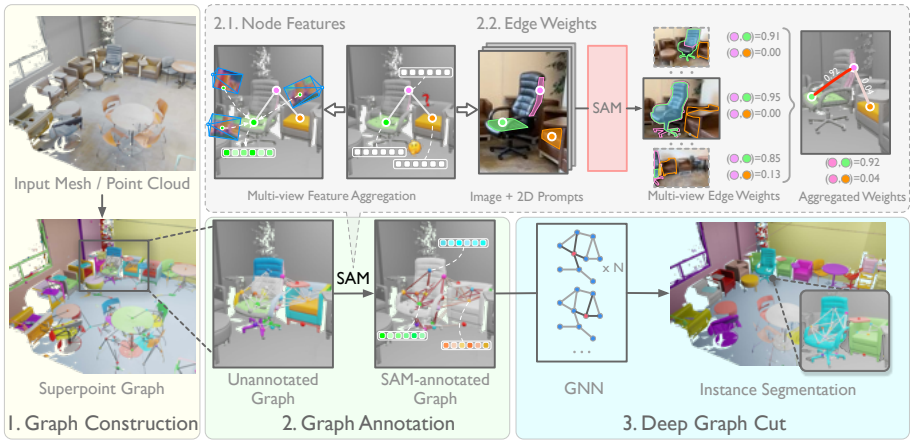

Fig. 2: Overview of our pipeline. Our proposed 3D instance segmentation pipeline consists of three main parts. **1.** We over-segment the input mesh / point cloud into superpoints and construct the structure of the superpoint graph based on adjacency (Sec. 3.1). **2.** We utilize the prompt mechanism of SAM [25] to annotate the nodes and edges of the graph (Sec. 3.2). The node features are aggregated from multi-view SAM backbone features corresponding to each superpoint. The edge weights are calculated based on the intersection ratio between the multi-view SAM masks corresponding to each pair of superpoints that constitute an edge. **3.** We use a graph neural network to further process the SAM-annotated graph and perform graph cut based on the calculated edge affinity scores to obtain the instance segmentation results (Sec. 3.3).

object separated from the scene and encoded with a learnable object activation codes, allowing more flexible downstream applications. To cope with the ambiguity of conventional volume rendering pipelines, [57,58] further utilizes the Signed Distance Function (SDF) representation to exert explicit surface constraint. [26] represents each object in the scene with a small MLP and builds an object-level dense SLAM that detects objects on-the-fly and dynamically adds them to its map.

3 Method

Given 3D geometry and calibrated multi-view images of a scene, our goal is to obtain its 3D instance segmentation. In this paper, we propose a novel segmentation framework, as illustrated in Fig. 2. We first perform over-segmentation on the 3D geometry to generate a set of superpoints, reformulating the task into a graph cut problem (Sec. 3.1). Then, Sec. 3.2 describes how to leverage SAM to construct node features and edge weights of the superpoint graph. In Sec. 3.3, we introduce a graph neural network for 3D segmentation, which is trained with pseudo labels generated by 2D segmentation predictions.

3.1 Building the superpoint graph

In an indoor or outdoor scene, not only can we easily acquire multi-view images, but we can also obtain the scene's geometry (either in point clouds or mesh form) through depth cameras/laser scanners (for indoor scenes) or LiDAR (for large-scale outdoor scenes). With the geometry of the scene available, we can proceed to pre-segment the scene based on traditional methods to obtain a set of superpoints. For mesh, we apply the method in [6], which calculates the similarity between mesh vertices based on their normal directions and then conducts the graph cut algorithm [14]. For point clouds, we employ the method in [17], which first computes a local geometric feature vector (dimensionality and verticality) for each point, then performs Potts energy segmentation [8].

We can formulate the scene's instance segmentation task as a graph cut problem by employing superpoints. Specifically, we first represent the scene as a graph $G = (V, E)$, where V denotes the set of superpoints in the scene and E denotes the adjacency relationships between these superpoints. Two superpoints are considered as adjacent if their distance is within a predefined threshold. By employing superpoints, we can simplify the segmentation task in two ways. Firstly, the number of superpoints is substantially lower than the number of points in the original point cloud. Secondly, superpoints serve as a 3D proxy, enabling us to utilize multi-view image information and SAM's prompt mechanism to determine the connection between regions in 3D space.

3.2 Constructing edge weights and node features

To accomplish segmentation of the 3D scene, our primary task is to determine whether two superpoints on each edge should be merged. To this end, we leverage multi-view image information and employ SAM to annotate the graph so that we can apply graph cut for segmentation. Specifically, we utilize the prompt mechanism of SAM to annotate the edges and use SAM encoder features to annotate the nodes.

Prompt mechanism of SAM. Unlike previous 2D instance segmentation methods that take an image as input and output its segmentation map, SAM (Segment Anything Model) [25] operates by taking an image and a prompt as inputs and producing corresponding segmentation result. A typical prompt could be one or several 2D points. The image and the prompt are fed into an image encoder and a prompt encoder separately. Subsequently, a transformer-based decoder computes the cross-attention between prompt features and image features to generate the mask. Specifically, SAM can output multiple valid masks with associated confidence scores. In our experiments, we tend to prefer masks with a larger area because they are more likely to represent an entire object. We only resort to selecting masks with a relatively smaller area when the confidence of the larger mask is low (please refer to the supplementary materials for detailed implementation).

Edge weights. The prompt mechanism of SAM introduces flexibility to calculate the edge weights between two 3D superpoints. Part 2.2 of Fig. 2 presents

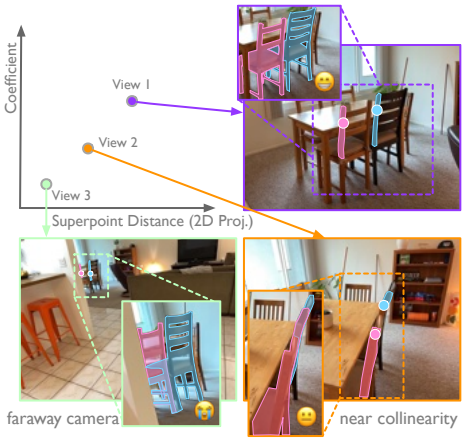

Fig. 3: Relationship of coefficient and 2D superpoints distance. For two superpoints, their distance in 2D images will be farther under near and frontal views than faraway or collinear views. We assume that SAM achieves better performance on near and frontal views. Thus, we consider the 2D distance as a factor in calculating the coefficient of each view.

an illustration of computing the weight. Specifically, we first select a view where both superpoints are visible (if such a view does not exist, we regard the weight to be zero). Then, the two superpoints are projected onto the image space and k points are uniformly sampled in each projection to serve as prompts for running SAM, thereby obtaining a mask for each superpoint. For masks corresponding to the two superpoints, we examine their intersection situation. Specifically, if two superpoints of an edge correspond to masks A and B, then we calculate the edge weight as $w = \max(\frac{A \bigcap B}{A}, \frac{A \bigcap B}{B})$, where $A \bigcap B$ denotes the intersection of A and B.

Taking into account that two superpoints may be co-visible in multiple views, we calculate edge weights estimation across all these views and take their weighted average: $w = \sum_i c_i w_i$, where w_i is the edge weight estimation of view i and c_i is the corresponding coefficient. c_i is computed based on the following two factors: first, we consider the confidence of the two masks predicted by the SAM network; then, we consider the distance between the projections of the two superpoints in that view, as illustrated in Fig. 3. Specifically, for a view, we obtain its score by multiplying the 2D distance of two superpoints with the confidence of two corresponding masks. Then, we perform L1 normalization to multi-view scores to obtain coefficient c_i for each view.

Node features. In addition to edge weights, we also annotate node features with SAM. Specifically, for each node in the graph model, we identify all views that observe the corresponding superpoint. Within the projection range of that superpoint in each view, we randomly sample several points, interpolate to obtain features extracted by the SAM encoder and average the features obtained from

all views to represent the attributes of the node. Part 2.1 of Fig. 2 presents an example of obtaining node features.

3.3 Deep graph cut

We first propose a simple segmentation method that is not based on neural networks. Specifically, based on the superpoint graph constructed in Sec. 3.1 and the edge weights of the graph calculated in Sec. 3.2, we use the edge weights to determine whether two superpoints are connected, then employ a method based on the union-find algorithm [50] to merge all connected superpoints, thereby achieving 3D segmentation of the scene.

To improve the robustness of segmentation, we feed the graph model into a graph neural network (GNN) before segmentation. The GNN has a certain receptive field and can utilize the information of the surrounding nodes and edges and can predict edge affinity scores which can be more reliable than original edge weights. To process the input graph, we design a GNN that consists of graph convolutional layers and fully connected layers. The graph model is first passed through graph convolutional layers to extract features for each vertex. Then, we concatenate features of two vertices with the corresponding edge weight computed in Sec. 3.2, which are fed into fully connected layers to predict the affinity between two vertices. After that, we apply the same segmentation method based on the predicted affinity scores.

Pseudo labels generation. We propose a strategy for training the GNN without 3D ground truth annotation. To supervise the network, we first generate pseudo labels based on a 2D segmentation model. For pseudo-labels, the most ideal case would be to obtain the correct affinity for all edges, but this is unrealistic. In fact, while we require a high degree of accuracy for pseudo-labels, the completeness of these labels is a lesser priority. For this purpose, we use a 2D segmentation network, CropFormer [39], for this task. We first ran CropFormer on all views to obtain the instance segmentation results. For each pair of co-visible superpoints in every view, we record whether these superpoints are within the same mask. If they are co-visible in at least n views and their records are consistent across all these n views, then we treat the pair as a pseudo-label. For example, if two superpoints are within the same mask in all co-visible views, they are treated as a positive sample in the pseudo-labels, vice versa.

The reason for choosing CropFormer is based on our empirical observation that it tends to yield relatively more complete and accurate masks for common object categories. For example, CropFormer consistently segments a chair entirely, whereas SAM sometimes segments parts of it, such as a single chair leg. Although CropFormer has its advantages, we opt for SAM in the previous graph construction stage due to the following considerations: SAM's unique prompt mechanism can use superpoints to control the granularity of segmentation to some extent. Moreover, SAM's design, which allows for predicting multiple masks from a single prompt, makes it more adept at handling uncommon objects. Consequently, we chose SAM for constructing the entire graph, while

using CropFormer to generate relatively incomplete but high-quality pseudo-labels.

Training of the GNN. We employ pseudo-labels generated by CropFormer as supervision. For the edges included in these pseudo-labels, we compare the affinity scores of these edges predicted by GNN with the labels to calculate binary cross-entropy loss, which is defined as

$$L_{\text{BCE}} = s_{\text{ps}} \log(s) + (1 - s_{\text{ps}}) \log(1 - s), \tag{1}$$

where s is the predicted affinity score and s_{ps} is the pseudo-label of corresponding edge. Since pseudo-labels are sparse, we observed that direct training in this manner tends to limit the network's capabilities. To improve the accuracy of the graph network's predictions, we introduce an additional regularization for edges not included in the pseudo-labels during training. This regularization aims to ensure the predicted affinity score s, and the corresponding edge weight predicted by SAM, w_{SAM}, to be as consistent as possible. This loss is defined as

$$L_{\text{reg}} = |w_{\text{SAM}} - 0.5| L_1(s, w_{\text{SAM}}). \tag{2}$$

With this design, the closer w_{SAM} is to 0 or 1, the greater the penalty for inconsistency between s and w_{SAM}. The final loss is defined as $L = L_{\text{BCE}} + L_{\text{reg}}$.

4 Implementation details

Graph construction. When sampling points within the projection of a superpoint as prompts for SAM, we uniformly sample $k = 5$ points within the projected mask. Considering that there might be slight inaccuracies in the camera poses, the points too close to the boundaries of projection masks could potentially fall outside the object. Therefore, we take care to avoid sampling these points during this process.

Training of GNN. We implement the GNN with PyTorch [35] and PyG [15]. When generating pseudo labels with CropFormer, we only consider the superpoint pairs that are co-visible in at least $n = 10$ views with all consistent records. We train the GNN on ScanNet200 with pseudo-labels for 200 epochs, which takes about 20 minutes on an NVIDIA A6000 GPU.

5 Experiments

5.1 Datasets, metrics and baselines

Datasets. We perform the experiments on ScanNet200 [43], ScanNet++ [64] and KITTI-360 [30]. ScanNet200 is built on ScanNet [6], which is a large-scale RGB-D dataset that contains 1613 indoor scenes with geometry acquired by BundleFusion [7] and images captured by iPad Air2. ScanNet++ contains 280 indoor scenes with high-fidelity geometry acquired by the Faro Focus Premium laser scanner as well as high-resolution RGB images captured by iPhone 13 Pro

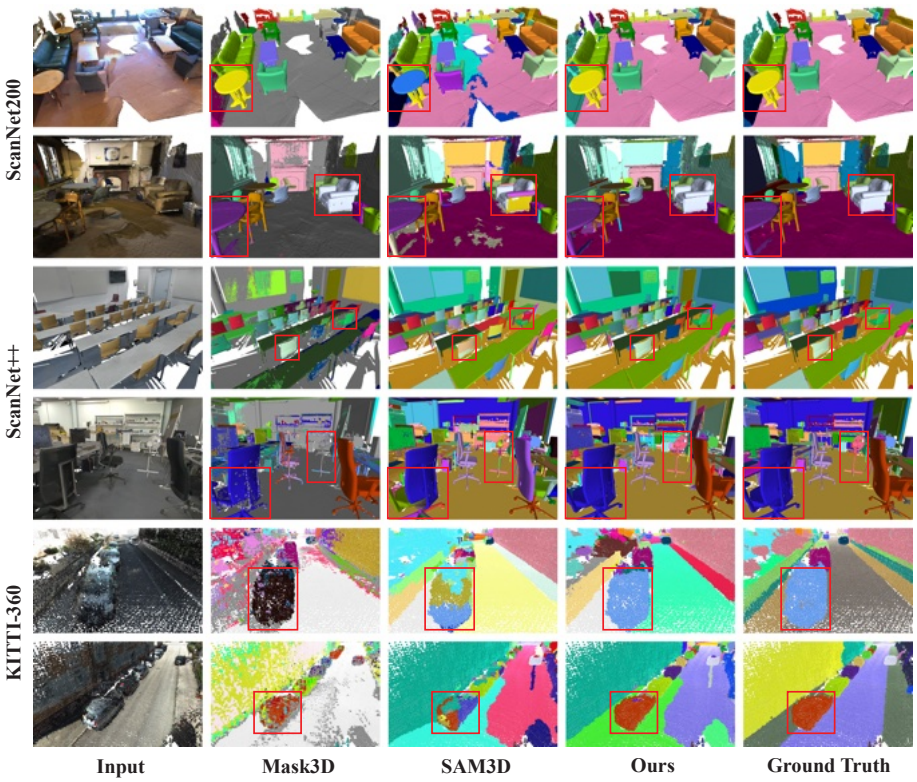

Fig. 4: 3D segmentation results on ScanNet200, ScanNet++ and KITTI-360 datasets. Please zoom in for details. Compared to Mask3D, our method exhibits significantly better generalization on ScanNet++ and KITTI-360 datasets. Moreover, in comparison to SAM3D, our approach can segment objects in the scene more completely and accurately. We observed that Panoptic Lifting struggles to extract satisfactory geometry, so we leave the qualitative comparison with it to the supplementary material.

and a DSLR camera with a fisheye lens. KITTI-360 is a large outdoor dataset with 300 suburban scenes, which consists of 320k images and 100k laser scans obtained through a mobile platform in a driving distance of 73.7 km. All of these datasets are annotated with ground truth camera poses and instance-level semantic segmentations. The GNN in our method is trained on 1201 training scenes of ScanNet200 with our generated pseudo-labels, we then evaluate our method on 312 validation scenes of ScanNet200, 50 of ScanNet++ and 61 scenes of KITTI-360, according to the official split of each dataset.

Metrics. We evaluate our segmentation performance with the widely-used Average Precision (AP) score. We follow the standard defined in [6, 40, 43] for evaluation, calculating AP with thresholds of 50% and 25% (denoted as AP_{50} and AP_{25}, respectively) as well as AP averaged with IoU thresholds from 50% to 95% with a step size of 5% (mAP). Since our method as well as most of baseline

Table 1: Quantitative results of 3D segmentation on ScanNet200, ScanNet++ and KITTI-360 datasets. We report the AP scores averaged on all test scenes. Our method significantly outperforms the baseline methods on all datasets.

	ScanNet200			ScanNet++			KITTI-360		
	mAP	AP_{50}	AP_{25}	mAP	AP_{50}	AP_{25}	mAP	AP_{50}	AP_{25}
Felzenswalb	4.8	9.8	27.5	8.8	16.9	36.1	-	-	-
Guinard	2.9	8.2	33.1	4.3	10.6	32.3	9.3	18.9	39.6
SAM3D (w/o ensemble)	12.1	28.6	54.1	3.0	7.9	22.3	4.6	10.6	26.0
SAM3D (w/ ensemble)	20.9	34.8	51.4	9.3	16.6	29.5	13.0	24.2	41.1
Ours + NCuts	15.7	31.7	59.0	10.1	18.6	34.7	19.5	30.2	45.0
Ours + DBSCAN	10.3	18.6	27.8	10.5	17.2	25.0	20.5	31.4	42.1
Ours (w/o GNN)	19.7	37.7	61.6	13.7	25.2	43.0	22.6	36.2	48.5
Ours (w/ GNN)	**22.1**	**41.7**	**62.8**	**15.3**	**27.2**	**44.3**	**23.8**	**37.2**	**49.1**

methods are class-agnostic, we do not consider semantic class label in evaluation, which follows the setting of [44]. Additionally, we exclude the predicted instances in unannotated regions for all methods to facilitate a fairer comparison.

Baselines. We compare our method with the following baselines: (1) Traditional segmentation methods: [14, 17], which only use geometric information to perform segmentation. (2) 2D-to-3D lifting method: SAM3D [63] and Panoptic Lifting [48]. We report the results of SAM3D with and without the ensemble process. Since Panoptic Lifting is based on NeRF [33] and requires hours for per-scene optimization, we only report corresponding qualitative analyses. (3) Point cloud segmentation method: Mask3D [46], we use their official pretrained models. (4) We segment our SAM annotated graph (without GNN) with traditional spectral clustering methods: Normalized Cuts [47] and DBSCAN [45], as well as the graph cut used in our method.

5.2 Comparisons with the state-of-the-art methods

We evaluate 3D segmentation metrics on ScanNet200, ScanNet++ and KITTI-360 datasets. Averaged quantitative results are shown in Tab. 1. We also provide qualitative results in Fig. 4. By analyzing quantitative and qualitative results, we found that our method significantly outperforms state-of-the-art unsupervised methods.

Panoptic Lifting can achieve reasonably good segmentation results in simpler scenes, but its performance deteriorates in more complex environments with a large number of objects. SAM3D can handle complex scenes, however, its segmentation often results in both large structures, like floors, and smaller objects, like chairs, being divided into multiple segments. In contrast, our method is able to produce more accurate and complete segmentation results. In the experiments of our method combined with Normalized Cuts and DBSCAN, we found that carefully tuning hyperparameters can yield relatively good result for a single

Table 2: Comparison with Mask3D. While Mask3D shows better performance than ours on ScanNet200, it cannot generalize well to ScanNet++ and KITTI-360.

		Training Data	mAP	AP_{50}	AP_{25}
ScanNet200	Mask3D	GT of ScanNet200	**53.3**	**71.9**	**81.6**
	Mask3D	GT of ScanNetV2	45.1	62.6	70.5
	Ours (w/o GNN)	-	19.7	37.7	61.6
	Ours (w/ GNN)	Pseudo-GT of ScanNet200	22.1	41.7	62.8
ScanNet++	Mask3D	GT of ScanNet200	4.6	10.5	22.9
	Mask3D	GT of ScanNetV2	3.7	7.9	15.6
	Ours (w/o GNN)	-	13.7	25.2	43.0
	Ours (w/ GNN)	Pseudo-GT of ScanNet200	**15.3**	**27.2**	**44.3**
KITTI-360	Mask3D	GT of ScanNet200	0.2	0.9	7.0
	Mask3D	GT of ScanNetV2	0.3	1.0	8.0
	Ours (w/o GNN)	-	22.6	36.2	48.5
	Ours (w/ GNN)	Pseudo-GT of ScanNet200	**23.8**	**37.2**	**49.1**

scene. However, each scene varies significantly in scale and the number of objects, leading to considerable differences in the optimal hyperparameters. When we apply uniform hyperparameters across all dataset, the averaged metrics are not ideal.

Comparisons with supervised method. In addition to unsupervised baselines, we also compare our method with Mask3D, which is the state-of-the-art supervised learning method. We evaluate the class-agnostic AP scores, and the results are shown in Tab. 2. When evaluation, we use their official pretrained models. Specifically, we first evaluate their model trained on the ScanNet200 training set, which shows very good results on the ScanNet200 validation set. However, when applied to the ScanNet++ dataset, there is a significant performance drop. This is because ScanNet++, although also indoor scene data, has a different data collection method from ScanNet200, indicating that the Mask3D method is quite sensitive to differences in aspects such as the point cloud collection method. When applied to KITTI-360, the performance becomes extremely poor, as KITTI-360 is an outdoor scene, which is quite different from ScanNet200 so that Mask3D cannot generalize well. Then we evaluate Mask3D model trained on the ScanNetV2 training set. Compared to ScanNet200, ScanNetV2 lacks annotations for some categories of objects, resulting in decreased performance on the ScanNet200 validation set. This indicates that Mask3D has limited ability to generalize to different categories of objects even in the same type of scenes, and it also suffers significant performance declines on ScanNet++ and KITTI-360.

In contrast, our method (without GNN) achieves good results on all three datasets without any training, although our method performs worse on ScanNet200 than Mask3D, which is because Mask3D is trained with GT annotations provided by ScanNet200, allowing it to learn strong priors of this dataset. Our method shows a clear advantage on the ScanNet++ and KITTI-360 datasets.

Table 3: Ablation studies of graph cut implementation.

	mAP	AP_{50}	AP_{25}
w/o GNN	19.7	37.7	61.6
w/o node features	19.4	37.5	61.3
w/o edge weights	10.1	21.5	42.2
w/o regularization loss	19.9	38.1	61.4
our full method	**22.1**	**41.7**	**62.8**

Table 4: Ablation studies of k.

k	mAP	AP_{50}	AP_{25}
1	20.3	37.7	60.7
3	21.2	38.8	61.2
5	**22.1**	**41.7**	**62.8**
7	22.0	40.1	62.1
9	21.9	39.9	62.3

Table 5: Ablation studies of n.

n	mAP	AP_{50}	AP_{25}
1	20.3	38.7	61.2
5	21.7	40.9	**63.6**
10	**22.1**	**41.7**	62.8
20	20.4	39.1	62.6
30	20.2	39.3	61.9

After using the GNN trained with pseudo-labels on ScanNet200, our method not only improves on the ScanNet200 validation set but also shows improvement on ScanNet++ and KITTI-360. This indicates that the GNN module in our method helps with segmentation and learns general prior for segmentation agnostic to the training dataset, thus has good generalization capabilities to data with large differences.

5.3 Ablation studies

To analyze the effectiveness of each module and design in our method, as well as the impact of hyper-parameters on performance, we conduct ablation studies on ScanNet200.

Ablation studies of graph cut implementation. We evaluate with five configurations: (1) Apply graph cut directly on edge weights of the SAM annotated graph (without GNN refinement). (2) Annotate the graph without node features. (3) Annotate the graph without edge weights. (4) Train GNN without regularization loss. (5) Our full method. We report the quantitative results in Tab. 3. The results indicate that the absence of a GNN leads to a decline in segmentation performance. Node features have a certain impact on the predictive performance of the GNN, while edge weights have a very significant impact, almost playing a dominant role. Additionally, we found that regularization loss is also important.

Ablation studies of number of prompt points. We conduct ablation studies on the number of prompt points k sampled in each superpoint projection, the results are shown in Tab. 4. In our experiment, we chose $k = 5$, and the ablation results suggest that our method is not very sensitive to the value of k, but

Table 6: Analyses of the choice for pseudo-labels generation.

	Accuracy	Precision	Recall	F1-Score
SAM	0.892	0.934	0.888	0.911
CropFormer	**0.912**	**0.934**	**0.923**	**0.929**

it is affected to some extent. The results are relatively poor when $k = 1$, as sampling only one point within a projection mask leads to significant information loss, especially when the projection mask area is large. When k is larger, the performance slightly deteriorates, which may be due to SAM not being suitable for accepting too many points as a prompt.

Ablation studies of minimum number of co-visible views. We conduct ablation studies on the minimum number of co-visible views n required for generating pseudo labels, the results are shown in Tab. 5. In our experiment, we chose $n = 10$, and the ablation results indicate that our method is robust to the choice of n, but our performance is relatively poorer when n is either too small or too large. This is because when n is small, a large number of unreliable pseudo-labels are generated. When n is large, although the quality of the pseudo-labels is high, they become very sparse. In both cases, this leads to suboptimal results.

Ablation studies of pseudo-label generation. Furthermore, we conduct experiments to analyze the advantages of using CropFormer to generate pseudo-labels compared to SAM. For the labels generated by CropFormer, we also use SAM to predict a label. The accuracy scores of both methods are evaluated using the ground truth annotations provided by ScanNet200. As indicated in the Tab. 6, CropFormer achieves higher accuracy.

6 Conclusion

In this paper, we introduced a novel 3D segmentation method with SAM guided graph cut. The key idea is to pre-segment 3D scenes into superpoints, and then utilize the prompt mechanism of SAM to assess the affinity scores between superpoints. We propose a GNN based graph cut method to achieve robust segmentation, which is trained with pseudo-labels generated by a 2D segmentation network. Experiments showed that the proposed method is able to achieve accurate segmentation results and can generalize well to different datasets.

Discussion. Our method not only requires geometric data (mesh/point cloud) but also needs multi-view images as input, which to some extent limits its application scenarios. Moreover, we perform segmentation based on merging superpoints. When an object is part of a superpoint, we are unable to segment it out. This situation occurs occasionally, for example, a poster adhered to a wall. To address this, a more sophisticated pre-segmentation model, which consider not only geometric information but also semantics, should be designed. One viable approach is to also use the guidance from SAM or other visual models during the pre-segmentation stage. We leave it to future works.

Acknowledgement. The authors would like to acknowledge the support from NSFC (No. 62322207), Ant Group and Information Technology Center and State Key Lab of CAD&CG, Zhejiang University.

References

1. Bhalgat, Y., Laina, I., Henriques, J.F., Zisserman, A., Vedaldi, A.: Contrastive lift: 3d object instance segmentation by slow-fast contrastive fusion. arXiv preprint arXiv:2306.04633 (2023) 2, 4
2. Cen, J., Wu, Y., Wang, K., Li, X., Yang, J., Pei, Y., Kong, L., Liu, Z., Chen, Q.: Sad: Segment any rgbd. arXiv preprint arXiv:2305.14207 (2023) 4
3. Chen, X., Tang, J., Wan, D., Wang, J., Zeng, G.: Interactive segment anything nerf with feature imitation. arXiv preprint arXiv:2305.16233 (2023) 4
4. Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention mask transformer for universal image segmentation. In: CVPR (2022) 2, 4
5. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016) 4
6. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: CVPR (2017) 6, 9, 10
7. Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C.: Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. ACM ToG (2017) 9
8. Dann, C., Gehler, P., Roth, S., Nowozin, S.: Pottics–the potts topic model for semantic image segmentation. In: Pattern Recognition. Springer (2012) 6
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009) 4
10. Ding, R., Yang, J., Xue, C., Zhang, W., Bai, S., Qi, X.: Pla: Language-driven open-vocabulary 3d scene understanding. In: CVPR (2023) 3
11. Elich, C., Engelmann, F., Kontogianni, T., Leibe, B.: 3D-BEVIS: Birds-Eye-View Instance Segmentation. In: The German Conference on Pattern Recognition (2019) 1, 3
12. Engelmann, F., Bokeloh, M., Fathi, A., Leibe, B., Nießner, M.: 3D-MPA: Multi Proposal Aggregation for 3D Semantic Instance Segmentation. In: CVPR (2020) 1, 3
13. Everingham, M., Eslami, S.M.A., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. IJCV (2015) 4
14. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV (2004) 6, 11
15. Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds (2019) 9
16. Graham, B., Engelcke, M., van der Maaten, L.: 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In: CVPR (2018) 3
17. Guinard, S., Landrieu, L., Vallet, B.: Weakly supervised segmentation-aided classification of urban scenes from 3d lidar point clouds. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences (2017) 6, 11

18. Gupta, A., Dollar, P., Girshick, R.: Lvis: A dataset for large vocabulary instance segmentation. In: CVPR (2019) 4
19. Han, L., Zheng, T., Xu, L., Fang, L.: OccuSeg: Occupancy-aware 3D Instance Segmentation. In: CVPR (2020) 1, 3
20. Hou, J., Dai, A., Nießner, M.: 3D-SIS: 3D Semantic Instance Segmentation of RGB-D Scans. In: CVPR (2019) 3
21. Huang, J., You, S.: Point Cloud Labeling Using 3D Convolutional Neural Network. In: ICPR (2016) 3
22. Hui, L., Tang, L., Shen, Y., Xie, J., Yang, J.: Learning superpoint graph cut for 3d instance segmentation. In: NeurIPS (2022) 4
23. Jiang, L., Zhao, H., Shi, S., Liu, S., Fu, C.W., Jia, J.: PointGroup: Dual-Set Point Grouping for 3D Instance Segmentation. In: CVPR (2020) 1, 3
24. Ke, L., Ye, M., Danelljan, M., Liu, Y., Tai, Y.W., Tang, C.K., Yu, F.: Segment anything in high quality. arXiv preprint arXiv:2306.01567 (2023) 2, 4
25. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment anything. In: ICCV (2023) 2, 4, 5, 6
26. Kong, X., Liu, S., Taher, M., Davison, A.J.: vmap: Vectorised object mapping for neural field slam. In: CVPR (2023) 2, 5
27. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.J., Shamma, D.A., et al.: Visual genome: Connecting language and vision using crowdsourced dense image annotations. IJCV (2017) 4
28. Lahoud, J., Ghanem, B., Pollefeys, M., Oswald, M.R.: 3D Instance Segmentation via Multi-task Metric Learning. In: CVPR (2019) 1, 3
29. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: PointCNN: Convolution on X-transformed Points. In: NeurIPS (2018) 1, 3
30. Liao, Y., Xie, J., Geiger, A.: Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. PAMI (2022) 3, 9
31. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014) 4
32. Long, J., Shelhamer, E., Darrell, T.: Fully Convolutional Networks for Semantic Segmentation. In: CVPR (2015) 3
33. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020) 4, 11
34. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM ToG (2022) 4
35. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. NeurIPS (2019) 9
36. Peng, S., Genova, K., Jiang, C.M., Tagliasacchi, A., Pollefeys, M., Funkhouser, T.: OpenScene: 3D Scene Understanding with Open Vocabularies. In: CVPR (2023) 3
37. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In: CVPR (2017) 1
38. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In: NeurIPS (2017) 1, 3
39. Qi, L., Kuen, J., Shen, T., Gu, J., Li, W., Guo, W., Jia, J., Lin, Z., Yang, M.H.: High quality entity segmentation. In: ICCV (2023) 2, 4, 8
40. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NeurIPS (2015) 10

41. Robert, D., Raguet, H., Landrieu, L.: Efficient 3d semantic segmentation with superpoint transformer. In: ICCV (2023) 4
42. Robert, D., Raguet, H., Landrieu, L.: Scalable 3d panoptic segmentation as superpoint graph clustering. In: 3DV (2024) 4
43. Rozenberszki, D., Litany, O., Dai, A.: Language-grounded indoor 3d semantic segmentation in the wild. In: ECCV (2022) 3, 9, 10
44. Rozenberszki, D., Litany, O., Dai, A.: Unscene3d: Unsupervised 3d instance segmentation for indoor scenes. arXiv preprint arXiv:2303.14541 (2023) 11
45. Schubert, E., Sander, J., Ester, M., Kriegel, H.P., Xu, X.: Dbscan revisited, revisited: why and how you should (still) use dbscan. ACM Transactions on Database Systems (TODS) (2017) 11
46. Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., Leibe, B.: Mask3d for 3d semantic instance segmentation. arXiv preprint arXiv:2210.03105 (2022) 1, 3, 11
47. Shi, J., Malik, J.: Normalized cuts and image segmentation. PAMI (2000) 11
48. Siddiqui, Y., Porzi, L., Bulò, S.R., Müller, N., Nießner, M., Dai, A., Kontschieder, P.: Panoptic lifting for 3d scene understanding with neural fields. In: CVPR (2023) 2, 4, 11
49. Tang, L., Hui, L., Xie, J.: Learning inter-superpoint affinity for weakly supervised 3d instance segmentation. In: ACCV (2022) 4
50. Tarjan, R.E.: Efficiency of a good but not linear set union algorithm. Journal of the ACM (JACM) (1975) 3, 8
51. Tchapmi, L.P., Choy, C.B., Armeni, I., Gwak, J., Savarese, S.: SEGCloud: Semantic Segmentation of 3D Point Clouds. In: 3DV (2017) 3
52. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: KPConv: Flexible and Deformable Convolution for Point Clouds. In: ICCV (2019) 1, 3
53. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention Is All You Need. In: NeurIPS (2017) 3
54. Vu, T., Kim, K., Luu, T.M., Nguyen, X.T., Yoo, C.D.: SoftGroup for 3D Instance Segmentation on 3D Point Clouds. In: CVPR (2022) 1, 3
55. Wang, T., Li, J., An, X.: An Efficient Scene Semantic Labeling Approach for 3D Point Cloud. In: IEEE International Conference on Intelligent Transportation Systems (ITSC) (2015) 3
56. Wang, W., Yu, R., Huang, Q., Neumann, U.: SGPN: Similarity Group Proposal Network for 3D Point Cloud Instance Segmentation. In: CVPR (2018) 1, 3
57. Wu, Q., Liu, X., Chen, Y., Li, K., Zheng, C., Cai, J., Zheng, J.: Object-compositional neural implicit surfaces. In: ECCV (2022) 2, 5
58. Wu, Q., Wang, K., Li, K., Zheng, J., Cai, J.: Objectsdf++: Improved object-compositional neural implicit surfaces. In: ICCV (2023) 2, 5
59. Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y.: SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters. In: ECCV (2018) 3
60. Yan, M., Zhang, J., Zhu, Y., Wang, H.: Maskclustering: View consensus based mask graph clustering for open-vocabulary 3d instance segmentation. arXiv preprint arXiv:2401.07745 (2024) 4
61. Yang, B., Zhang, Y., Xu, Y., Li, Y., Zhou, H., Bao, H., Zhang, G., Cui, Z.: Learning object-compositional neural radiance field for editable scene rendering. In: ICCV (2021) 2, 4
62. Yang, B., Wang, J., Clark, R., Hu, Q., Wang, S., Markham, A., Trigoni, N.: Learning Object Bounding Boxes for 3D Instance Segmentation on Point Clouds. In: NeurIPS (2019) 1, 3

63. Yang, Y., Wu, X., He, T., Zhao, H., Liu, X.: Sam3d: Segment anything in 3d scenes. arXiv preprint arXiv:2306.03908 (2023) 2, 4, 11
64. Yeshwanth, C., Liu, Y.C., Nießner, M., Dai, A.: Scannet++: A high-fidelity dataset of 3d indoor scenes. In: ICCV (2023) 3, 9
65. Yin, Y., Liu, Y., Xiao, Y., Cohen-Or, D., Huang, J., Chen, B.: Sai3d: Segment any instance in 3d scenes. arXiv preprint arXiv:2312.11557 (2023) 4
66. Yin, Y., Fu, Z., Yang, F., Lin, G.: Or-nerf: Object removing from 3d scenes guided by multiview segmentation with neural radiance fields. arXiv preprint arXiv:2305.10503 (2023) 4
67. Zhi, S., Laidlow, T., Leutenegger, S., Davison, A.J.: In-place scene labelling and understanding with implicit scene representation. In: ICCV (2021) 4