

# SeMLaPS: Real-time Semantic Mapping with Latent Prior Networks and Quasi-Planar Segmentation

Jingwen Wang<sup>1,2</sup>, Juan Tarrío<sup>1</sup>, Lourdes Agapito<sup>2</sup>, Pablo F. Alcantarilla<sup>1</sup>, Alexander Vakhitov<sup>1</sup>

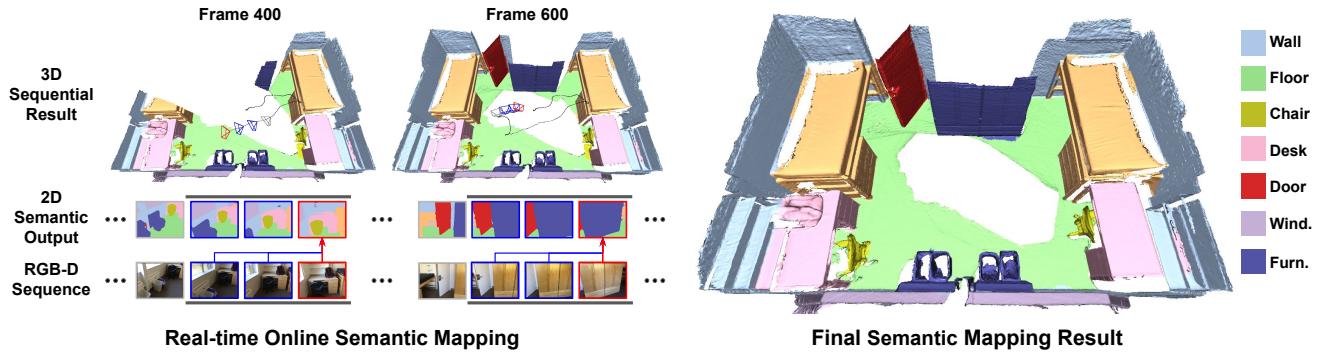


Fig. 1: SeMLaPS takes an RGB-D sequence as input and incrementally builds a semantic map of the scene. Our Latent Prior Network (LPN) outputs 2D semantic labels that encode history information of prior views via differentiable feature re-projection. 2D labels are lifted to 3D and fused temporarily via Bayesian fusion. Based on a novel geometric map over-segmentation, our Segment-Convolutional Network further refines the 3D semantic map by applying convolutions at the segment-level, resulting in state-of-the-art accuracy within real-time 2D-3D networks, and good cross-sensor generalization capabilities at the same time.

**Abstract**—The availability of real-time semantics greatly improves the core geometric functionality of SLAM systems, enabling numerous robotic and AR/VR applications. We present a new methodology for real-time semantic mapping from RGB-D sequences that combines a 2D neural network and a 3D network based on a SLAM system with 3D occupancy mapping. When segmenting a new frame we perform latent feature re-projection from previous frames based on differentiable rendering. Fusing re-projected feature maps from previous frames with current-frame features greatly improves image segmentation quality, compared to a baseline that processes images independently. For 3D map processing, we propose a novel geometric quasi-planar over-segmentation method that groups 3D map elements likely to belong to the same semantic classes, relying on surface normals. We also describe a novel neural network design for lightweight semantic map post-processing. Our system achieves state-of-the-art semantic mapping quality within 2D-3D networks-based systems and matches the performance of 3D convolutional networks on three real indoor datasets, while working in real-time. Moreover, it shows better cross-sensor generalization abilities compared to 3D CNNs, enabling training and inference with different depth sensors. Code and data can be found at <https://github.com/slamcore/semlaps>.

## I. INTRODUCTION

SLAM systems use semantics for better pose estimation or re-localization [2], [3] or to work in dynamic scenes [2], [3]. It can facilitate downstream tasks such as robotic navigation [4] or augmented reality (AR) experiences [5].

Real-time semantic mapping methods usually rely on 2D convolutional neural networks with optional 3D post-processing (2D-3D networks) to annotate incoming images with semantics, using back-projection to lift the semantic labels to the 3D map [6], [3], [7], [5], [8], [1], while recent FP-Conv [7] or SVCNN [6] also rely on lightweight

3D post-processing. 2D-3D networks repetitively process images with similar visual content, solving 2D semantic segmentation from scratch for each image, which may be redundant [9]; lack multi-view consistency in 2D labels [10]; suffer from occlusions or object scale uncertainty [11]. Direct processing of 3D geometry with 3D convolutions, e.g. with MinkowskiNet [12] or SparseConvNet [13], outperforms state-of-the-art 2D-3D networks. However, many semantic SLAM tasks require 2D image-level semantic labels, e.g. for semantic landmark annotation as in [2] or for semantics-based processing of dynamics as in [14]. This makes the use of 3D networks as main semantic inference engines in SLAM systems less appealing, even given recent progress towards using 3D networks in an online and incremental fashion [15].

Semantic SLAM systems may struggle with cross-sensor generalization, because depth sensors have different accuracy and noise properties. For some sensors we have access to large datasets with semantic annotations (ScanNet [16]), whilst for others, e.g. RealSense [17], this is not the case.

In this work we overcome some of the aforementioned drawbacks of 2D-3D networks-based semantic SLAM systems, see Fig. 1. Firstly, we propose Latent Prior Networks (LPNs) - 2D convolutional networks that leverage prior knowledge about the scene using a latent feature re-projection mechanism based on differentiable rendering, eliminating the image processing redundancy. Compared to prior methods [9], [10] that re-project late features or segmentation labels, we use early features and rely on differentiable rendering. This leads to improved quality of image-level semantic labels compared to the state-of-the-art 2D-3D networks-based method SVCNN [6].

Secondly, we propose quasi-planar over-segmentation (QPOS) for lightweight 3D map post-processing. Together

<sup>1</sup> Slamcore Ltd. <sup>2</sup> University College London

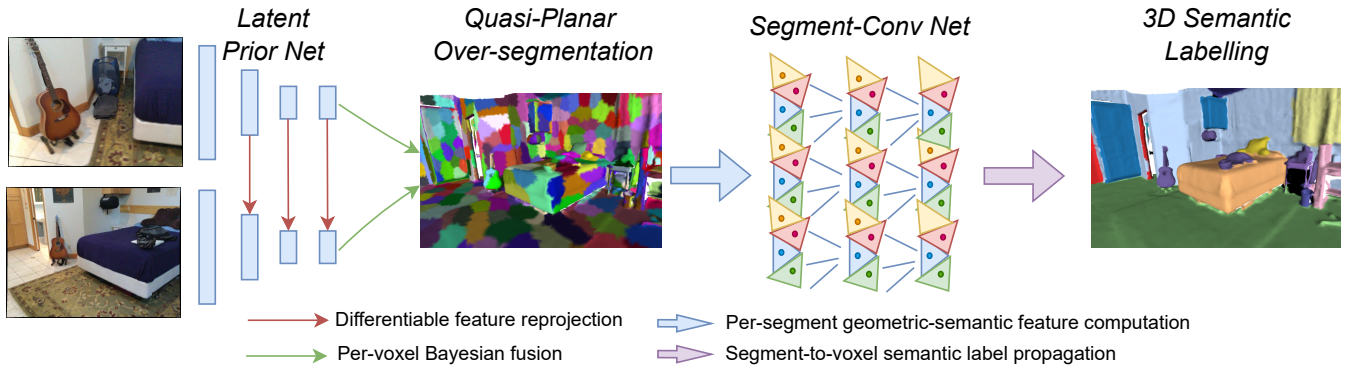


Fig. 2: SeMLaPS runs on top of a state-of-the-art visual inertial SLAM system performing camera pose tracking and voxel-based 3D geometry reconstruction. SeMLaPS has three main stages: First, a Latent Prior Network (LPN), relying on a fast 2D neural network backbone enhanced with differentiable latent feature re-projection, processes the RGB-D frames. It outputs 2D semantic labels, which are further lifted to 3D and aggregated over time with Bayesian fusion [1]. Next, Quasi-Planar Over-Segmentation (QPOS) groups surface voxels into segments, associated with semantic class probabilities and geometric features. Finally, our Segment-Convolutional Network (SegConvNet) applies convolutions at segment-level and obtains final semantic labels.

with a novel SegConvNet for map post-processing this helps our system to improve semantic map annotation quality over SVCNN, matching the accuracy of 3D network-based approaches.

As a third contribution we report a significant improvement in cross-sensor generalization of a trained model based on a 2D LPN compared to a baseline 3D CNN. This may be an additional justification 2D-3D networks-based designs in semantic SLAM systems. We demonstrate our novel Semantic Mapping approach on our new RealSense dataset. See the code and data at <https://github.com/slamcore/semlaps>.

## II. RELATED WORK

**2D-3D networks-based semantic mapping.** Single frame RGB semantic segmentations can be lifted to 3D and fused using Bayesian [1], [19] or geometric [5] fusion. For frame-level fusion DA-RNN [9] and MVCNet [10] rely on depth-based re-projection of labels or late features, while our Latent Prior Network uses differentiable rendering of early features.

3D semantic label refinement is done with a CRF in ProgressiveFusion [20], a lightweight PointConv [21]-based method in FP-Conv [7] and SVCNN [6]. SVCNN defines state-of-the-art for the 2D-3D networks-based semantic mapping systems. The design of SeMLaPS is similar to SVCNN, with several major differences in 2D and 3D networks and map over-segmentation.

**3D over-segmentation.** Early methods for voxel grid over-segmentation were designed by extending image over-segmentation methods [22], [23], [24]. VCCS [25] was one of the first methods directly aimed at 3D over-segmentation, proposing a new error metric for this task. Online 3D mapping requires incremental and real-time methods. Tateno *et al.* [26] uses a 3D map as a set of object-like point clusters found using surface normals, ProgressiveFusion [20] uses a simplistic incremental voxel clustering method, while SVCNN [6] relies on a modified supervoxel segmentation [27]. Our proposed incremental QPOS improves over supervoxel segmentation [6], [27].

**3D networks for semantic mapping.** Methods of this class process a 3D reconstruction of a scene directly and produce semantic labels as output. PointNet-based semantic segmentation methods process unordered point clouds at multiple scales [28]. PointConv [21] and KPConv [29] propose convolution operations on point clouds. Sparse sub-manifold convolutions [13] and MinkowskiNet [12] process only occupied surface voxels in dense voxel grids, with reasonable memory requirements. BP-Net [30] leverages both a 3D network and a 2D network, linked by a feature projection mechanism. INS-Conv [15] shows a way to run 3D network-based inference in an online fashion, matching the accuracy of the offline 3D networks. However, it does not produce image-level semantic labels required for other semantic SLAM tasks.

## III. REAL-TIME SEMANTIC MAPPING SYSTEM

### A. System Overview

Our pipeline relies on a SLAM system, see Sec. III-B; we process the 2D images with a novel 2D Latent Prior Network (LPN), Sec. III-C. We over-segment the map with the novel Quasi-Planar Over-Segmentation (QPOS) described in Sec. III-D. The Segment-Convolutional Network (SegConvNet), see Sec. III-E, gives the final output. Fig. 2 shows the overall pipeline.

### B. Real-Time Dense 3D Occupancy Mapping

We rely on a feature-based visual inertial SLAM system (similar to [31]) which additionally outputs globally corrected trajectories, when loop-closures are detected and optimised. We fuse depth images and 6 Degrees of Freedom (DoF) poses into a sub-map based 3D occupancy map. Inside each submap, occupancy information is stored in an adaptive resolution octree following [32].

### C. Latent Prior Network

Our LPN relies on the knowledge obtained from several frames, see Fig. 3. As opposed to [9], [10], we aim to inject this prior knowledge as early as possible.

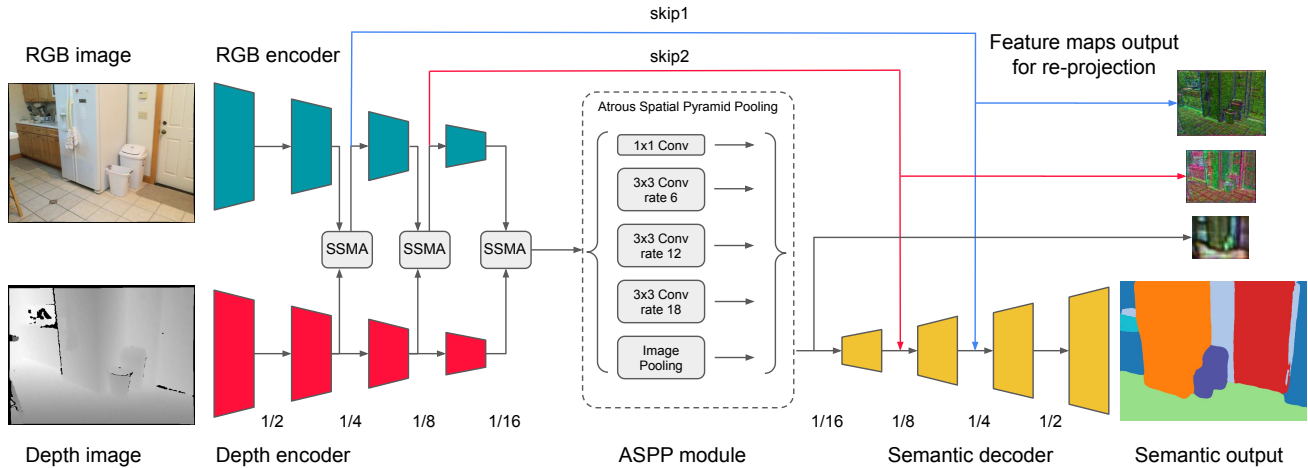


Fig. 3: Our Latent Prior Network (LPN) architecture. For an individual frame, LPN takes in RGB and depth as input and produces the pixel-wise semantic class probabilities as well as intermediate feature maps at  $\times 4$ ,  $\times 8$  and  $\times 16$  stride. Features extracted from RGB and depth are fused at multiple resolutions with SSMA module [18]. The Atrous Spatial Pyramid Pooling (ASPP) is applied at the bottleneck followed by a lightweight decoder.

We adopt SSMA [33], using separate encoders for RGB and depth inputs, but replace the ResNet-50 encoder with a lightweight MobileNetV3 [34], keeping computational complexity in mind, see Fig. 3.

To enforce the prior from previous views, we propose to re-project feature maps at different resolutions to a common reference view using the depth maps and the camera poses provided by the SLAM system. We use  $\times 4$ ,  $\times 8$  and  $\times 16$  downsampled feature maps for re-projection, see Fig. 3. To get smooth gradient propagation, we employ Pytorch3D for feature re-projection, see Sec. 3.3 in [35]. We fuse re-projected and reference view feature maps using average pooling.

During training, at each iteration we sample  $N$  adjacent frames and randomly select one as the reference view. We randomly select the reference view instead of using the last one directly to simulate more diverse motion patterns, which serves as a data augmentation process as well. Latent priors from  $N - 1$  neighbouring frames are warped to the reference view. We then compute the cross-entropy losses:  $\mathcal{L}_{sem}$  from the reference view output, and  $\mathcal{L}_{aux}$  neighboring views, to encourage plausible single view prediction. Our final loss is the weighted sum  $\mathcal{L} = \mathcal{L}_{sem} + w\mathcal{L}_{aux}$ . We empirically found the auxiliary loss to improve around 0.7 2D mIoU points for our default setting.

LPN design is flexible: firstly, although  $N$  is fixed during training, at test time our LPN can take an arbitrary number of views; secondly, we can do inference in sequential mode processing each frame once and reusing feature maps computed for previous view; finally, while LPN requires depth for feature re-projection and cross-view fusion, at the feature extraction stage we can rely on RGB-only input and drop the depth encoder (red part in Fig. 3).

#### D. Quasi-Planar Over-Segmentation

Following [6], [26], [36], we aim to reduce the map cardinality by grouping voxels into segments. We extract a list of surface voxels and estimate their normals using the

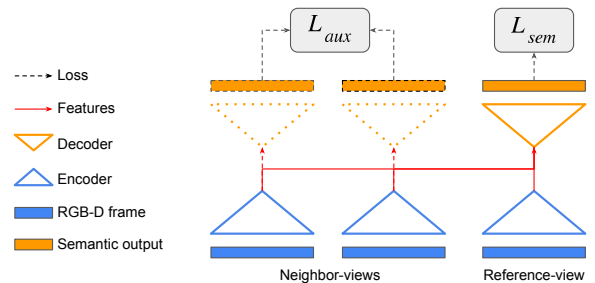


Fig. 4: Latent prior re-projection process. Feature maps at multiple resolutions are re-projected and fused to the reference view. At training time the views come in mini-batches. We predict semantic outputs for computing the main  $\mathcal{L}_{sem}$  and auxiliary  $\mathcal{L}_{aux}$  losses. At test-time we process frames sequentially, predicting the output for the reference view only.

distance field, build an undirected weighted graph  $(V, E, w)$  with voxels as vertices  $V$ , edges  $E$  between voxels sharing a face, and weights as in [36]. An over-segmentation  $S$  maps voxels to segment labels. We set  $S(i) = i$ , sort the edges by weight as in [24] and merge one by one if the cost  $F(S)$  decreases:

$$F(S) = \frac{1}{|L(S)|} \sum_{l \in L(S)} \left| \sum_i 1_{S(i)=l} - \bar{s} \right|, \quad (1)$$

where  $L(S)$  is the set of segment labels and  $\bar{s}$  is the expected spatial segment size.

To improve the quality of segment boundaries, we apply a fast post-processing step, assigning a voxel to a segment with the lowest association cost:

$$C(i, l) = (p_i - p_l)^T \Sigma_l^{-1} (p_i - n_l) + \nu(1 - |n_i \cdot n_l|), \quad (2)$$

where  $p_i, n_i$  are the position and normal of the voxel  $i$ ,  $p_l, \Sigma_l, n_l$  are the center, covariance and normal of the segment  $l$ , and  $\nu$  is a weighting factor. For efficiency, we use a KD-tree of the segment centres.

The QPOS method has  $O(|E|)$  complexity, and we use it in an incremental fashion after the map update, using only updated voxels in  $V$  and  $E$ , trying to associate new

voxels with observed segments using (2) before creating new segments. We use the QPOS results as input to the SegConvNet described next.

### E. Segment-Convolutional Network

Optimal segment sizes may differ depending on scene properties. We propose a convolution operation tailored for non-uniformly-sized segments while SVCNN [6] assumes size uniformity. As in PointConv [21], we define the weight-net  $W(\cdot)$  as a multi-layer perceptron (MLP) for predicting the convolution kernel and the feature-net  $\Phi(\cdot)$  as a MLP for feature processing:

$$SF_{out}(s_x) = \sum_{s_y \in N(s_x)} W(s_y - s_x, \phi_{xy}) \Phi(SF_{in}(s_y)), \quad (3)$$

where  $N(s_x)$  are the neighbouring segments for  $x$ ,  $s_x, s_y$  are the segments' centers, the input segment feature  $SF_{in}$  is the averaged feature within a segment, and  $\phi_{xy} = [n_x \cdot n_y, r \cdot n_x, r \cdot n_y, n_y \cdot v, n_y \cdot w, r_x^y \cdot n_x, r_x^y \cdot (n_y \times n_x), \|r_x^y\|, \sigma_x, \sigma_y]$  where  $n_x$  is the surface normal of segment  $s_x$ ,  $r_x^y = s_y - s_x$  is the displacement vector between the two segments, and  $r = \frac{r_x^y}{\|r_x^y\|}$ ,  $v, w$  is the ortho-normal basis constructed by the Gram-Schmidt process from  $r_x^y$  and  $n_x$ ,  $\sigma_x = \sqrt{r^T \Sigma_x r}$ ,  $\sigma_y = \sqrt{r^T \Sigma_y r}$ , and  $\Sigma_x, \Sigma_y$  represents the spatial covariances of  $s_x$  and  $s_y$ . Our  $\phi_{x,y}$  augments the view-invariant (VI) features proposed in [37] with directional variances  $\sigma_x, \sigma_y$ . As for the input segment features  $SF_{in}$ , we use averaged voxel-wise predicted class probabilities, as well as a 9-D geometric feature consisting of RGB color, position and surface normal. For each segment-convolutional layer, we use a 2-layer MLP with a hidden dimension of 8 as  $W(\cdot)$  and a 2-layer MLP with a hidden dimension of 64 as  $\Phi(\cdot)$ . We stack 3 segment-convolutional layers to form our Segment-Convolutional Network (SegConvNet).

### F. Semantic Mapping with RealSense

While SceneNN [38] and ScanNet [16], are captured with Kinect-like sensors, depth sensing accuracy of popular RealSense devices is significantly lower. To understand the cross-sensor generalization abilities of different semantic mapping methods, we propose a four-sequence RGB-D test dataset captured using RealSense D455 [17]. It has ground truth poses obtained using our visual inertial SLAM system and meshes reconstructed using our dense mapping system based on TSDF fusion with a voxel size of 0.01 m. Meshes are manually annotated with semantic labels in consistency with ScanNet [16]. There are four indoor scenes (meeting room, lab, kitchen and office lounge). We will release the dataset upon paper acceptance.

### G. System Implementation Details

Finally, we provide implementation details of our entire system and the networks proposed to perform online real-time semantic mapping.

**Latent Prior Network.** We train LPN on 1201 training sequences of the ScanNet v2 dataset with a step of 20 between adjacent frames, using Adam [39] for 20 epochs

with an initial learning rate of  $1e-4$  and one-cycle learning rate scheduler; We use  $N = 3$  and perform random scale, crop, flip, Gaussian blur and random view-order permutation for data augmentation. Training takes around 3 days on a single nVidia RTX-3090ti GPU with a batch size of 8.

**Segment-Convolutional Network.** We train the SegConvNet described in sec. III-E on the 1201 meshes of ScanNet v2 training split. We first run our QPOS for meshes with segment size  $\bar{s} = 60$  vertices and transfer the vertex-wise GT labels to segment-level GT labels via majority vote. We run our trained LPN sequentially on each scene, transfer the 2D label predictions to 3D meshes using Bayesian Fusion [1].

The above data generation process creates around 2M segments in total. We train our SegConvNet with Adam [39] optimizer with an initial learning rate of  $5e-4$  and one-cycle scheduler. The training takes around 2 hours on a single Nvidia RTX-3090ti GPU with a batch size of 12 for 100 epochs.

**System Design Details.** We run LPN sequentially and perform Bayesian fusion at each key-frame when the SLAM system updates the map geometry. So, each frame is processed by the LPN only once, reference frames getting re-projected feature maps from non-reference frames as inputs. Next, we perform QPOS only on regions that are affected by the map update, updating the properties (features, segment centers, connecting topology, etc.) of the segments. Those updated segments and their K-nearest neighbors are fed to SegConvNet to predict the updated class labels.

## IV. EXPERIMENTS

We start by introducing baselines and metrics, followed by a description of the evaluation of 2D and 3D semantics and over-segmentation, and conclude with an ablation study and timing analysis.

We use three different datasets. ScanNet [16] has 1201 and 312 indoor RGB-D sequences in the training and validation sub-sets, poses and meshes are obtained with BundleFusion [40]. SceneNN [38] has 76 indoor RGB-D sequences, poses and meshes obtained using ElasticReconstruction [41]. ScanNet and SceneNN are captured with Kinect-like sensors. Our new Semantic Mapping dataset, captured with RealSense (SMR), has 4 RGB-D sequences described in III-F. All datasets have semantically labelled meshes, and ScanNet has some frames with 2D semantic labels as well.

For 3D semantic segmentation experiments (Sec IV-A), we include the following baselines: 3D network-based methods SparseConvNet [13], MinkowskiNet [12] and INS-CONV [15], which is an incremental version of SparseConvNet; 2D-3D networks-based methods SVCNN [6], Fusion-aware PointConv (FP-Conv) [7] and ProgressiveFusion (PsF) [20]. For 2D semantic segmentation evaluation we use AdapNet++ [18], FuseNet [42], SSMA [18] and PS-res-excite [43], and our closest baseline MVCNet [10] based on our LPN backbone, in single-view inference (MVCNet-S) and in X-view inference (MVCNet-MX) which we re-implemented. LPN-base is trained in a single-view mode,

TABLE I: 3D Semantic segmentation on ScanNet v2 (Val). SeMLaPS outperforms other state-of-the-art 2D-3D network-based methods.

Method	Type	mIoU	bath	bed	bkskf	cab	chair	cnt	curt	desk	door	floor	other	pic	fridge	shwr	sink	sofa	tbl	toilet	wall	win
<b>3D networks</b>																						
MkNet [12]	offline	72.2	85.9	81.0	<b>80.7</b>	66.0	<b>91.1</b>	63.4	<b>75.9</b>	<b>64.2</b>	61.4	94.8	<b>60.0</b>	29.6	<b>62.0</b>	69.1	66.8	81.7	<b>76.3</b>	92.1	83.3	59.1
SparseConvNet [13]	offline	69.3	86.5	78.6	76.7	61.2	89.0	60.9	70.8	59.1	60.1	94.5	47.6	31.5	47.4	68.9	62.3	83.3	68.3	85.4	82.3	60.0
INS-CONV-m32 [15]	online	71.5	84.3	79.3	78.7	64.4	90.1	63.6	72.5	63.5	60.4	<b>95.2</b>	55.8	33.8	52.3	<b>74.5</b>	63.3	83.2	74.5	92.8	84.0	63.3
INS-CONV-m64 [15]	online	<b>72.4</b>	<b>87.4</b>	<b>81.2</b>	79.6	<b>67.7</b>	91.0	<b>64.5</b>	74.9	60.8	<b>62.1</b>	95.1	57.8	<b>36.0</b>	52.0	72.2	<b>67.0</b>	<b>83.3</b>	72.3	<b>93.3</b>	<b>85.1</b>	<b>65.2</b>
<b>2D-3D networks</b>																						
PsF [20]	online	55.0	65.6	61.2	65.7	48.6	68.4	41.7	54.9	48.9	47.5	87.1	43.7	25.7	41.8	34.5	53.4	59.8	54.0	78.9	70.6	47.0
FPC [7]	online	67.2	85.4	<b>82.3</b>	64.0	60.9	75.1	56.0	64.8	58.2	<b>64.8</b>	91.9	46.4	<b>40.6</b>	<b>64.2</b>	51.7	<b>63.5</b>	<b>77.9</b>	68.9	87.0	83.8	56.3
SVCNN [6]	online	68.3	73.4	78.5	79.1	60.5	80.6	59.3	70.4	59.9	60.5	91.1	57.8	35.0	57.5	75.2	61.3	72.6	64.4	86.4	80.5	61.7
SeMLaPS (ours)	online	<b>72.2</b>	<b>87.7</b>	80.4	<b>82.2</b>	<b>63.0</b>	<b>87.3</b>	<b>66.2</b>	<b>76.3</b>	<b>65.8</b>	63.5	<b>94.6</b>	<b>52.9</b>	36.0	63.7	<b>75.2</b>	63.0	74.7	<b>72.0</b>	<b>91.9</b>	<b>84.0</b>	<b>64.2</b>

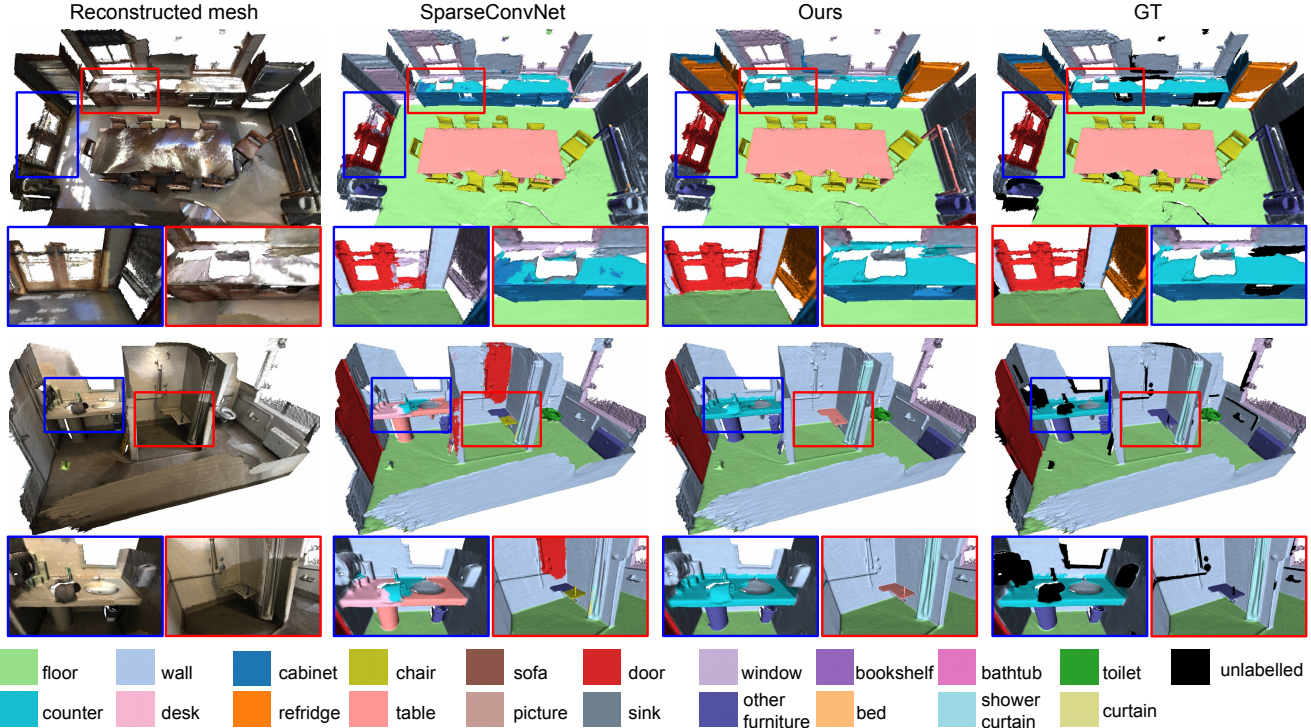


Fig. 5: 3D semantic segmentation results on ScanNet. SeMLaPS achieves better results in classes reliant on visual features, such as refrigerator and counter (see zoomed-ins), and produces smoother and less-fragmented boundaries thanks to our QPOS and SegConvNet.

while LPN-S and LPN-MX are trained using  $N = 3$  views and use one and  $X$  views for inference respectively<sup>1</sup>.

### A. 3D Semantic segmentation

We show quantitative and qualitative 3D semantic segmentation results. As mentioned before, our LPN can have multiple variants, such as using a different number of views at inference time, using depth encoder or not, mini-batch or sequential inference mode. By default, we use LPN-M3 with both RGB and depth encoders in sequential mode for 2D inference. Our SegConvNet is trained with data generated by the same trained LPN-M3 model. If not otherwise stated, we refer to our default method as SeMLaPS.

**ScanNet dataset.** SeMLaPS shows the best results among state-of-the-art 2D-3D networks-based semantic mapping methods, demonstrating big gains in segmenting classes like bathtub, chair or counter, see Tab. I. Our proposed approach is on par with off-line and online incremental 3D networks.

<sup>1</sup>Results are from respective papers apart from re-implemented MVCNet and re-trained SparseConvNet evaluated on SMR.

TABLE II: Online 3D Semantic segmentation on SceneNN.

Method	Type	mAcc
FP-Conv [7]	2D-3D	71.5
SVCNN [6]	2D-3D	76.9
INS-CONV [15]	3D	79.5
Ours	2D-3D	<b>80.0</b>

Fig. 5 shows some qualitative results. From the figure it can be seen that our 2D-3D pipeline performs better in classes that rely more on visual features, such as refrigerator (first row) and counter (last row). Moreover, compared with a pure 3D-based method, our method produces smoother boundaries and fewer floating artefacts thanks to the post-processing of QPOS and SegConvNet.

**SceneNN dataset.** We follow the protocol of SVCNN [6] and INS-CONV [15], using SeMLaPS trained on ScanNet to test the generalization ability. We report average mean accuracy (mAcc %) on the 48 scenes used in SVCNN. As shown in Tab. II, SeMLaPS outperforms all the previous online semantic segmentation methods.

**SMR dataset.** We compare the SparseConvNet 3D network and the proposed SeMLaPS-RGB (still using LPN-M3 but

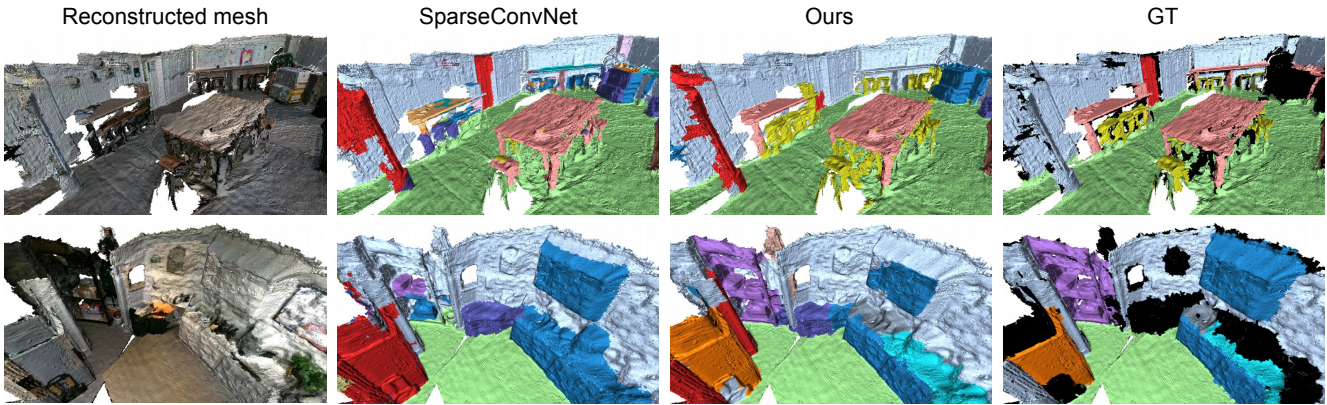


Fig. 6: 3D semantic segmentation on Semantic Mapping with Realsense (SMR) sequences. Our approach is more consistent especially for less frequent classes, demonstrating better cross-sensor generalization than the 3D network.

dropping the depth encoder), both trained on ScanNet, see Tab. III. SeMLaPS-RGB shows significantly better cross-sensor generalization: our mIoU is better by 69%, and mAcc is better by 6%. Our method is better at dealing with less frequent classes, such as refrigerator, counter, chair and window, while performing on par with our baseline on frequent classes. Fig. 6 shows some qualitative results. The reconstruction quality in SMR is much worse than in ScanNet or SceneNN. Apart from the classes already mentioned like refrigerator and counter, our SeMLaPS-RGB also performs much better on thin structures and small objects, like the stools in the first row. It suggests that 2D-3D networks deal with inaccurate depth better than 3D networks.

### B. 2D Semantic segmentation

In this section we evaluate 2D networks on the ScanNet v2 dataset, see Tab. IV. In the single-view RGB-D inference mode, LPN-base has the lowest performance; the MVCNet [10] has better metrics, while our proposed LPN-S achieves the best accuracy. In the multi-view inference mode, we see that our LPN-M3 outperforms MVCNet-M3, both using three views, while increasing the number of views up to 7 improves the LPN results further. Comparing with the other architectures, we see that our LPN backbone has reasonable performance, while relying on computationally efficient MobileNetV3, when other approaches use VGG-16 or ResNet50. LPN-M7 outperforms all single-view baselines.

### C. Over-Segmentation

We run the proposed QPOS and Supervoxel Clustering [6], [27], on ScanNet val. To compute the over-segmentation error (OSE) we find semantic labels for segments (supervoxels) from ground truth vertex labels using majority vote, propagate labels back to vertices, and compute mean accuracy. The under-segmentation error (USE) [25] is a standard error metric. As shown in Tab. V, our QPOS outperforms Supervoxel Clustering by 6.2% in USE and by 2.2% in OSE, producing a similar average number of segments.

### D. Ablation study

We conduct ablation studies to investigate the effectiveness of different components in SeMLaPS.

**Effect of LPN and QPOS.** We first study the effect of Latent Prior Network (LPN) and Quasi-Planar Over-segmentation

(QPOS), see Tab. VI. We show the mIoU after Bayesian fusion as **BF mIoU** and the mIoU of the final output as **Final mIoU**. We start by replacing our default LPN with other 2D network variants. The first row shows that QPOS+SegConvNet improve the results by 7.7 mIoU%, justifying the importance of our proposed 3D post-processing. Our default LPN-M3 is significantly better compared with LPN-base (second row), which indicates the effectiveness of multi-view training and incorporating latent priors. In the third row, the final mIoU drops further to 68.5 % after replacing our LPN-base with SSMA [18]. In all cases, the final result is significantly better than the Bayesian fusion, highlighting the universal applicability of the proposed QPOS+SegConvNet step. Next, in rows 4-6 we replace the proposed QPOS method with Supervoxel clustering [27]. QPOS consistently yields better final mIoU results compared to the baseline.

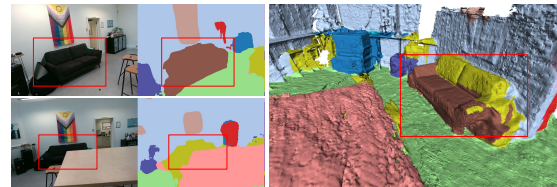


Fig. 7: An example failure case of SeMLaPS. **Top Left:** A good view for LPN, **Bottom-left:** A bad view for LPN. **Right:** Final 3D results. Heavy occlusion causes LPN output wrong labels, which SegConvNet is unable to correct.

**Effect of SegConvNet.** We first analyze the importance of input features to the weight-net  $W(\cdot)$  and input segment feature. We train our SegConvNet with data generated by our default LPN-M3 variant, but removing  $\phi_{xy}$  from the weight-net input and the 9-D geometric features from the input segment feature. As shown in Tab. VII, both VI-features and the directional variance are important for the weight-net to predict better convolution weights. Also it can be seen that 2D-only features are not descriptive enough to achieve the best result (70.1 vs. 72.2). We also studied the effect of  $K$ . It can be seen that increasing  $K$  from 16 to 64 constantly improves the result, but the improvement saturates when further increasing  $K$  to 96.

**Failure Case.** A major limitation of SeMLaPS is that it heavily relies on the 2D semantic output, which could easily

TABLE III: 3D Semantic segmentation results on Semantic Mapping with RealSense (SMR) dataset.

Method	mAcc	mIoU	bkskf	cab	chair	cntr	desk	door	floor	other	pic	fridge	sink	sofa	tbl	wall	win
SparseConvNet [13]	71.4	26.5	22.19	18.15	37.88	0.00	2.75	34.15	<b>91.91</b>	0.00	0.00	0.00	0.00	<b>82.89</b>	40.81	60.33	6.43
SeMLaPS-RGB	<b>75.3</b>	<b>44.7</b>	<b>30.92</b>	<b>41.13</b>	<b>53.69</b>	<b>81.62</b>	0.09	<b>41.70</b>	87.73	0.00	0.00	<b>82.86</b>	<b>73.70</b>	40.29	<b>54.08</b>	<b>62.98</b>	<b>20.01</b>

TABLE IV: Performance of different 2D semantic segmentation models on ScanNet v2 (Val). Best result in bold.

Method	Backbone	Test views	mIoU
<b>Single-view trained</b>			
AdapNet++ [18]	ResNet-50	1	54.6
FuseNet [42]	VGG-16	1	56.7
SSMA [18]	ResNet-50	1	66.1
PS-res-excite [43]	ResNet-50	1	<b>69.8</b>
LPN-base	MobileNetV3	1	65.1
<b>Multi-view trained</b>			
LPN-S	MobileNetV3	1	66.5
MVCNet-S	MobileNetV3	1	66.2
LPN-M3	MobileNetV3	3	69.0
MVCNet-M3	MobileNetV3	3	68.6
LPN-M5	MobileNetV3	5	69.7
LPN-M7	MobileNetV3	7	<b>70.1</b>

TABLE V: 3D Over-Segmentation Evaluation on ScanNet v2 (Val).

Method	US Err. %	OS Err. %	Avg Segments	Avg Seg. Size	Std Seg. Size
Supervoxel [27]	37.8	5.6	1760	90	39
QPOS	<b>31.6</b>	<b>3.4</b>	1760	90	30

fail under difficult situations such as heavy occlusion and extreme motion, see Fig. 7. In sequence shown in Fig. 7, the black sofa (red box) is severely occluded by the table in 920 out of 1420 frames in which it is observed, which caused the LPN to wrongly label it as chair (yellow) in those frames. Although one of the motivations of LPN is to mitigate this kind of situation by leveraging multi-view consistency, this sequence proves too difficult to handle. We believe that adopting better view selection and temporal fusion strategy should help solve this issue.

### E. System Timing

To evaluate the run-time performance of SeMLaPS, following [5], [15] we profile the timing for each stage of our method on scene0645\_01, a typical large scene in ScanNetv2. All experiments are conducted on a desktop computer with Intel Core-i7 12700K CPU and a Nvidia RTX-3090ti GPU. As SeMLaPS consists of 4 stages: LPN, Bayesian fusion, QPOS and SegConvNet, we report the average timing for each of the 4 stages in VIII. We also report the run-time breakdown for our LPN.

As shown in Tab. VIII (a), our default LPN-M3 model could run at 50 Hz. Thanks to our lightweight encoder backbone and sequential operating mode, the feature extraction (encoder forward pass) and decoder forward pass only take 7 ms and 1.7 ms respectively. The feature re-projection stage presents as a major bottleneck which requires 11.8 ms for processing feature maps from the other two neighboring views, but it could be optimized by replacing PyTorch3D with a better implementation. Tab. VIII (b) shows timing for all stages. SeMLaPS achieves 25 Hz performance, which

TABLE VI: Ablation study on the effect of different modules on ScanNet v2 (Val). We show mIoU (%) after each stage.

2D Net	OS	BF mIoU %	Final mIoU %
LPN-M3	QPOS	<b>64.5</b>	<b>72.2</b>
LPN-base	QPOS	63.5	70.7
SSMA [18]	QPOS	59.1	68.5
LPN-M3	SV [27]	<b>64.5</b>	70.2
LPN-base	SV [27]	63.5	69.0
SSMA [18]	SV [27]	59.1	67.0

TABLE VII: Ablation study on the effect of SegConvNet. All networks are trained with data generated with our default LPN-M3.

Model	mIoU %
SegConvNet w/o VI-feature	71.0
SegConvNet w/o dir. variance	71.5
SegConvNet w/o geom. feature	70.1
SegConvNet FULL (K=64)	<b>72.2</b>
SegConvNet K=16	71.2
SegConvNet K=32	71.8
SegConvNet K=96	72.1

TABLE VIII: Runtime (ms) of (a) LPN-M3; (b) whole SeMLaPS.

(a) LPN-M3 run-time breakdown		(b) SeMLaPS system run-time	
Stage	Time (ms)	Stage	Time (ms)
Encoder	7.0	LPN	20.5
Feature re-proj.	11.8	Bayesian fusion	0.4
Decoder	1.7	QPOS	11.6
Overall	20.5	SegConvNet	4.6

is more than enough for online real-time semantic mapping with our visual inertial SLAM system.

## V. CONCLUSION

We presented SeMLaPS, a real-time online semantic mapping system that follows a 2D-3D pipeline. It benefits from fusing latent features from historical views using a novel Latent Prior Network (LPN), while Quasi-Planar-Over-Segmentation (QPOS) and Segment-Convolutional Network (SegConvNet) improve the final results to the same level as 3D offline methods while still maintaining real-time performance. SeMLaPS achieves better cross-sensor generalization than 3D-only networks.

Currently, SeMLaPS cannot handle environments with frequent changes or dynamic elements. Future work could leverage the 2D-3D design to skip 2D dynamic features in the re-projection and global 3D map. Alternatively optical or scene flow field could be used for differentiable re-projection.

## REFERENCES

- [1] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2017, pp. 4628–4635.
- [2] Y. Ren, B. Xu, C. L. Choi, and S. Leutenegger, "Visual-inertial multi-instance dynamic slam with object-level relocalisation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 11 055–11 062.

- [3] Y. Tian, Y. Chang, F. H. Arias, C. Nieto-Granda, J. P. How, and L. Carlone, "Kimera-multi: Robust, distributed, dense metric-semantic slam for multi-robot systems," *IEEE Trans. Robotics*, vol. 38, no. 4, 2022.
- [4] Y. Kantaros, S. Kalluraya, Q. Jin, and G. J. Pappas, "Perception-based temporal logic planning in uncertain semantic maps," *IEEE Trans. Robotics*, vol. 38, no. 4, pp. 2536–2556, 2022.
- [5] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji, "Panopticfusion: Online volumetric semantic mapping at the level of stuff and things," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4205–4212.
- [6] S.-S. Huang, Z.-Y. Ma, T.-J. Mu, H. Fu, and S.-M. Hu, "Supervoxel convolution for online 3d semantic segmentation," *ACM Trans. Graph.*, vol. 40, no. 3, pp. 1–15, 2021.
- [7] J. Zhang, C. Zhu, L. Zheng, and K. Xu, "Fusion-aware point convolution for online semantic 3d scene segmentation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 4534–4543.
- [8] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto, "Volumetric instance-aware semantic mapping and 3D object discovery," (*IEEE*) *Robotics and Automation Letters*, vol. 4, no. 3, pp. 3037–3044, 2019.
- [9] Y. Xiang and D. Fox, "DA-RNN: Semantic mapping with data associated recurrent neural networks," in *Robotics: Science and Systems (RSS)*, 2017.
- [10] L. Ma, J. Stückler, C. Kerl, and D. Cremers, "Multi-view deep learning for consistent semantic mapping with RGB-D cameras," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 598–605.
- [11] L. Han, T. Zheng, L. Xu, and L. Fang, "Occuseg: Occupancy-aware 3d instance segmentation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2940–2949.
- [12] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3075–3084.
- [13] B. Graham, M. Engelcke, and L. Van Der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 9224–9232.
- [14] B. Bescos, C. Campos, J. D. Tardós, and J. Neira, "DynaSLAM ii: Tightly-coupled multi-object tracking and slam," (*IEEE*) *Robotics and Automation Letters*, vol. 6, no. 3, pp. 5191–5198, 2021.
- [15] L. Liu, T. Zheng, Y.-J. Lin, K. Ni, and L. Fang, "INS-Conv: Incremental sparse convolution for online 3d segmentation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 18975–18984.
- [16] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3d reconstructions of indoor scenes," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5828–5839.
- [17] L. Keselman, J. Iselin Woodfill, A. Grunnet-Jepsen, and A. Bhowmik, "Intel realsense stereoscopic depth cameras," in *IEEE Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 1–10.
- [18] A. Valada, R. Mohan, and W. Burgard, "Self-supervised model adaptation for multimodal semantic segmentation," *Intl. J. of Computer Vision*, vol. 128, no. 5, pp. 1239–1285, 2020.
- [19] T. Cavallari and L. Di Stefano, "Semanticfusion: Joint labeling, tracking and mapping," in *Eur. Conf. on Computer Vision (ECCV)*, 2016, pp. 648–664.
- [20] Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, "Real-time progressive 3d semantic segmentation for indoor scenes," in *IEEE Workshop on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1089–1098.
- [21] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep convolutional networks on 3d point clouds," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9621–9630.
- [22] A. P. Moore, S. J. Prince, J. Warrell, U. Mohammed, and G. Jones, "Superpixel lattices," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2008, pp. 1–8.
- [23] Y. Zhou, L. Ju, and S. Wang, "Multiscale superpixels and supervoxels based on hierarchical edge-weighted centroidal voronoi tessellation," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 3834–3845, 2015.
- [24] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Intl. J. of Computer Vision*, vol. 59, pp. 167–181, 2004.
- [25] J. Papon, A. Abramov, M. Schoeler, and F. Worgotter, "Voxel cloud connectivity segmentation-supervoxels for point clouds," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 2027–2034.
- [26] K. Tateno, F. Tombari, and N. Navab, "Real-time and scalable incremental segmentation on dense slam," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 4465–4472.
- [27] Y. Lin, C. Wang, D. Zhai, W. Li, and J. Li, "Toward better boundary preserved supervoxel segmentation for 3d point clouds," *ISPRS journal of photogrammetry and remote sensing*, vol. 143, pp. 39–47, 2018.
- [28] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.
- [29] H. Thomas, C. R. Qi, J.-E. Deschard, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6411–6420.
- [30] W. Hu, H. Zhao, L. Jiang, J. Jia, and T.-T. Wong, "Bidirectional projection network for cross dimension scene understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14373–14382.
- [31] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual-inertial, and multi-map slam," *IEEE Trans. Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.
- [32] N. Funk, J. Tarrío, S. Papatheodorou, M. Popović, P. F. Alcantarilla, and S. Leutenegger, "Multi-resolution 3d mapping with explicit free space representation for fast and accurate mobile robot motion planning," (*IEEE*) *Robotics and Automation Letters*, vol. 6, no. 2, pp. 3553–3560, 2021.
- [33] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.
- [34] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.
- [35] N. Ravi, J. Reizenstein, D. Novotny, T. Gordon, W.-Y. Lo, J. Johnson, and G. Gkioxari, "Accelerating 3d deep learning with PyTorch3D," *arXiv:2007.08501*, 2020.
- [36] A. Karpathy, S. Miller, and L. Fei-Fei, "Object discovery in 3d scenes via shape analysis," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2013, pp. 2088–2095.
- [37] X. Li, W. Wu, X. Z. Fern, and L. Fuxin, "Improving the robustness of point convolution on k-nearest neighbor neighborhoods with a viewpoint-invariant coordinate transform," in *IEEE Workshop on Applications of Computer Vision (WACV)*, 2023, pp. 1287–1297.
- [38] B.-S. Hua, Q.-H. Pham, D. T. Nguyen, M.-K. Tran, L.-F. Yu, and S.-K. Yeung, "Scenenn: A scene meshes dataset with annotations," in *2016 fourth international conference on 3D vision (3DV)*. Ieee, 2016, pp. 92–101.
- [39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [40] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration," *TOG*, vol. 36, no. 4, p. 1, 2017.
- [41] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5556–5565.
- [42] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, "Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture," in *Asian Conference on Computer Vision (ACCV)*, November 2016.
- [43] M. Sodano, F. Magistri, T. Guadagnino, J. Behley, and C. Stachniss, "Robust double-encoder network for rgb-d panoptic segmentation," *arXiv preprint arXiv:2210.02834*, 2022.