Technische Universität München
Chair of Media Technology
Prof. Dr.-Ing. Eckehard Steinbach

Forschungspraxis

Leveraging Point Cloud Completion for Point Cloud-based Place Recognition

Author:	Xiaoang Zhang
Matriculation Number:	03743260
Address:	Goerrestrasse 9 80798 Munich
Advisor:	Adam Misik
Begin:	19.12.2022
End:	04.04.2023

With my signature below, I assert that the work in this thesis has been composed by myself independently and no source materials or aids other than those mentioned in the thesis have been used.

München, April 4, 2023

Place, Date

Zhang

Signature

This work is licensed under the Creative Commons Attribution 3.0 Germany License. To view a copy of the license, visit http://creativecommons.org/licenses/by/3.0/de

Or

Send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.

München, April 4, 2023

Place, Date

Zhang

Signature

Abstract

Place recognition has become a relevant research topic in several applications such as visual SLAM and augmented reality. In recent years, a large number of approaches have been proposed to tackle the place recognition problem, including matching of global descriptors from a large database. Among these approaches, point cloud-based methods have gained significant popularity because they are generally more robust than image-based methods. However, raw point cloud data captured by RGB-D cameras or LiDAR is usually sparse and occluded in large-scale scenes, which can result in inaccurate submap retrieval. We argue that a possible approach to tackle the high occlusion degree in the raw point clouds is by semantic point cloud completion. With the point cloud completion approach, missing geometry can be restored to provide more potentially useful information for constructing global descriptors. In this study, we aim to investigate whether point cloud completion can enhance the performance of place recognition using established methods for both scene completion and place recognition. Our experiment uses the sequential real-world LiDAR dataset, SemanticKITTI, which is widely used for autonomous driving research. Our goal is to evaluate the effectiveness of point cloud completion in improving the accuracy of submap retrieval in challenging large-scale scenes. Based on our experiments with the SemanticKITTI dataset, we have observed that when employing PointNetVLAD global descriptors for submap retrieval, the improvement by utilizing completed scenes appears to be insignificant. Our findings suggest that a more sophisticated approach is required to fully harness the potential of completed point clouds.

Contents

Contents ii

1 Introduction 1
- 1.1 Motivation and Goal . . . 1
- 1.2 Structure of the Thesis . . . 3

2 Related Work 5
- 2.1 Point Cloud Based Place Recognition . . . 5
- 2.2 Point Cloud Completion and Semantic Scene Completion . . . 6

3 Methodology 9
- 3.1 JS3C-Net: Semantic Scene Completion . . . 10
  - 3.1.1 Semantic Segmentation of Input Sparse Point Cloud . . . 10
  - 3.1.2 Scene Completion . . . 11
  - 3.1.3 Metric Training . . . 11
- 3.2 Devoxelization of Voxel Grid . . . 12
- 3.3 PointNetVLAD: Global Feature Extractor . . . 12
- 3.4 Evaluation Metrics . . . 14
  - 3.4.1 Precision and Recall . . . 14
  - 3.4.2 Wasserstein Distance . . . 15

4 Experiment 16
- 4.1 Dataset . . . 16
  - 4.1.1 Properties of SemanticKITTI . . . 16
  - 4.1.2 Dataset Preparation . . . 17
- 4.2 Implementation Details . . . 19
  - 4.2.1 Utilization of Pre-trained Models . . . 19
  - 4.2.2 Generation of Global Descriptors . . . 19
- 4.3 Results and Comparison . . . 19
  - 4.3.1 Submap Retrieval with Completed Scene . . . 19
  - 4.3.2 Submap Retrieval using Ground Truth Completed Scene . . . 22
  - 4.3.3 Submap Retrieval with Semantic Labels . . . 24

4.3.4 Discussion 25

5 Conclusion and Future Work **26**

List of Figures **27**

List of Tables **29**

Bibliography **30**

Chapter 1

Introduction

1.1 Motivation and Goal

With the rapid advancements in computer vision, visual information can now play an essential role in solving complex problems, such as place recognition[UL, KWT, KYZ^+, KWT21, LZS^+, SWZ^+, LKZ^+] and pose estimation[SWZ^+]. Place recognition typically precedes the 6 degrees of freedom (6 DoF) pose estimation of autonomous robots and vehicles. Moreover, it can detect loop closure to avoid cumulative odometry drift error in visual SLAM[KYZ^+, LKZ^+]. The goal of this approach is to facilitate navigation for autonomous vehicles and robots through the use of a database containing global descriptors that represent smaller regions within the entire map. These smaller regions are often referred to as submaps[UL]. They typically cover a much more narrow range, usually only up to tens of meters, in comparison to the entire environment.

In recent years, point cloud-based place recognition and localization have become increasingly important research topic[UL, LKZ^+]. Unlike RGB image-based approaches that rely on visual features susceptible to environmental changes, point cloud-based methods rely solely on the geometric structures of a scene, making them more resilient to changing weather conditions like rain or bright sunlight[UL]. This is particularly advantageous for real-world applications where a scene's appearance may look significantly different depending on the time of day or season.

Over the years, several datasets have been created and shared to tackle point cloud-based place recognition, such as the Oxford robotcar dataset[MPLN17]. Since then, numerous pipelines have been developed to extract global descriptors for point cloud submaps, allowing for discriminative retrieval within a larger environment[UL, LZS^+, XXL^+, KWT, KWT21]. While point clouds offer several advantages over other types of sensor data, they can also be very noisy and sparse, particularly in outdoor scenarios of the real world. One approach to address the sparsity is through point cloud completion.

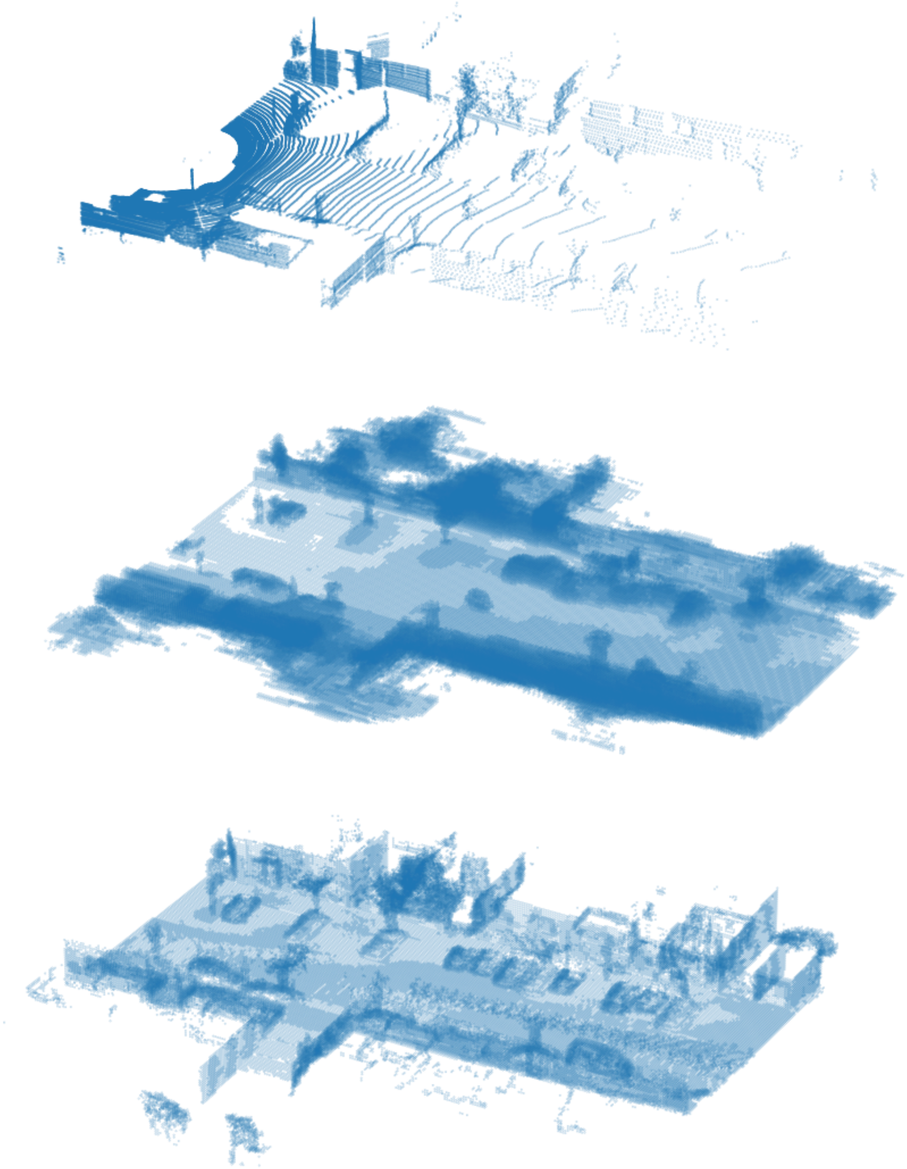

Figure 1.1: Comparison of a raw scan and its corresponding completed scene from SemanticKITTI. From top to bottom: raw scan, predicted completed scene, ground truth completed scene.

Most existing point cloud completion approaches are designed to complete a single object instead of a full environment or large-scale scene[YKH+18, PCC+]. Additionally, the unstructured and unordered nature of point clouds also presents the challenge of requiring the use of permutation-invariant feature extractors[CSKG, UL, YKH+18]. This property ensures that the order of points does not affect the result of encoding[UL, YKH+18] and the subsequent completion. In 2017, PointNet feature extractor has been introduced, which satisfies this requirement[CSKG]. Since then, it has been exploited a lot by many further works, such as PF-Net[HYX$^+$] and CRN[WAL]. Recently, PointNet was employed to encode the point-wise features of the large-scale scenes by PointNetVLAD[UL], which is one of the earliest learning-based methods for encoding large-scale point clouds that show promising performance in generating discriminative global descriptors. The interest of this work lies in, whether the down-sampled completed scene leads to more robust and discriminative global descriptors than the sparse and occluded raw scans. In our further experiments, we leverage the point cloud completion and the predicted semantic labels to investigate the place recognition improvements from semantic distribution.

1.2 Structure of the Thesis

First, in the related work section, we present state-of-art methods for point cloud-based place recognition, followed by an introduction to some of the latest approaches for point cloud completion and semantic scene completion. Regarding the complexity of the problems, the semantic scene completion is quite different from the point cloud completion of a single object. To provide more context for this work, we will also present a performance comparison of the of the state of the art for the semantic scene completion challenge on the SemanticKITTI dataset[BGM$^+$]. This comparison will enable us to have an overview of the state-of-the-art methods in the field and identify the strengths and weaknesses of existing approaches.

The methodology section will provide a comprehensive overview of the methods and components used for place recognition and semantic scene completion. Chapter 3 will first elaborate on the primary research goal of our work, followed by the presentation of the established methods employed for our experiments: JS3C-Net[YGL$^+$21] and PointNetVLAD[UL]. There we will review the essential components of their network architecture. Since these two networks were developed separately, an additional de-voxelization module needs to be implemented to cascade JS3C-Net[YGL$^+$21] with PointNetVLAD[UL].

In chapter 4, we outline the experimental settings for the submap retrieval leveraging scene completion and semantics. To begin with, the utilized dataset in the experiment will be introduced with a detailed description of its properties. Since the semantic scene completion task necessitates the prediction of semantic labels for each point, having an overview of the distribution of each category in the submap is also important.

In the second section of Chapter 4, we demonstrate the implementation details of the Python tools and deep learning frameworks used in our experiments. Then we will introduce the de-voxelization module, which is the essential component for combining JS3C-Net with PointNetVLAD. Since model training can be time-consuming there will be no training included in our experiment. Instead, the publicly available models will be used[KYZ$^+$, YGL$^+$21]. The evaluation and comparison will be shown with qualitative plots as well as quantitative metrics. Through this comparison, we aim to assess the effectiveness of the semantic scene completion and provide insights into its strengths and limitations in aiding to place recognition task. By presenting our findings in a clear and organized manner, we hope to contribute to the broader body of research in the field of point cloud-based place recognition and provide a valuable resource for future studies.

Chapter 2

Related Work

2.1 Point Cloud Based Place Recognition

Place recognition is the problem of determining the approximate position of an agent[UL, KWT21]. From the last decade up to now, deep learning has emerged as the most effective and relevant approach in computer vision, particularly after the wide exploitation of convolutional neural networks (CNNs) and residual neural networks (RNNs). When dealing with 3D problems involving point clouds, convolution operations are not as easily applicable as with 2D images[UL]. For place recognition problems involving 3D representations, some studies initially recommended using handcrafted features, e.g. interest point extraction[SRKB10] or surface matching[HKK$^+$]. The PointNet from 2017 was the first learning-based method that makes extraction of permutation invariant point-wise features possible[CSKG, UL]. Consequently in 2018, Mikaela Angelina Uy et al. introduced their approach, which combines the PointNet with NetVLAD that can be trained end-to-end to extract discriminative global descriptor for large-scale point clouds[UL, CSKG, AGT$^+$]. Moreover, they proposed in their work the âlazy triplet and quadrupletâ loss functions, which are essential tricks for effective metric training[UL]. Their network was trained and tested on the Oxford Robotcar dataset, where the generality of the global descriptors was also verified. An apparent limitation of this approach is that, while it extracts local features using PointNet, it only aggregates the outputs of the final layer of PointNet during the construction of the global descriptor by NetVLAD. This approach overlooks the potential benefits of incorporating local features of lower levels in the aggregation process, which may yield more informative and discriminative global descriptors.

There are already some recent works focusing on utilizing lower-level features to construct global descriptors. For example, the MinkLoc3D[KWT] used voxelized point cloud and applied sparse 3D convolution, whose performance also reached the state of the art. In fact, MinkLoc3D has outperformed PointNetVLAD with a large margin. This success highlights the importance of exploitation of lower-level features. Another example from recent publication is EgoNN that achieves good performance also in pose estimation[KWT21]. Similar to MinkLoc3D they also voxelized the point cloud in order to use convolution operations to extract local features. Since most of the later works treated PointNetVLAD as a baseline approach in performance comparison, PointNetVLAD is also proposed in the pipeline of this study for place recognition.

Another way of aggregating the point-wise local features more effectively would be using the self-attention module after the encoder part of the network. For example, SOE-Net has presented an approach to concatenate a self-attention unit to the PointNet feature extractor[XXL+]. This mechanism helps to explore the long-range contextual dependencies between points and therefore the more detailed geometrical information can be aggregated to the global feature descriptor[XXL+]. This approach has the additional advantage of not requiring voxelization steps, making it computationally more efficient compared to methods that discretize the point cloud. According to their experiment, the SOE-Net is able to outperform PointNetVLAD on multiple benchmark datasets, including the Oxford robotcar[XXL+].

2.2 Point Cloud Completion and Semantic Scene Completion

Similar to PointNetVLAD, the development of the point cloud completion pipeline has also benefited a lot from the PointNet feature extractor. Recent methods have demonstrated the ability to reconstruct local details while avoiding the generation of dissimilar global shapes compared to the ground truth[HYX+, WAL]. However, they mainly focus on completing the point cloud of one single object instead of large-scale LiDAR scans from the outdoor scenes. The completion problem of large-scale point cloud scenes is often associated with semantic segmentation. Jens Behley et al. have published a dataset named SemanticKITTI along with several challenging tasks for the community, such as semantic segmentation, moving object segmentation, and semantic scene completion[BGM+]. An important point to note is that the ground truth of their completed scene has been labeled for voxel grids instead of point clouds. This is maybe because at the time of their project beginning there were still not that many successful works for point cloud completion and convolutional neural network was extremely relevant in the computer vision research field. As a result, the participants of the SemanticKITTI challenges have developed their methods by working with voxel grids instead of point clouds.

Among all the challenges introduced by SemanticKITTI, the semantic scene completion is probably the most difficult and complicated one according to the mean IoU scores in the following ranking lists.

According to Table 2.1, the best approach for semantic scene completion has also only achieved an average IoU of 29.5%. Although it showed a significant improvement in label prediction accuracy for the rarer categories such as moving cars, bicycles, and persons

Table 2.1: SemanticKITTI[BGM+] Challenge Rankings. The JS3C-Net in bold is the network employed in this study.

Task	Rank	Method	mIoU
Semantic Segmentation	1	2DPASS	72.9
	2	Point-Voxel-KD	71.2
	3	RPVNet	70.3
	4	AF2S3Net	69.7
	5	Cylinder3D	67.8
	6	SPVNAS	66.4
	7	**JS3C-Net**[YGL+21]	**66.0**
	8	AMVNet	65.3
	9	Lite-HDSeg	63.8
	10	TORNADONet	63.1
Semantic Scene Completion	1	S3CNet[CAR+21]	29.5
	2	**JS3C-Net**[YGL+21]	**23.8**
	3	Local-DIFs	22.7
	4	LMSCNet	17.0

Table 2.2: Scene Completion Ranking on SemanticKITTI[BGM+]. The JS3C-Net in bold is the network employed in this study.

Rank	Method	IoU
1	Local-DIFs	57.7
2	LMSCNet-SS	56.7
3	**JS3C-Net**[YGL+21]	**56.6**
4	LMSCNet	55.3
5	TS3D+DNet+SATNet	50.6
6	SSCNet-full	50.0
7	S3CNet	45.6

compared to the second-best one, the key to this performance boost is regrettably not elaborated in their publication[CAR+21]. Since the JS3C-Net[YGL+21] has demonstrated superior performance in pure scene completion compared to the current state-of-the-art method and its source code is publicly available, we decide to utilize the JS3C-Net as the component for generating completed scenes in our pipeline.

The JS3C-Net ranked in the 2nd place for semantic scene completion and 7th for semantic segmentation, which can be checked in Table 2.1. It is worth noting that although the semantic segmentation of the raw point cloud can achieve an average Intersection over Union (IoU) of 66.0%, the mean IoU of semantic scene completion is relatively low. In particular, smaller and rarer categories in the dataset tend to suffer from significantly lower prediction accuracy[YGL+21, CAR+21]. However, in the case of purely scene completion, the IoU can actually reach up to 56.6% on the SemanticKITTI benchmark. This suggests that scene completion may offer more accurate information for enhancing place recognition compared to semantic scene completion. Even though the predicted semantic labels of the completed scene are mostly incorrect (see Figure 2.1), their distribution can still be beneficial for uniquely identifying sub-maps. Therefore an experiment was conducted in this work to confirm if the label distribution in a sub-map is actually useful. The findings are demonstrated in Chapter 4.

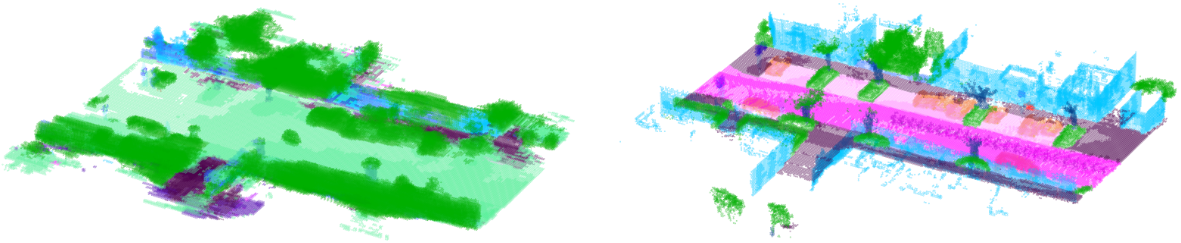

Figure 2.1: Left: predicted semantic scene completion obtained by JS3C-Net. Right: ground truth completed scene obtained by point cloud registration. Both submaps represent the first LiDAR frame of sequence 00 from SemanticKITTI. The submaps illustrate the region of interest before the driving direction.

As for the point cloud completion without any voxel grids as an intermediate representation, several recent methods, including those employing unsupervised learning, can already successfully reconstruct local details without significantly altering the object's global shape[HYX^{+}, WAL, CLZ^{+}22]. However, most of those methods have not yet been applied to the scene completion of large-scale point clouds. The existing methods dealing with large-scale semantic scene completion are still using voxelized point clouds. When looking at the semantic masks shown in Figure 2.1 on the left side, we can see that even the state-of-art method incorporating 3D convolution lacks the ability to reconstruct local details. Regarding the recent development of object-level point cloud completion, the potential of directly using point cloud completion for large-scale scenes without voxelization will be discussed in the conclusion chapter.

Chapter 3

Methodology

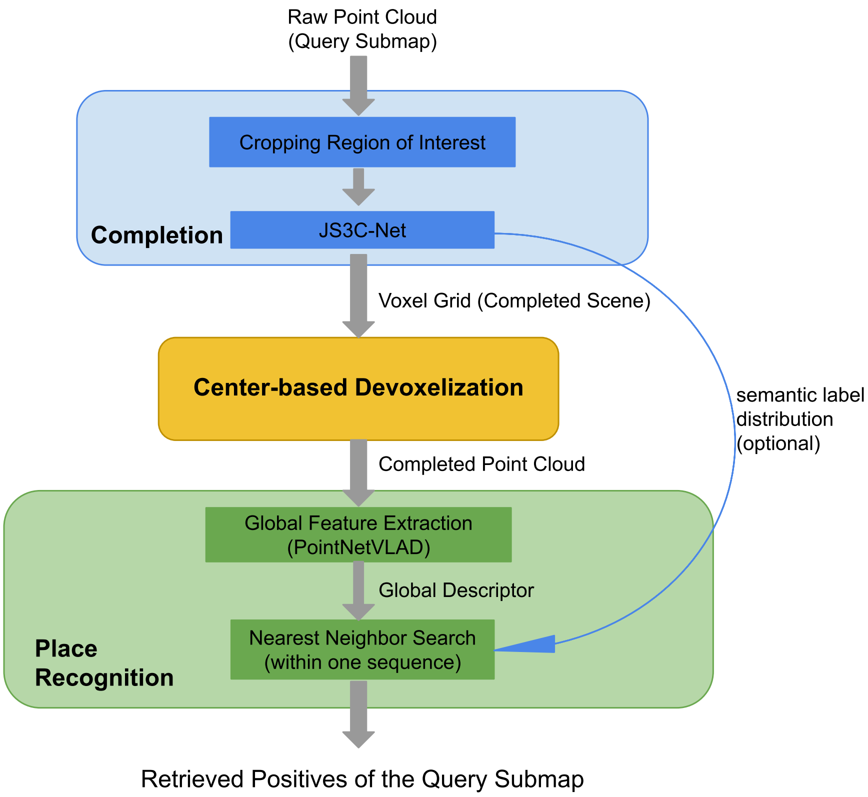

Figure 3.1: Overall pipeline for submap retrieval in this work.

The primary objective of this study is to examine whether completed point clouds can aid in extracting more discriminative global descriptors using PointNetVLAD. Moreover, the

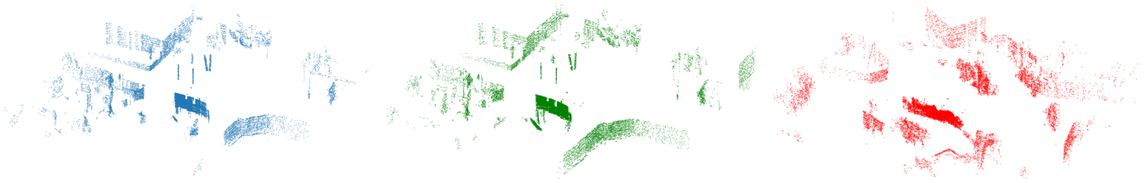

Figure 3.2: Left: anchor submap, which is the query sub-map. Middle: positive sub-map of the anchor. Right: negative submap of the anchor. Note that the hereby presented negative was erroneously retrieved as positive by PointNetVLAD[UL]. The submaps shown here are not completed.

semantic labels of the completed scene may also serve as an extra feature to differentiate the query sub-map from its negatives in the database. The baseline consists of two main modules: the semantic scene completion and the global descriptor extractor.

3.1 JS3C-Net: Semantic Scene Completion

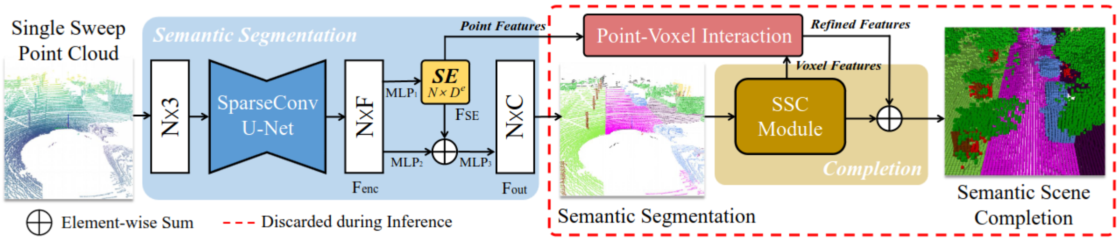

Figure 3.3: Network architechture of JS3C-Net.[YGL+21]

Xu Yan et al. developed the JS3C-Net, a neural network architecture designed for point cloud semantic segmentation and semantic scene completion[YGL+21].

3.1.1 Semantic Segmentation of Input Sparse Point Cloud

The first module of JS3C-Net performs semantic segmentation on an incomplete point cloud by voxelizing it and using SparseConv U-Net[YGL+21]. Non-empty voxels are stored using a Hash table for efficient convolution. The resulting features are converted to point-wise features through a nearest-neighbor operation. Then the point-wise features are embedded by a Multi-layer Perceptron (MLP) into another latent space for shape priors to be used in the point-voxel interaction module for scene completion. The shape embedding module is trained during the joint training of semantic segmentation and scene completion, without excessive manual support.

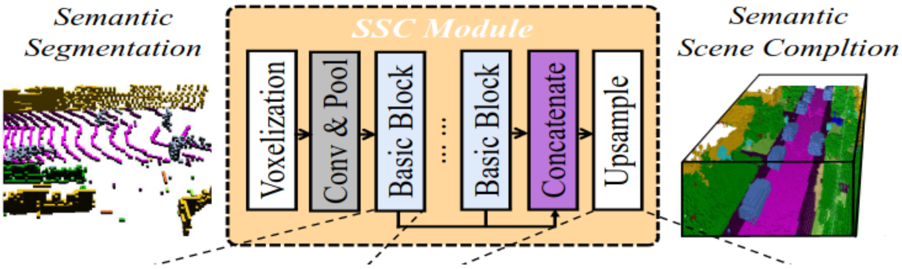

Figure 3.4: Insight of the semantic scene completion module.[YGL+21]

3.1.2 Scene Completion

In Figure 3.4, the structural details of the Semantic Scene Completion(SSC) module are depicted. Instead of giving only raw point cloud as the input, each point is labeled with semantic probability F_{out} predicted in the previous step and also voxelized. Because of the voxelization, the storage requirement, as well as computational complexity, become significantly larger. The voxel grid should first be down-sampled by convolution and pooling operation. Next, it will be passed to the basic blocks, which consist of convolution operations and skip connections. These basic blocks are responsible for learning the local geometries of the voxel grid. Afterward, the features from different scales are aggregated by the concatenation operation. As the dilated convolution can lead to interpolation inaccuracy for up-sampling, the JS3C-Net utilized dense up-sampling to achieve the original resolution of the output from the semantic scene completion[YGL+21]. The final output is a voxel grid with labels of objects and non-objects.

At this point, the scene completion module could already be considered finished. However, due to discretization, the predicted complete voxel grid suffers from a relatively coarse completion[YGL+21]. In order to address this issue, the Point-voxel Interaction(PVI) module is utilized for refining the coarse completion[YGL+21]. As a result, the semantic scene completion network is able to leverage the context provided by the previous module. This is one of the most relevant contributions of JS3C-Net, which enables it to construct a relatively detailed completed scene.

3.1.3 Metric Training

Another important contribution of JS3C-Net is the exploitation of an uncertainty weighting method that automatically adjusts the weights between the segmentation and scene completion losses. The loss function of the joint training is formulated as

$$L(W, \sigma_1, \sigma_2) = \frac{1}{2\sigma_1} L_{seg}(W_{seg}) + \frac{1}{2\sigma_2} L_{complet}(W) + log(\sigma_1) + log(\sigma_2)$$

Here we can clearly identify the two training objectives of JS3C-Net. The weights of both training losses σ_1 and σ_2 are trainable variables, which weigh the two training objectives for the optimal trade-off. The loss function L utilized here is cross entropy.

3.2 Devoxelization of Voxel Grid

Once the completed voxel grid is acquired, the scene must be converted into a point cloud to become a valid input for PointNetVLAD[UL]. To this end, we extract the center point of non-empty voxels to construct the new point cloud, which is very dense as can be seen in the following picture.

3.3 PointNetVLAD: Global Feature Extractor

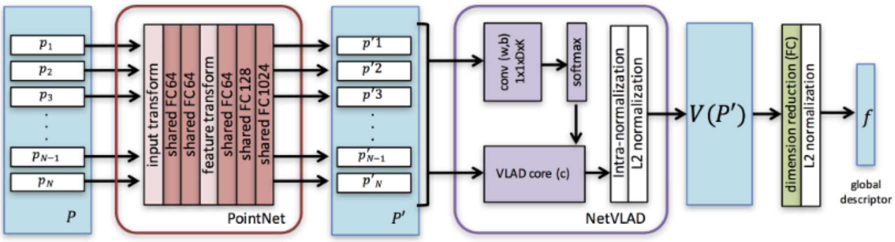

Figure 3.5: Network architechture of PointNetVLAD.[UL]

The second stage of the experiment is to apply a global feature extractor to the point cloud submaps and then calculate the distances between the query and the descriptors stored in the database. This module enables retrieving the coarse position of the LIDAR frame. We choose PointNetVLAD because it has become a baseline method in many comparisons[UL, XXL+, LKZ+, KYZ+]. The point clouds P_V obtained in the de-voxelization step are down-sampled to the same number of points in order to be applied to PointNetVLAD. Compared to JS3C-Net, PointNetVLAD employs a relatively simple network architecture, which is illustrated in Figure 3.5:

The PointNetVLAD[UL] is composed of two main sections: the first section utilizes the PointNet module to extract local features[CSKG], and the second section uses NetVlAD for feature aggregation[UL]. The PointNet takes the point set $P_V = \{p_1^V, ..., p_N^V | p_n^V \in R^3\}$ as input and then uses multiple layers of shared MLPs to embed the points into feature vectors $P'_V = \{{p'_1}^V, ..., {p'_N}^V | {p'_n}^V \in R^D\}$. The dimension of those feature vectors is typically much higher than their coordinates. The most essential part of this architecture is the symmetric max pooling function that makes its output latent vectors invariant to the order of the given point set. Afterward, the feature vectors are passed to the NetVLAD section to be aggregated[UL]. The aggregation works in the following way: Firstly, K cluster centers should be learned by the NetVlad section. In each cluster, the local features vectors are aggregated into a vector $V_k(P') \in R^D$, which is calculated as:

$$V_k(P') = \sum_{i=1}^{N} \frac{e^{w_k^T p'_i + b_k}}{\sum_{k'} e^{w_k^T p'_i + b_k}} (p'_i - c_k)$$

. Consequently, the NetVLAD layer outputs a high-dimensional vector consisting of K aggregated representations.:

$$V(P') = [V_1(P'), ..., V_K(P')]$$

Although $V(P')$ could already be considered as a global descriptor of the given point set P_V it has such a high dimensionality that makes it infeasible for efficient retrieval by nearest neighbor search. To alleviate this problem, the high-dimensional global descriptor should be mapped to the lower-dimensional descriptor by the fully connected layers.

Before starting with the training, each submap is assigned a triplet, which is essential for the supervision of the training. A triplet T is composed of three types of sub-maps: the anchor P_a, the positive P_{pos}, and the negative P_{neg}. The anchor P_a is a sub-map that is simply a LiDAR frame from a driving sequence. The positive sub-maps are the neighboring frames that are not necessarily consecutive but are relatively close to the anchor. Usually, fixed distance and/or time thresholds are defined for determining the positive sub-maps. Conversely, the negative sub-maps are considered dissimilar to the anchor and are also defined by a threshold. The training objective of PointNetVLAD is to minimize the distance between global descriptors of P_{pos} and P_a while maximizing the distance between P_{neg} and P_a. The distance metric used by the authors is Euclidean distance. This training objective can be formulated as the following lazy triplet loss:

$$L_{lazyTrip}(T) = \max_j [\alpha + \delta_{pos} - \delta_{neg_j}]_+$$

$$\delta_{pos} = d(f(P_a), f(P_{pos_i}))$$
$$\delta_{neg_j} = d(f(P_a), f(P_{neg_j}))$$

α is a constant margin for the hinge loss $[...]_+$. In this loss function, the negative sub-map with the smallest distance to the anchor sub-map will be selected to calculate the loss value. In the context of a LiDAR sequence, this means the selected negative sub-map is structurally most similar to the anchor. Therefore the network learns from the hardest negative in each iteration. This approach guarantees that the learned features are more discriminative.

While making the hard negatives more distinguishable from the anchor sub-map, the global descriptor of hard-negatives $f(P^-_{neg_j})$ could also become too close to other sub-maps, which are not really similar to them[UL]. To avoid this issue, the lazy quadruplet loss is employed, which is formulated as follows:

$$L_{lazyQuad}(T, P_{neg^*}) = \max_j([\alpha + \delta_{pos} - \delta_{neg_j}]_+) + \max_k([\beta + \delta_{pos} - \delta_{neg^*_k}]_+)$$

$$\delta_{neg^*_k} = d(f(P_{neg^*}), f(P_{neg_k}))$$

The second maximization term aims to find the most similar negative P_{neg_k} to a randomly sampled P_{neg^*} that is structurally dissimilar to P_{neg_k}.

By using the "lazy" version of triplet and quadruplet loss, training can converge faster while the resulting global descriptors also tend to be more discriminative compared to those generated using the original versions[UL].

The PointNet is already well-known to be permutation invariant[CSKG]. But still, it is important to ensure that the entire PointNetVLAD pipeline retains this property. The authors of the PointNetVLAD provided detailed proof, demonstrating that NetVLAD is a symmetric function and fulfilling this requirement[UL].

3.4 Evaluation Metrics

3.4.1 Precision and Recall

The whole pipeline can be seen as a classifier with the anchor sub-map as its input, classifying all other sub-maps into either positives or negatives. Therefore, precision and recall can be very naturally utilized as evaluation metrics for the model in this work. The recall and precision are defined as follows. Note that the term "true neighbors of the anchor submap" refer to the ground truth positives, which are defined by a distance threshold.

$$\text{Recall} = \frac{\text{number of true positive}}{\text{number of true neighbors of the anchor sub-map}}$$

$$\text{Precision} = \frac{\text{number of true positive}}{\text{number of the retrieved positives of the anchor sub-map}}$$

The number of retrieved positives influences recall and precision. Therefore it is also meaningful for us to evaluate and review the results with the precision-recall curve, which shows us how well the model performs across different retrieved numbers of positives. In this curve, the precision is plotted against the recall, and a good model should have a curve that hugs the top right corner of the plot, indicating high precision and high recall at low probability thresholds. The area under the precision-recall curve (AUPRC) is also a commonly used metric to evaluate the overall performance of the model, with a larger AUPRC indicating better performance. Moreover, the F1-score is also applied as the harmonic mean of the precision and recall, ranging from 0 to 1. The higher the score the better the performance of the classification model, indicating both high recall and precision.

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

If the completion of a scene does not improve place recognition, the experiment will be repeated with the ground truth completed scene assuming an IoU of 100%. This will help to verify the potential effectiveness of the completed scene using the approach proposed in this study.

3.4.2 Wasserstein Distance

The Wasserstein distance is usually used to evaluate the difference between two probability distributions. In this work, we use the first Wasserstein distance, which is calculated in the following way:

$$l_1(u, v) = \inf_{\pi \in \Gamma(u,v)} \int_{R \times R} |x - y| d\pi(x, y)$$

, where the u and v are the probability distributions to be compared.

In the context of our work, the Wasserstein distance is used to evaluate how differently the semantic labels are distributed between two complete scenes. Those complete scenes with the smallest Wasserstein distance to the anchor will be classified as positive.

According to Table 2.1, the results of the predicted semantic scene completion present lower accuracy. Therefore the PointNetVLAD descriptors remain the primary criterion for differentiating between positives and negatives. The Wasserstein distance serves as an additional metric, according to which the most distant retrieved positives should be sorted out. A fixed number, denoted as n, will determine the number of predicted positives to be removed. To this end, the global descriptors' KDTree should predict an additional n positive results.

Chapter 4

Experiment

4.1 Dataset

PointNetVLAD used the Oxford robotcar dataset, which contains 3D submaps of the same places at different times[MPLN17]. However, this dataset is unsuitable for our experiment because it lacks point-wise semantic labeling and densely labeled voxel grids, which are essential when using JS3C-Net. Instead, we use the SemanticKITTI dataset[BGM+], which also offers 3D sub-maps similar to Oxford as well as specific labeling for developing semantic scene completion methods. Besides, the SemanticKITTI dataset is also the currently largest dataset providing sequential real-world LiDAR scans. It contains 22 sequences, among which the sequence 00-10 are used for training[BGM+]. In total, there are more than 23000 LiDAR frames in the training set. Although there are no LiDAR scans of an alternative date, in some sequences the LiDAR scanner-equipped car did visit the same place multiple times. Therefore, it is also possible to generate queries and database for the SemanticKITTI sequences and evaluate whether they are discriminative enough to be used for place recognition.

4.1.1 Properties of SemanticKITTI

The SemanticKITTI contains real-world LiDAR sequences that vary in the environment from one to another. Some of the LiDAR sequences come mainly from the residential area, e.g. the sequences 00, 05, and 07, while the other ones are more taken from the countryside, such as 02 and 03. Sequence 01 uniquely consists of LiDAR scans of highways, where there are almost no buildings in the whole environment. This variation of scene demonstrates a challenge to the scene completion network but also ensures that it can learn many different shape priors[RDCVB22].

The dense semantic annotated point cloud is another important feature provided by SemanticKITTI[BGM+]. It has point-wise dense annotation of in total 28 classes but not all of them are necessary to be considered for the JS3C-Net[YGL+21]. The JS3C-Net classifies the points of each LiDAR frame in 19 categories. Their distributions are illustrated in Figure 4.1.

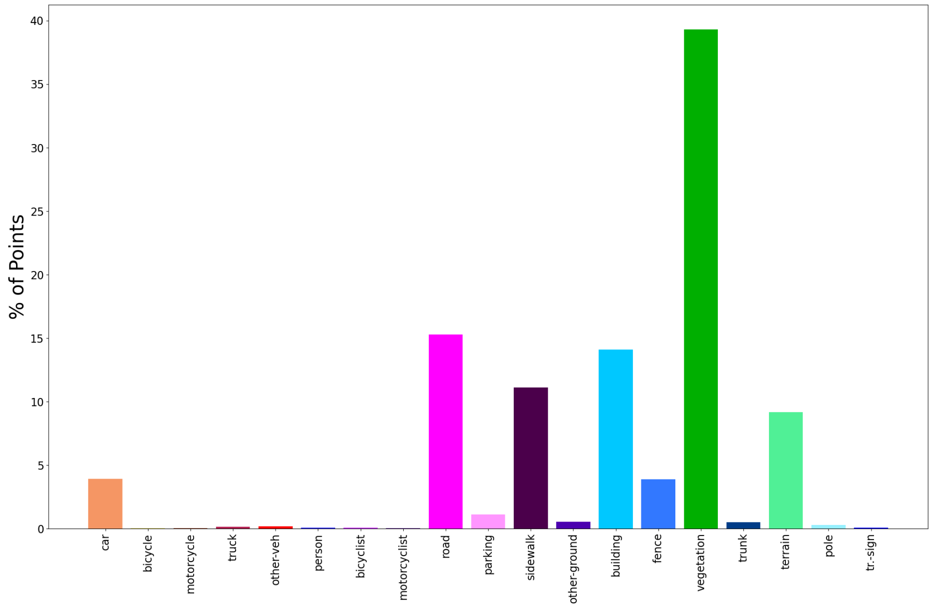

Figure 4.1: Distributions of the 19 categories in SemanticKITTI[BGM+] for semantic scene completion. The imbalance is clear to see.

As we can see in Figure 4.1, the frequencies of different categories in this dataset are extremely imbalanced. Vegetation is obviously the dominant category, while some of the dynamic objects such as cars, trucks and bicycles appear rarely. This is a typical problem of real-world scenes. For achieving semantic scene completion, the imbalance could lead to strong biases, especially for the completed scene, where the proportion of each category might change significantly after the occlusion problem is addressed. Whether this leads to a problem in submap retrieval will be shown in the result section of this chapter. It will be shown if the distribution of semantic labels in the completed scene could help to sort out the false positive submaps, which are retrieved based on the PointNetVLAD descriptors[UL].

4.1.2 Dataset Preparation

Generation of Triplets: For the experiment in this study, only the first 11 sequences (00 to 10) of SemanticKITTI[BGM+] were used. Before the sub-map retrieval begins, the triplets (anchor, positives, negatives) should be generated for the LiDAR sequences. The distance threshold value is set to 3m. This means that the neighboring submaps near the anchor within 3 meters are considered to be the positives. Ideally, all the positives should

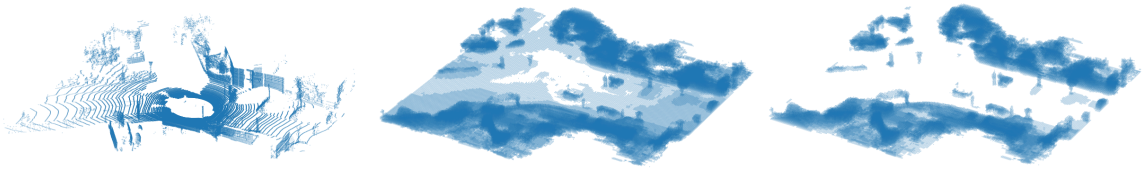

Figure 4.2: Submaps from left to right: input to JS3C-Net[YGL+21], outcome of JS3C-Net after de-voxelization (labels not shown here), input to PointNetVLAD with ground removal before down-sampling. The scanning car is at the center of the submaps. The submap here is the first LiDAR frame of sequence 00

be successfully retrieved during the KDTree search based on Euclidean distance, when the retrieved number of neighbors is set the same as the number of true neighbors. The scripts for generating the triplets are provided by the authors of SG-PR[KYZ+].

Defining the Region of Interest: Secondly, the region of interest for completion of a sub-map should also be adjusted, in order to be consistent with the settings of PointNetVLAD[UL]. By default of JS3C-Net[YGL+21], the completed region is a cube of size $51.2 \times 51.2 \times 6.4m^3$ *in front of* the scanning car, where the x-axis is aligned with the driving direction. However, PointNetVLAD chooses a region of interest where the car is at the center of the point cloud. Therefore, the completed region should be correspondingly shifted. This modification is can be advantageous for localization because the accuracy and density of the LiDAR scan decrease with a longer distance to the scanner (Figure 1.1). The voxel size is set to 20 cm, resulting in the resolution of the voxel grid to be $256 \times 256 \times 32$. Since the PointNetVLAD accepts only point cloud input, the voxel grid outcome after JS3C-Net requires de-voxelization. The center points of each non-empty voxel are extracted to construct the new point cloud. Because the origin of the voxel centers is no longer aligned with the scanning car's coordinate, they are consequently adjusted by an offset obtained with the ground truth voxel centers of the input voxel grid.

Point Cloud Downsampling and Ground Removal: Both JS3C-Net and PointNetVLAD utilized TensorFlow to build their neural networks[YGL+21, UL]. Firstly, the raw and completed submaps are down-sampled to a fixed number of points. For the completed submaps, the down-sampled point cloud has a higher resolution in some of our experiments. On the contrary, the raw point clouds do not have sufficient points to be down-sampled to 8192 or higher resolution inside the region of interest. In the later part of this chapter, we will examine whether using more points significantly enhances the accuracy of submap retrieval. Simultaneously, the ground points must be removed for both completed and raw point clouds because they are redundant and non-informative for submap retrieval. The threshold for cropping of the ground points is set to -1.2m. In Figure 4.3, we can see an example of a pre-processed submap.

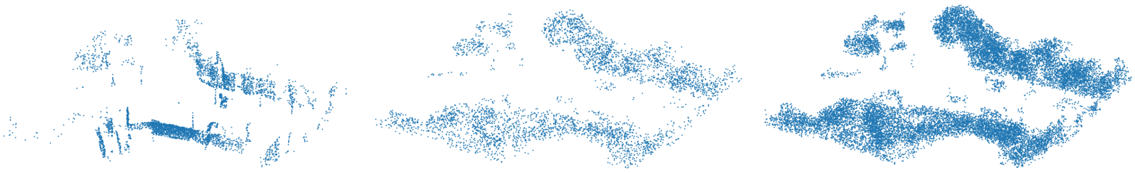

Figure 4.3: Down-sampled submaps from left to right: raw scan to 4096 points, predicted completion to 4096 points, predicted completion to 16384 points. All three submaps here represent the first LiDAR frame of sequence 00

4.2 Implementation Details

4.2.1 Utilization of Pre-trained Models

For both JS3C-Net[YGL+21] and PointNetVLAD[UL], we use publicly available pre-trained models without any fine-tuning. As previously mentioned, the SemanticKITTI[BGM+] suffers from a drawback in comparison to other benchmarks in that it only offers LiDAR scans of one date for all the sequences. To suppress the influence of adjacent LiDAR frames, the sequences with loop closures are utilized as test sequences. It is also meaningful in the sense that loop closure detection is relevant to visual SLAM[KYZ+, LKZ+]. Those sequences in SemanticKITTI are 00, 02, 05, 06, 07, and 08. Thanks to the contribution of Xin Kong et al.[KYZ+], the pre-trained model provided by their repository was trained using 1-Fold strategy with those sequences, where the loop closures appear.

4.2.2 Generation of Global Descriptors

Initially, the PointNetVLAD was set to receive a point cloud input of 4096 points[UL]. In our experiments, we also use this setting for the raw point clouds. As for the completed scenes, there are two different resolutions applied: 4096 and 16384. The generation of global descriptors was run on a TITAN Xp GPU. For sequence 02 with 4661 submaps, the generation of the database takes about 4 min. The other sequences need less inference time as they are all shorter than sequence 02. The descriptors for the queries are also generated offline to make the retrieval more efficient for this study.

4.3 Results and Comparison

4.3.1 Submap Retrieval with Completed Scene

During the dataset preparation phase, the true neighbors of each submap are already defined. However, due to loop closures and revisiting of some places in the test sequences, the number of true neighbors is not fixed for all submaps. For the first part of the comparison, we evaluate the retrieval of true neighbors in each submap by setting the number of positives to be extracted the same as the number of true neighbors. The quantitative results can be found in Table 4.1:

Table 4.1: Performance comparison of retrieval performance on SemanticKITTI dataset using original(PointNetVLAD) and completed(JS3C-PointNetVLAD) scene. The recall values are shown for both configurations. Both raw and completed point clouds are down-sampled to 4096 points.

Sequence	**PointNetVLAD**	**JS3C-PointNetVLAD**	**Improvement**
00	**0.8028**	0.7740	-3.59%
02	**0.8488**	0.8157	-3.89%
05	**0.6815**	0.5496	-19.36%
06	**0.8451**	0.8011	-5.20%
07	**0.8325**	0.8023	-3.63%
08	**0.8058**	0.7701	-4.43%
Average	**0.8028**	0.7521	-6.32%

From the table, we can see that using JS3C-PointNetVLAD results in a decrease in recall compared to PointNetVLAD[UL] in all sequences. It seems that using JS3C-Net[YGL+21] to pre-process the sub-maps does not improve the retrieval performance of PointNetVLAD on the SemanticKITTI dataset[BGM+]. Especially for sequence 05, the decrease in the recall is already quite significant.

Secondly, we also would like to determine the improvement achieved by increasing the number of points applied to PointNetVLAD. The corresponding results are presented in **??**.

Table 4.2: Performance comparison of retrieval performance with increased the number of points being used, on SemanticKITTI dataset. The completed scenes are down-sampled to 16384 points while the raw point clouds are down-sampled to 4096 points. The recalls are shown for both configurations.

Sequence	**PointNetVLAD**	**JS3C-PointNetVLAD**	**Improvement**
00	**0.8028**	0.7808	-2.74%
02	**0.8488**	0.8261	-2.68%
05	**0.6815**	0.5649	-17.09%
06	**0.8451**	0.8129	-3.82%
07	**0.8325**	0.8075	-3.00%
08	**0.8058**	0.7784	-3.40%
Average	**0.8028**	0.7618	-5.11%

As we can see in Table 4.2, the increased number of points did bring much improvement to the recall. In fact, the improvement is quite poor and the performance of JS3C-PointNetVLAD still does not surpass the PointNetVLAD[UL], where only the point clouds without completion are used. This is probably because the predicted complete scene suffers from more noise and inaccuracy due to discretizing as well as the learning-based approach. The inaccuracy also leads to significant shape changes in the scene, which can be seen in Figure 1.1.

For practice, the predicted top closest neighbors to the anchor point cloud are more interesting. The next evaluation was conducted to obtain the precision @5, and the results are presented in Table 4.3.

Table 4.3 shows a decrease in average @5 precision of -4.39%. Similar to the previous evaluation, the decrease in sequence 05 appears to be the most significant.

Table 4.3: The performance comparison on SemanticKITTI, where the network should retrieve the top 5 most likely positives of the query sub-map.

Sequence (4096 points)	**PointNetVLAD (@5 precision)**	**JS3C-PointNetVLAD (@5 precision)**	**Improvement**
00	**0.9597**	0.9369	-2.37%
02	**0.9076**	0.8762	-3.46%
05	**0.8680**	0.7494	-13.66%
06	**0.9366**	0.9117	-2.66%
07	**0.9748**	0.9526	-2.27%
08	**0.9593**	0.9332	-2.72%
Average	**0.9343**	0.8933	-4,39%

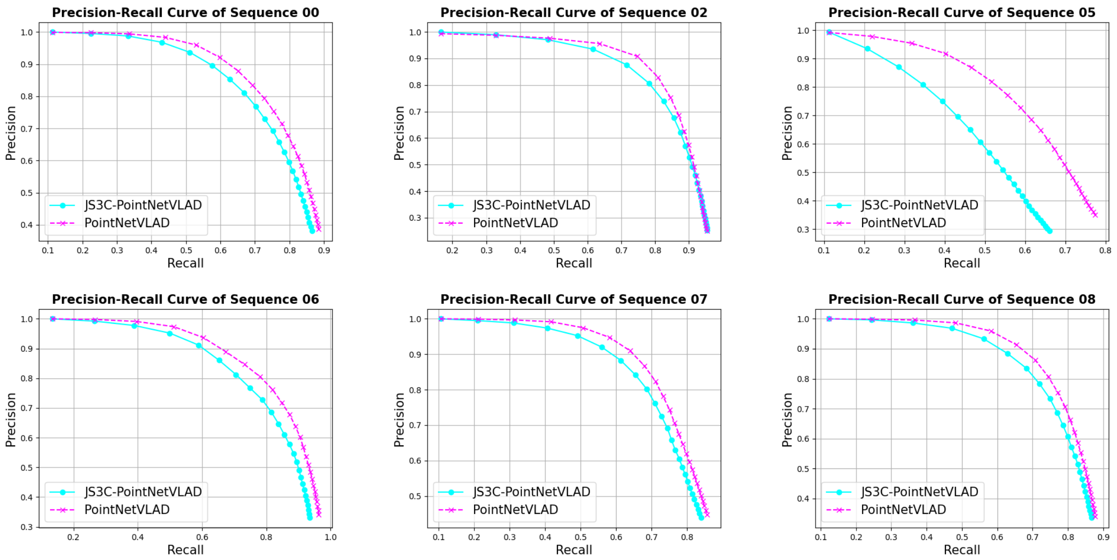

Figure 4.4: Qualitative comparison of retrieving completed scene(JS3C-PointNetVLAD) and the original raw scan(PointNetVLAD)

Moreover, the precision-recall curves also help visually evaluate the quality of the generated global descriptors. The precision-recall curves of the test sequences are depicted in Figure 4.4, providing a more comprehensive evaluation of the effectiveness of scene completion. For all the test sequences, the JS3C-PointNetVLAD leads to a smaller AUC, meaning that the descriptors are less distinguishable between positive and negative by Euclidean distance. Consequently, we can obtain the maximal F1 scores for the test sequences. The max F1 score shows us the best accuracy of the model considering both recall and precision. The results of using a completed scene with 4096 input point clouds can be checked in Table 4.4.

Table 4.4: The F1 scores of the test sequences. The point clouds are down-sampled to 4096 points.

Sequence	**PointNetVLAD**	**JS3C-PointNetVLAD**	**Improvement**
00	**0.7590**	0.7341	-3.28%
02	**0.8197**	0.7938	-3.16%
05	**0.6507**	0.5414	-16.80%
06	**0.7929**	0.7582	-4.38%
07	**0.7636**	0.7401	-3.07%
08	**0.7772**	0.7510	-3.37%
Average	**0,7605**	0,7198	-5.36%

From Table 4.4, it is clear to see that the PointNetVLAD alone generates better global descriptors, which can be distinguished from each other more accurately.

4.3.2 Submap Retrieval using Ground Truth Completed Scene

In the previous section, the performance of JS3C-PointNetVLAD was found to be unsatisfactory. This could be a result of the fact that publicly available well-trained was trained for a relatively small number of epochs. To assess whether further training is necessary, this section tests the potential performance improvement that could be achieved by repeating the same experiments using the ground truth completed scenes. Same as in the first experiment, all sub-maps are firstly down-sampled to 4096 points. The model is referred to as "GT-PointNetVLAD", if it uses ground truth complete scenes.

According to Table 4.5, GT-PointNetVLAD achieves a higher recall in two sequences. The decrease in recall in most of the sequences is significant while the improvements in sequences 02 and 08 appear to be poor. On average, there is still a performance drop of -2.62%. By increasing the number of points applied to PointNetVLAD, the performance of GT-PointNetVLAD becomes better. However, it still outperforms PointNetVLAD in only 3 sequences. In terms of the improvement achieved by GT-PointNetVLAD over PointNetVLAD, we can see that using the ground truth completed scene did not bring significant advantages for generating more discriminative global descriptors. A similar conclusion can also be drawn by observing the qualitative evaluation with PR-curves(Figure 4.5).

Table 4.5: Comparing submap retrieval performance using the raw scans(PointNetVLAD) and ground truth completed scenes(GT-PointNetVLAD) on test sequences.

Sequence (4096 points)	**PointNetVLAD (recall)**	**GT-PointNetVLAD (recall)**	**Improvement**
00	**0.8046**	0.7981	-0.81%
02	0.8466	**0.8470**	0.047%
05	**0.6790**	0.6043	-11.00%
06	**0.8437**	0.8057	-4.50%
07	**0.8328**	0.8167	-1.93%
08	0.8043	**0.8130**	1.08%
Average	**0.8018**	0.7808	-2.62%
Sequence (16384 points)	**PointNetVLAD (recall)**	**GT-PointNetVLAD (recall)**	**Improvement**
00	0.8028	**0.8069**	0.51%
02	0.8488	**0.8627**	1.64%
05	**0.6815**	0.6461	-5.20%
06	**0.8451**	0.8174	-3.28%
07	**0.8325**	0.8313	-0.14%
08	0.8058	**0.8281**	2.77%
Average	**0.8027**	0.7988	-0.49%

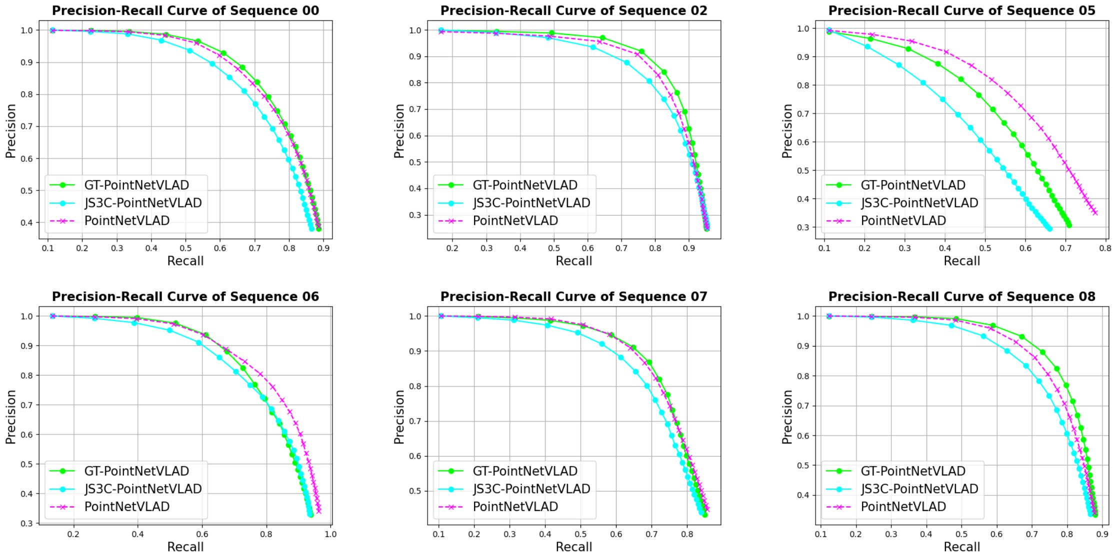

Figure 4.5: Qualitative comparison of retrieving ground truth completed scene(GT-PointNetVLAD), predicted completed scene(JS3C-PointNetVLAD) and the raw scan(PointNetVLAD). All of them applied 4096 points to the PointNetVLAD.

Based on the observations to Figure 4.5, GT-PointNetVLAD outperforms PointNetVLAD[UL] in terms of AUC in three sequences by small marginal differences. Even though GT-PointNetVLAD already results in larger AUC than JS3C-PointNetVLAD, it has been demonstrated that training JS3C-Net for additional epochs will probably not help for an obvious performance boost.

4.3.3 Submap Retrieval with Semantic Labels

In addition to the global descriptors, we also evaluate if the distribution of point-wise labels can also aid in submap retrieval. As mentioned in the previous chapter, the retrieval is firstly conducted only by the PointNetVLAD descriptors[UL]. After the predicted positive submaps are obtained, the least similar 5 positives will be removed regarding the Wasserstein distance between histograms. The recalls of trying to retrieve all true neighbors are shown in Table 4.6:

Table 4.6: Comparison of recalls with/without leveraging semantic segmentation. The point clouds are down-sampled to 4096 points. GT-PointNetVLAD refers to the utilization of ground truth completed scene for PointNetVLAD.

Sequence	**JS3C-PointNetVLAD (without semantics)**	**JS3C-PointNetVLAD (with semantics)**	**Improvement**
00	**0.7712**	0.7069	-8.34%
02	**0.8133**	0.6972	-14.28%
05	0.5514	**0.5663**	2.70%
06	**0.8058**	0.7086	-12.06%
07	**0.8004**	0.7315	-8.61%
08	**0.7704**	0.6764	-12.20%
Average	**0.7521**	0.6812	-9.43%
Sequence	**GT-PointNetVLAD (without semantics)**	**GT-PointNetVLAD (with semantics)**	**Improvement**
00	**0.7982**	0.6900	-13.56%
02	**0.8470**	0.6778	-19.98%
05	**0.6043**	0.6004	-0.65%
06	**0.8057**	0.7024	-12.82%
07	**0.8167**	0.6910	-15.39%
08	**0.8130**	0.6744	-17.05%
Average	**0.7808**	0.6727	-17.26%

In Table 4.6 we can see that the recall dropped drastically in recall for almost all the test sequences except for sequence 05. Through this comparison, it is obvious to see that the label information did not aid in sorting out false positives of the retrieval. Due to the poor performance using predicted labels and the low mIoU of semantic scene completion (Table 2.1), we repeated the experiment using ground truth completed point clouds and labels. However, despite using ground truth labeling, the drop in recall was still substantial.

4.3.4 Discussion

As noted in the previous chapter, the retrieval performance did not exhibit significant improvement from the use of the completed scene. At first glance, the output scene from completion appears to be satisfactory in terms of its completion results, as it enriches certain local details such as the contour of vehicles and trees (see Figure 4.6). However, the issue lies in the fact that PointNetVLAD is aimed at low-resolution point clouds, not the high-resolution point clouds generated from semantic scene completions[UL]. E.g. in our experiment, the number of points is down-sampled to 4096 or 16384. Even when a submap consists of 16384 points, it is a mere fraction of the 100000 points present in a typical complete scene. Specifically for the predicted complete scene, the visual quality of the down-sampled complete scene appears to be much worse than the raw scan shown in Figure 4.3. This is because of the limitation of the learning-based approach that the prediction can not match the ground truth 100%. Even for humans, it becomes hard to identify different semantic parts of a scene. Furthermore, as a result of random sampling, there was a lot of additional noise mixed with relevant geometric structures. So the PointNetVLAD[UL] ended up focusing more on the noise than the actual important information, which made the submap retrieval less accurate overall.

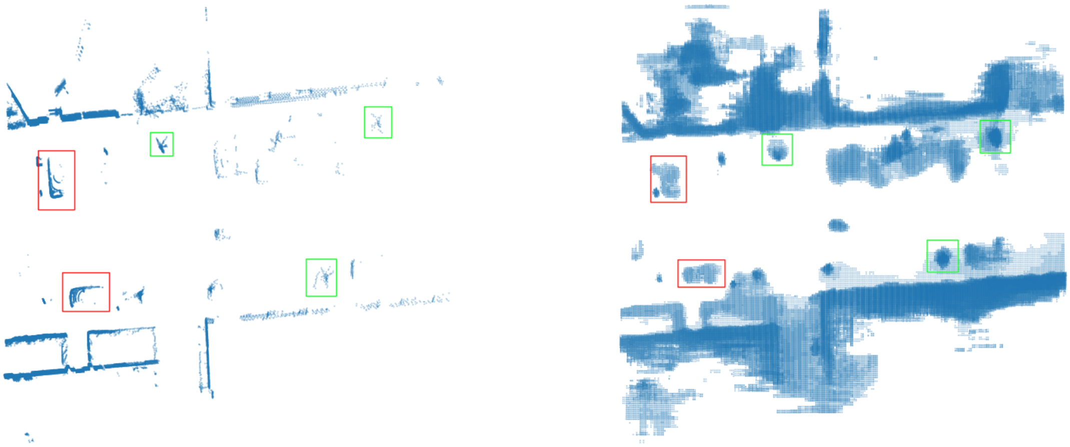

Figure 4.6: Green boxes: trees. Red boxes: vehicles

It should be noted that the usefulness of a complete scene cannot be entirely dismissed. In fact, the ground truth complete scenes obtained through point cloud registration aided in achieving higher recall and precision for certain sequences (Figure 4.5). A more advanced and sophisticated approach may be required to fully bring out the potential of scene completion.

Chapter 5

Conclusion and Future Work

As the final conclusion, the completed scene as well as semantic labels can not directly help improve the quality of the global descriptors generated by PointNetVLAD[UL]. It has been observed that despite utilizing the ground truth information, the enhancement achieved remains minimal. This suggests that a more sophisticated approach might be necessary to incorporate semantic scene completion in addition to the geometry of the sparse point cloud. An example of a recent approach that incorporates semantic labels in a more advanced manner is SG-PR[KYZ$^+$]. This approach has already exhibited significantly enhanced accuracy in place recognition in comparison to PointNetVLAD[KYZ$^+$]. Besides, the limitation of PointNetVLAD is its focus on constructing global features, without much consideration given to lower and mid-level features extracted by the PointNet module, which could contribute to constructing a more discriminative global descriptor.

As for the scene completion, it will make more sense to complete part of the large-scale scene, which is more relevant to the place recognition problem. However, the relevant part of a point cloud is not easy to be explicitly defined because it can also depend on the distribution of features in the whole sequence. Recently the attention mechanism has gained much popularity in the artificial intelligence community[XXL$^+$], which can probably help learn those relevant semantics to be completed. An end-to-end network inclusive scene completion with attention module and place recognition might contribute to more accurate and reliable place recognition.

Regarding the computational complexity and the advances in object point cloud completion, it might also be meaningful to develop semantic scene completion methods without any voxelization. Taking the example of JS3C-Net[YGL$^+$21] the outcome consists of more than 2 million voxels, where the majority of them are actually empty. It means that they do not represent any geometric structures and are therefore non-informative[YGL$^+$21]. But in the case of point cloud completion, the number of output points can be set to a much smaller value of e.g. 16384, where all of the predicted points are valid.

List of Figures

1.1 Comparison of a raw scan and its corresponding completed scene from SemanticKITTI. From top to bottom: raw scan, predicted completed scene, ground truth completed scene. 2

2.1 Left: predicted semantic scene completion obtained by JS3C-Net. Right: ground truth completed scene obtained by point cloud registration. Both submaps represent the first LiDAR frame of sequence 00 from SemanticKITTI. The submaps illustrate the region of interest before the driving direction. 8

3.1 Overall pipeline for submap retrieval in this work. 9
3.2 Left: anchor submap, which is the query sub-map. Middle: positive submap of the anchor. Right: negative submap of the anchor. Note that the hereby presented negative was erroneously retrieved as positive by PointNetVLAD[UL]. The submaps shown here are not completed. 10
3.3 Network architechture of JS3C-Net.[YGL+21] 10
3.4 Insight of the semantic scene completion module.[YGL+21] 11
3.5 Network architechture of PointNetVLAD.[UL] 12

4.1 Distributions of the 19 categories in SemanticKITTI[BGM+] for semantic scene completion. The imbalance is clear to see. 17
4.2 Submaps from left to right: input to JS3C-Net[YGL+21], outcome of JS3C-Net after de-voxelization (labels not shown here), input to PointNetVLAD with ground removal before down-sampling. The scanning car is at the center of the submaps. The submap here is the first LiDAR frame of sequence 00 18
4.3 Down-sampled submaps from left to right: raw scan to 4096 points, predicted completion to 4096 points, predicted completion to 16384 points. All three submaps here represent the first LiDAR frame of sequence 00 19
4.4 Qualitative comparison of retrieving completed scene(JS3C-PointNetVLAD) and the original raw scan(PointNetVLAD) 21

4.5 Qualitative comparison of retrieving ground truth completed scene(GT-PointNetVLAD), predicted completed scene(JS3C-PointNetVLAD) and the raw scan(PointNetVLAD). All of them applied 4096 points to the PointNetVLAD. 23
4.6 Green boxes: trees. Red boxes: vehicles 25

List of Tables

2.1 SemanticKITTI[BGM+] Challenge Rankings. The JS3C-Net in bold is the network employed in this study. 7
2.2 Scene Completion Ranking on SemanticKITTI[BGM+]. The JS3C-Net in bold is the network employed in this study. 7

4.1 Performance comparison of retrieval performance on SemanticKITTI dataset using original(PointNetVLAD) and completed(JS3C-PointNetVLAD) scene. The recall values are shown for both configurations. Both raw and completed point clouds are down-sampled to 4096 points. . . 20
4.2 Performance comparison of retrieval performance with increased the number of points being used, on SemanticKITTI dataset. The completed scenes are down-sampled to 16384 points while the raw point clouds are down-sampled to 4096 points. The recalls are shown for both configurations. 20
4.3 The performance comparison on SemanticKITTI, where the network should retrieve the top 5 most likely positives of the query sub-map. 21
4.4 The F1 scores of the test sequences. The point clouds are down-sampled to 4096 points. 22
4.5 Comparing submap retrieval performance using the raw scans(PointNetVLAD) and ground truth completed scenes(GT-PointNetVLAD) on test sequences. 23
4.6 Comparison of recalls with/without leveraging semantic segmentation. The point clouds are down-sampled to 4096 points. GT-PointNetVLAD refers to the utilization of ground truth completed scene for PointNetVLAD. . . . 24

Bibliography

[AGT$^+$] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5297–5307. IEEE.

[BGM$^+$] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9296–9306. IEEE.

[CAR$^+$21] Ran Cheng, Christopher Agia, Yuan Ren, Xinhai Li, and Liu Bingbing. S3cnet: A sparse semantic scene completion network for lidar point clouds. In *Conference on Robot Learning*, pages 2148–2161. PMLR, 2021.

[CLZ$^+$22] Yingjie Cai, Kwan-Yee Lin, Chao Zhang, Qiang Wang, Xiaogang Wang, and Hongsheng Li. Learning a structured latent space for unsupervised point cloud completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5543–5553, 2022.

[CSKG] R. Qi Charles, Hao Su, Mo Kaichun, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85. IEEE.

[HKK$^+$] David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision â ECCV 2010*, volume 6313, pages 356–369. Springer Berlin Heidelberg. Series Title: Lecture Notes in Computer Science.

[HYX$^+$] Zitian Huang, Yikuan Yu, Jiawen Xu, Feng Ni, and Xinyi Le. PF-net: Point fractal network for 3d point cloud completion. In *2020 IEEE/CVF Confer-*

ence on Computer Vision and Pattern Recognition (CVPR), pages 7659–7667. IEEE.

[KWT] Jacek Komorowski, Monika Wysoczanska, and Tomasz Trzcinski. MinkLoc++: Lidar and monocular image fusion for place recognition. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

[KWT21] Jacek Komorowski, Monika Wysoczanska, and Tomasz Trzcinski. Egonn: Egocentric neural network for point cloud based 6dof relocalization at the city scale. *IEEE Robotics and Automation Letters*, 7(2):722–729, 2021.

[KYZ+] Xin Kong, Xuemeng Yang, Guangyao Zhai, Xiangrui Zhao, Xianfang Zeng, Mengmeng Wang, Yong Liu, Wanlong Li, and Feng Wen. Semantic graph based place recognition for 3d point clouds. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8216–8223. IEEE.

[LKZ+] Lin Li, Xin Kong, Xiangrui Zhao, Tianxin Huang, Wanlong Li, Feng Wen, Hongbo Zhang, and Yong Liu. SSC: Semantic scan context for large-scale place recognition. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2092–2099. IEEE.

[LZS+] Zhe Liu, Shunbo Zhou, Chuanzhe Suo, Peng Yin, Wen Chen, Hesheng Wang, Haoang Li, and Yunhui Liu. LPD-net: 3d point cloud learning for large-scale place recognition and environment analysis. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2831–2840. IEEE.

[MPLN17] Will Maddern, Geoff Pascoe, Chris Linegar, and Paul Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017.

[PCC+] Liang Pan, Xinyi Chen, Zhongang Cai, Junzhe Zhang, Haiyu Zhao, Shuai Yi, and Ziwei Liu. Variational relational point completion network. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8520–8529. IEEE.

[RDCVB22] Luis Roldao, Raoul De Charette, and Anne Verroust-Blondet. 3d semantic scene completion: A survey. *International Journal of Computer Vision*, 130(8):1978–2005, 2022.

[SRKB10] Bastian Steder, Radu Bogdan Rusu, Kurt Konolige, and Wolfram Burgard. Narf: 3d range image features for object recognition. In *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, volume 44, page 2. Citeseer, 2010.

[SWZ+] Jiaming Sun, Zihao Wang, Siyu Zhang, Xingyi He, Hongcheng Zhao, Guofeng Zhang, and Xiaowei Zhou. OnePose: One-shot object pose estimation without CAD models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6815–6824. IEEE.

[UL] Mikaela Angelina Uy and Gim Hee Lee. PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4470–4479. IEEE.

[WAL] Xiaogang Wang, Marcelo H. Ang, and Gim Hee Lee. Cascaded refinement network for point cloud completion. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 787–796. IEEE.

[XXL$^+$] Yan Xia, Yusheng Xu, Shuang Li, Rui Wang, Juan Du, Daniel Cremers, and Uwe Stilla. SOE-net: A self-attention and orientation encoding network for point cloud based place recognition. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11343–11352. IEEE.

[YGL$^+$21] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3101–3109, 2021.

[YKH$^+$18] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 international conference on 3D vision (3DV)*, pages 728–737. IEEE, 2018.