TOWARDS ACCURATE INSTANCE SEGMENTATION IN LARGE-SCALE LIDAR POINT CLOUDS

Binbin Xiang[1], Torben Peters[1], Theodora Kontogianni[1], Frawa Vetterli[1], Stefano Puliti[2], Rasmus Astrup[2], Konrad Schindler[1]

[1]ETH Zürich, Switzerland - (bxiang, tpeters, tkontogianni, vfrawa, schindler)@ethz.ch
[2]Norwegian Institute of Bioeconomy Research (NIBIO) - (Stefano.Puliti, rasmus.astrup)@nibio.no

KEY WORDS: 3D point cloud, panoptic segmentation, instance segmentation, semantic segmentation

ABSTRACT:

Panoptic segmentation is the combination of semantic and instance segmentation: assign the points in a 3D point cloud to semantic categories *and* partition them into distinct object instances. It has many obvious applications for outdoor scene understanding, from city mapping to forest management. Existing methods struggle to segment nearby instances of the same semantic category, like adjacent pieces of street furniture or neighbouring trees, which limits their usability for inventory- or management-type applications that rely on object instances. This study explores the steps of the panoptic segmentation pipeline concerned with clustering points into object instances, with the goal to alleviate that bottleneck. We find that a carefully designed clustering strategy, which leverages multiple types of learned point embeddings, significantly improves instance segmentation. Experiments on the *NPM3D* urban mobile mapping dataset and the *FOR-instance* forest dataset demonstrate the effectiveness and versatility of the proposed strategy.

1. INTRODUCTION

Laser scanning has emerged as a main sensing technology to digitise 3D scenes, thanks to its ability to deliver dense 3D point observations with high reliability. The unstructured point clouds it produces are, however, not directly usable as a product (except for visualisation) and must be processed further to extract meaningful entities for mapping and analysis. Panoptic segmentation (Kirillov et al., 2019, Zhou et al., 2021) addresses the case where the desired entities are semantically meaningful objects, like individual trees or traffic signs. [1]

Panoptic segmentation is a generic and versatile processing step that may be useful across many different fields. In the context of street scenes, it facilitates scene understanding and mapping at the level of objects, like buildings, traffic signs, pedestrians, etc. (Fong et al., 2022, Chen et al., 2022), which in turn supports applications from urban planning to autonomous vehicles. In forest regions, panoptic segmentation can localise and delimit individual trees, which in turn supports applications like resource management, environmental protection and ecological restoration (Calders et al., 2020).

Large-scale outdoor point clouds pose particular challenges for panoptic segmentation. Besides common problems of point cloud processing, such as occlusions, moving objects and a large range of object scales and point densities (Chen and Yang, 2016), an important issue is the lack of natural "processing units": unlike indoor scans that can be processed on a per-room basis (Armeni et al., 2016, Dai et al., 2017) or panoramic scans from robotic systems that can be processed on a per-scan basis (Behley et al., 2019), there is no natural way of dividing an outdoor scene into independent subsets. A specific difficulty of panoptic segmentation is the requirement to separate objects of the same category, which can be considerably harder than only assigning semantic labels to points, as in the case of trees with overlapping crowns.

Modern panoptic segmentation techniques are often built upon a 3D deep network backbone that extracts per-point features, followed by network branches that segment the points into semantic categories and into object instances, based on those features. The backbone network is not the focus of this paper. We treat it as a plug-in module of our overall network that ingests a point cloud and returns a feature vector of fixed length for every point. Multiple well-proven, trainable feature extractors exist for the task (Thomas et al., 2019, Choy et al., 2019). Semantic segmentation also has reached a certain level of maturity and can be regarded as a commodity. Technically, the associated network branch is a classifier that maps the feature representation to a list of (pseudo-)probabilities per point and is typically trained by minimising the cross-entropy loss. We follow that practice, but do not deeply delve into the details. The focus of the present paper is on the instance segmentation branch, arguably the least explored part of the problem and the current performance bottleneck. There are two different strategies to identify object instances in point clouds.

1.1 Top-down instance detection

The top-down approach first performs object detection to obtain a set of bounding boxes around 3D object candidates. Then the points inside each box are separated into points on the object and points on the background with a binary classifier. The quality of such methods largely depends on the object detection step. The earliest attempts at instance segmentation were top-down methods (Yang et al., 2019), following the success of Mask-RCNN (He et al., 2017) in the image domain, but later they were surpassed by bottom-up methods (see below). There is recent evidence that in certain types of (indoor) scenarios the top-down approach is competitive or even superior (Kolodiazhnyi et al., 2023). Here, we do not further investigate the top-down strategy for two reasons: (1) For outdoor scenes, bounding box detectors tend to work well only for a small number of categories, especially pedestrians and vehicles (Zhang et al., 2020), whereas they often miss small objects like bollards, and objects that have greatly varying shape and aspect ratio, for instance trees. (2) Outdoor mapping point clouds cannot be

[1] As opposed to low-level primitives without semantic meaning, such as salient keypoints or planar surfaces.

https://doi.org/10.5194/isprs-annals-X-1-W1-2023-605-2023

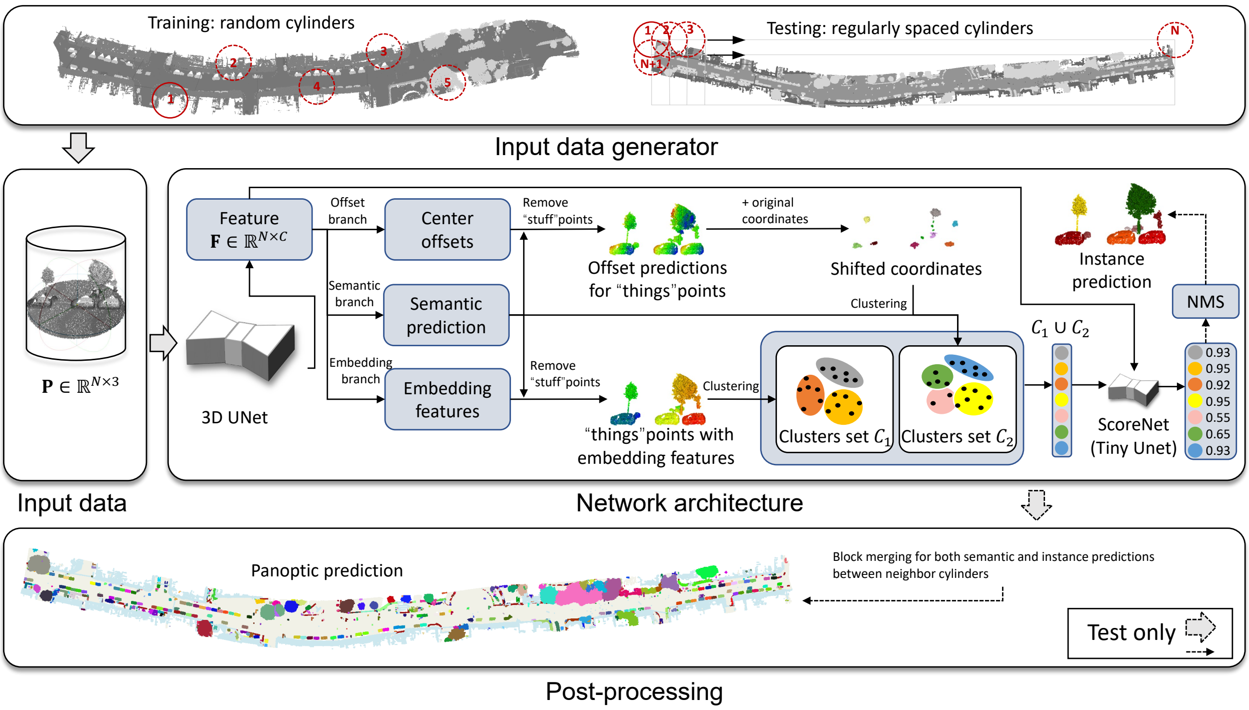

Figure 1. The bottom-up panoptic segmentation pipeline studied in this paper.

split into natural entities like rooms, instead they have to be subdivided into arbitrary, computationally manageable chunks using a sliding window or random sampling. Consequently, many objects – especially large ones – are cut into parts and only partially visible in each chunk, making them hard to find for an object detector based on global shape and layout.

1.2 Bottom-up instance grouping

The bottom-up strategy aims to equip each individual point with an instance-sensitive feature representation, such that instances can be found by clustering the points in the associated feature space. These (learned) instance features are computed with the help of a neural network, based on the point coordinates and/or the backbone features (Han et al., 2020, Lahoud et al., 2019, Engelmann et al., 2020). A natural feature to find instances is the offset from the point to the instance centre, in the spirit of the (generalised) Hough transform. An important finding in this context was that the unsupervised clustering step is, by itself, rather unreliable. To address the issue, PointGroup (Jiang et al., 2020) proposed to run multiple clustering variants and obtain a redundant set of clusters. The quality of these instance candidates is then estimated with a (learned) ScoreNet, such that they can be sorted by their scores and pruned to an optimal set of instances with non-maximum suppression (NMS). The principle to let multiple candidate segmentations compete, and to thereby benefit from the complementary strengths of different clustering methods, proved to work very well and sparked a series of follow-up works that further explored the idea. MaskGroup (Zhong et al., 2022) clusters at different spatial scales, and SoftGroup (Vu et al., 2022) keeps soft semantic labels, so as to enable clustering across different categories and rectify semantic segmentation errors. HAIS (Chen et al., 2021) adds a MaskNet after the clustering step, which examines each individual cluster and aims to detect and remove points that do not belong to the object instance. (Engelmann et al., 2020) is also based on instance proposals, and refines them by modelling their relations with a graph neural network, which is again more suitable for complete, self-contained target areas like indoor rooms. (Liang et al., 2021) construct a cluster hierarchy and traverse the associated tree to generate proposals, which are then again assessed with a ScoreNet. For the present study we also build on the PointGroup principle. We note that, contrary to most other existing work, it has also been shown to work in outdoor settings.

For completeness, we mention that directly clustering points into instances in 3D scene space – arguably the most obvious strategy – does not work well for many mapping tasks. For compact, well-separated objects like vehicles on the road this strategy can work quite well (Zhao et al., 2021), but it tends to fail as soon as objects are located in close proximity, such as tightly parked cars; or even touch, like the crowns of nearby trees. There is a consensus in the literature that in such situations a-priori knowledge about the objects' shapes and layouts is required, e.g., (Lahoud et al., 2019, Engelmann et al., 2020, Jiang et al., 2020). Conveniently, that knowledge is also needed for semantic segmentation, so it can be derived from the same latent features with little computational overhead.

Recently, transformer-type neural networks were applied for instance segmentation (Liu et al., 2022, Schult et al., 2023, Sun et al., 2023), following a trend in the 2D image domain (Zhang et al., 2021, Cheng et al., 2022). The principle is to replace the explicit instance feature extraction and clustering step with instance queries, based on the attention mechanism of the transformer architecture. So far these methods have only been demonstrated on indoor datasets. They appear to be particularly successful in terms of *detecting* instances, whereas the per-point segmentation performance is on par with PointGroup-style methods. In practical terms, the learned proposal generator is rather elegant, but comes at the cost of significantly higher memory demand. When processing densely scanned outdoor point clouds, GPU memory is the limiting factor even for conventional, convolution-based methods. Hence, we exclude transformers from this study, but note that adapting them to outdoor mapping is an interesting future direction.

1.3 Contributions

We have developed an effective deep learning-based workflow for panoptic segmentation of large outdoor mapping point clouds, based on the bottom-up grouping strategy. Design choices for instance segmentation are carefully evaluated and analysed in a series of experiments on two different data sets, one showing streets scenes (NPM3D) and one dense forests (FOR-instance). Our main findings are: (1) The often used grouping based on centroid offsets struggles to separate object instances located close to each other. A learned feature embedding, trained with a contrastive loss to discriminate instances, can often separate such instances. (2) On the other hand, offset vectors more accurately separate objects that have similar local shape, but are located far from each other. We find that the best results are achieved by combining both methods and letting the subsequent ScoreNet select from both proposal sets. (3) Clustering based on embedding features, which does not depend directly on the semantic segmentation result, reduce mistakes caused by incorrect semantic labels and thereby improve the completeness of the affected instances. (4) A simple block merging strategy is sufficient to combine the segmentations of local subsets into a coherent large-scale panoptic segmentation map. (5) State-of-the-art methods for 3D panoptic segmentation work well even for challenging tasks like separating tree crowns in dense forests. We expect those methods to be more widely adopted for practical applications in the near future.

2. METHOD

The overall pipeline of our proposed method, shown in Figure 1, consists of three main components: an input data generator (Section 2.1), a deep neural network (Section 2.2), and a post-processor (Section 2.3).

2.1 Input data generator

As a first step, the entire point cloud is voxel-grid subsampled to sparsify overly dense regions and achieve a homogeneous (maximum) point density. The voxel size for the filter depends on the scene. In our implementation we use $12{\times}12{\times}12\,\mathrm{cm}^3$ (579 points/m^3) for urban scenes and $20{\times}20{\times}20\,\mathrm{cm}^3$ (125 points/m^3) for forest scenes, see Table 1. These values were chosen based on extensive ablation studies performed in (Xiang et al., 2023). Even so, outdoor scans are far too large to process as a whole on existing hardware. As an example, a single scene from the NPM3D dataset (Roynard et al., 2018), covering a stretch of road of length $\approx 600\,\mathrm{m}$, has several million points. It is therefore necessary to process the data in local blocks. When applying the trained network to new data these blocks can be sampled in sliding-window fashion. During training, we simply sample them randomly. There are different ways to define the local neighborhood of points that constitutes a block around a sampled location, popular choices include cubic boxes, spheres or cylinders. We opt for the cylinder, for the following reasons: (1) It avoids cutting objects along the vertical, which on the one hand improves the handling of long vertical objects such as street lamps or trees, and on the other hand simplifies block merging to a 2D problem (Section 2.3). (2) It ensures computational efficiency. When working with large point clouds, the computational bottleneck is not error back-propagation, but rather geometric queries like finding neighbours. Cylindrical neighbourhoods are compliant with efficient algorithms like fast radius search (normally implemented via spatial search structures such as KD-trees, and available in the Torch-Point3D framework (Chaton et al., 2020)).

To sample training cylinders in a way that ensures sufficient coverage of rare categories, we take inspiration from KP-Conv (Thomas et al., 2019): the location of the vertical cylinder axis is found by randomly sampling one of the training data points, with sampling probabilities proportional to the square root of the inverse class frequencies, $P_i \propto \sqrt{1/N_i}$. After sampling a fixed training set of many cylindrical blocks, the points' (x, y)-coordinates in each of them are shifted to have their origin in the cylinder centre. Moreover, various data augmentation techniques are applied: isotropic, additive Gaussian random noise on the point coordinates (jittering), random rotations around the cylinder axis, random anisotropic scaling by factors $s \in [0.9, 1.1]$, and random reflection along the y-axis. At test time the cylindrical blocks are sampled regularly with fixed step size along the (x, y)-grid so as to ensure even coverage of the point cloud, see the illustration of the input data generator in Figure 1.

2.2 Network architecture

As **feature extraction backbone** we use the Minkowski Engine (Choy et al., 2019), which offers a favourable trade-off between performance and computational cost (Xiang et al., 2023). In a nutshell, it is a 3D U-Net that operates on the voxelised point cloud with sub-manifold sparse convolutions. The resulting per-point feature vectors of length 16 serve as input for three output branches: one that estimates point-wise semantic labels, one that regresses offsets to the instance center, and one that extracts instance-discriminative embedding features.

The **semantic segmentation branch** consists only of a multilayer perceptron (MLP) with a single hidden layer with *softmax* activations and outputs semantic class probabilities for each point. That branch is trained with a standard cross-entropy loss. Semantic labels are obtained by taking the *argmax* over the predicted category probabilities. Points assigned to categories that cannot be divided into well-defined instances (so-called "stuff" categories, like for instance "road" or "building facade") are ignored during instance segmentation.

The **centre offset branch**, advocated by several studies about instance segmentation (Jiang et al., 2020, Vu et al., 2022, Zhong et al., 2022, Chen et al., 2021), operates in 3D scene space: it takes as input the latent encoding extracted by the backbone and, for each point, predicts a 3D offset vector that would take that point to the estimated instance centre. I.e., if the predictions were perfect then shifting all points by their offsets would collapse each instance to a single point. The corresponding loss function is a combination of (1) the cosine distance between the true and predicted offset vectors and (2) the L_1 distance between their endpoints.

The **instance embedding branch** also ingests the latent encoding from the backbone. Instead of trying to find the geometric object centre, it embeds each point in a 5D feature space that is optimised to discriminate between instances. The embedding is supervised with a contrastive loss function that favours small distances between points from the same instance and large distances between points from different instances. Importantly, the embedding space has more than three dimensions, hence it has some spare capacity to represent object properties beyond being a compact cluster around a 3D centre point.

We found that the two ways of measuring point-to-instance affinities, either by regressing explicit centre offsets in geometric space (Jiang et al., 2020) or by contrastive embedding (De Brabandere et al., 2017, Wang et al., 2019), complement each other.

Algorithm 1: Block Merging

```
Input  : - list of blocks $B_i$
         - list of point indices $I_{i,j}$ for every instance $j$
           in each block $i$
         - overlap threshold $T_{\text{IoU}}$
Output: - global per-point label vector $P$
initialise all elements of $P$ to $-1$;
set instance counter $q \leftarrow 1$;
for every block $B_i$ do
    for every instance $I_{i,j}$ in $B_i$ do
        if all $P(I_{i,j}) = -1$ then
            set all $P(I_{i,j}) \leftarrow q$;
            $q \leftarrow q + 1$;
        else if all $P(I_{i,j}) \neq -1$ then
            continue;
        else
            $J_r \leftarrow$ instance in $P$ with highest IoU to $I_{i,j}$;
            $r \leftarrow$ instance label of $J_r$;
            if IoU$(J_r, I_{i,j}) > T_{\text{IoU}}$ then
                $P(I_{i,j} = -1) \leftarrow r$;
            else
                $P(I_{i,j} = -1) \leftarrow q$;
                $q \leftarrow q + 1$;
            end
        end
    end
end
```

In fact, instance segmentation based on local 3D point configurations must balance different a-priori expectations. On the one hand, points that form a compact structure surrounded by empty space are indeed likely to belong to the same object, and that situation is easy to encode in the form of centroid offsets – e.g., for a local region on an isolated car one can often guess the direction to the object centre just from the local surface shape. On the other hand, when objects are located near each other it becomes important to look past proximity – e.g., for a region of a forest canopy it is often easy to say which tree it belongs to, but nevertheless difficult to point to a clear object centre. This is why we employ both strategies.

The predicted offsets are simply applied to the 3D point coordinates to shift them to the estimated object centre, and then clustered into instance candidates by region growing with a distance threshold. Note, mapping the latent features to 3D offset vectors discards the semantic category information originally contained in the features. Hence, the clustering is constrained to only include points from the same category in a candidate instance. In the 5D embedding space, where distances do not have a direct geometric meaning, candidates are found with mean-shift clustering.

From the redundant set of instance candidates, we want to retain the subset that best explains the scene. To that end we train a network branch to predict how well each candidate matches a ground truth instance. This ScoreNet regresses the highest expected IoU between the candidate and any of the actual objects. It is a small 3D U-Net model on top of the backbone features, followed by max-pooling and a fully connected layer that outputs a scalar score between 0 and 1 per candidate.

2.3 Post-processing

After scoring we are left with an over-complete list of instance candidates, each consisting of a subset of the 3D point cloud, and equipped with an estimate of its goodness-of-fit to some actual object instance. These are post-processed into a final set of instances in the following way. First, clusters with very few points (in our implementation, <10; see Table 1 for a complete list of hyper-parameters) are discarded. Second, we perform non-maximum suppression (NMS) based on the predicted scores to get rid of redundant clusters. Third, clusters with low scores are also discarded. Having obtained our final estimate per cylindrical block, we run block merging to combine them into a single result for the entire region of interest, see Algorithm 1. In brief, the block merging re-assigns instance IDs such that they are globally unique, and greedily fuses instances that were split between different blocks.

Parameter	NPM3D	FOR-instance
Base learning rate of *Adam* optimizer	0.001	0.001
Batch size	4	4
Voxel side length (m)	0.12	0.2
Cylinder radius (m)	16	4
Region growing radius (m)	0.03	0.03
Mean-shift bandwidth$^\diamond$	0.6	0.6
Minimum cluster size (#points)	10	10
Score threshold for discarding clusters	0.6	0.5
IoU threshold during NMS	0.3	0.3
IoU threshold during block merging	0.01	0.01

$^\diamond$ Bandwidth is relative to the specified cluster radius of 1 unit in embedding space

Table 1. Default parameter settings.

After block merging we have a final segmentation of the voxel-grid subsampled point cloud into semantic categories and into object instances. As a final step, we upsample all labels back from the voxel-gridded point cloud to the complete, original one with the nearest-neighbour method. Instance labels for "stuff" classes that do not have well-defined instances are set to -1.

3. EXPERIMENTS

3.1 Experimental settings

Datasets. For our experiments we use two datasets, *NPM3D* (Roynard et al., 2018) and *FOR-instance*. NPM3D consists of mobile laser scanning (MLS) point clouds collected in four different regions in the French cities of Paris and Lille, where each point has been annotated with two labels: one that assigns it to one out of 10 semantic categories and another one that assigns it to an object instance. When inspecting the data, we found 9 cases where multiple tree instances had not been separated correctly (i.e., they had the same ground truth instance label). These cases were manually corrected using the CloudCompare software (https://www.cloudcompare.org, last accessed 03/2023), and 35 individual tree instances were obtained. Our variant of the dataset with 10 semantic categories and enhanced instance labels is publicly available.[2]

The FOR-instance dataset is a recent benchmark dataset from the forestry domain, aimed at tree instance segmentation and biophysical parameter retrieval. The point clouds were collected from drones equipped with survey-grade laser scanners such as the Riegl VUX-1 UAV and Mini-VUX. The dataset covers diverse regions and forest types across multiple countries. For our purposes, we removed points assigned to the category "outpoints" (i.e., partially observed tree instances on region borders), leaving us with only two semantic categories, *tree* and *non-tree* (where the latter includes the forest floor). The panoptic segmentation task thus becomes to separate trees from non-trees and to divide the tree class into individual instances.

[2] https://doi.org/10.5281/zenodo.8118986

Both datasets have been released only recently. Previous outdoor point cloud datasets either did not provide instance annotations or were too small to train deep neural networks, consequently there are hardly any baseline results to compare to. We have made both the data and our source code[3] publicly available for future reference.

Evaluation Metrics. Semantic segmentation quality is measured by the mean intersection-over-union (mIoU) across all categories. To assess instance segmentation we follow (Wang et al., 2019) and compute the mean precision (mPrec) and mean recall (mRec) over all instances, the corresponding F1-score (harmonic mean of precision and recall), as well as the mean coverage (mCov), defined as the average IoU between ground truth instances and their best-matching instance predictions. We also calculate a variant of mCov that weights instances by their ground truth point count (mWCov). For the combined panoptic segmentation quality we adopt the metrics proposed by (Kirillov et al., 2019), segmentation quality (SQ), recognition quality (RQ) and panoptic quality (PQ).

Implementation Details. Our source code is based on the Torch-Point3D library (Chaton et al., 2020). Unless explicitly specified for a given experiment, we use the default parameter values listed in Table 1. All experiments were conducted on a machine with 8-core Intel CPU, 8 GB of memory per core, and one Nvidia Titan RTX GPU with 24 GB of on-board memory.

3.2 Ablation studies on NPM3D

Experiments were conducted on NPM3D to investigate the effects of different hyper-parameters. In all ablation studies, the training portions of Lille1_1, Lille1_2, and Lille2 serve as training set and the test portion of Paris serves as test set.

Radius of cylindrical blocks. As explained, we sample local cylindrical regions from the data to keep computations tractable. A larger cylinder radius means more points, and thus more spatial context, and at the same time fewer incomplete objects and boundary effects; but also slower training and inference. Figures 2a illustrates the impact of the radius on instance segmentation and on semantic segmentation. As a general conclusion, the cylinder radius has little influence on the semantic segmentation quality (in terms of mIoU), seemingly a limited amount of local context is sufficient to categorise points. On the contrary, too small blocks markedly degrade instance segmentation (measured by the F1-score), confirming the intuition that it relies more on a complete view of object shape and layout. The performance metrics in Figure 2 refer to end-to-end segmentation performance from a system user view, after cylinder merging and upsampling to the original input point cloud. We point out that increasing the cylinder radius from 8 m to 20 m doubles the inference time for the complete set from 12 min to 24 min, and also the training takes roughly twice as long.

Mean-shift bandwidth. The discriminative training uses the two margins 0.5 and 1.5, meaning that in theory it should bring the feature vectors of all points on an instance to within 0.5 units of the associated cluster centre, whereas there should be a distance of at least 2×1.5 units between two cluster centres (De Brabandere et al., 2017). Based on these values we empirically determine the optimal bandwidth of the flat (rectangular) mean-shift kernel, see Figure 2b. Indeed, semantic segmentation performance and the closely related panoptic quality peak at a bandwidth of 0.5, whereas instance segmentation peaks at a slightly higher value of 0.6.

Setting ID	Instance generator			Use ScoreNet
	Raw 3D coords. + region growing	Shifted 3D coords. + region growing	5D embedding + meanshift	
I			✓	
II		✓		
III	✓	✓		✓
IV		✓	✓	✓
V	✓	✓	✓	✓

Table 2. Summary of the tested instance segmentation settings.

Voxel grid resolution. As expected, the overall trend is that point cloud analysis deteriorates with increasing voxel size (stronger down-sampling). As can be seen in Figure 2c, instance segmentation does not benefit from overly dense sampling and reaches its best performance at a voxel size of $12\times12\times12\,\mathrm{cm}^3$. Obviously, smaller voxels significantly increase the computational cost of both training and testing, Figure 2d. We note that the trade-off between resolution, cylinder radius and computational cost depends on the scene properties, c.f. 2e, which is why we chose different values for NPM3D and FOR-instance (Table 1).

3.3 Panoptic segmentation results for NPM3D

The focus of the present paper is on how to best segment object instances. We compare different designs of the instance clustering branches in Table 2. *Setting I* corresponds to only the discriminative embedding, without predicting and clustering 3D centroid offsets. Conversely, *setting II* only clusters based on the predicted centroid offsets and does not learn a discriminative embedding. *Setting III* denotes the configuration advocated by PointGroup (Jiang et al., 2020), where the clustering based on centroid offsets is complemented by clustering also the raw 3D points (before shifting them by the offset vectors), and the best instances are selected from the resulting, redundant set of clusters with a ScoreNet. *Setting IV* is the combination of centroid-based clustering and embedding feature clustering (again followed by a ScoreNet), as described above. Finally, *setting V* additionally includes clusters based on raw point coordinates, as advocated by (Jiang et al., 2020), on top of the two cluster sets of setting IV; thus further enlarging the candidate pool, but also making score-based pruning harder.

All results were computed with four-fold cross-validation: in turn, each sub-regions serves as test set once, whereas the other three are used for training. Then the predictions for all four regions are concatenated to obtain labels for the test dataset, and the performance metrics are calculated. The metrics for all five settings are given in Table 3. It can be seen that the proposed setting IV yields the best balance between precision and recall for instance segmentation (F1-score), as well as the best semantic segmentation (mIoU), and consequently also the highest PQ values. Clustering based solely on either offsets or embedding features significantly reduces precision. The clustering variant introduced by PointGroup, based on raw point coordinates, noticeably reduces recall. It appears that, for quite a number of object instances, the scan point distribution is too diffuse to delineate them. The results for setting V show that instance candidates based on raw points, surprisingly, not only miss many points but even distract from better, competing candidates. It appears that these poorly matching clusters inject noise into the ranking procedure. In other words, complementary methods to diversify the candidate set are only beneficial if the additional candidates are of sufficient quality.

Figure 3 illustrates the differences qualitatively for four representative examples. In Area 1, adjacent trash cans challenge the

[3] https://github.com/bxiang233/PanopticSegForLargeScalePointCloud

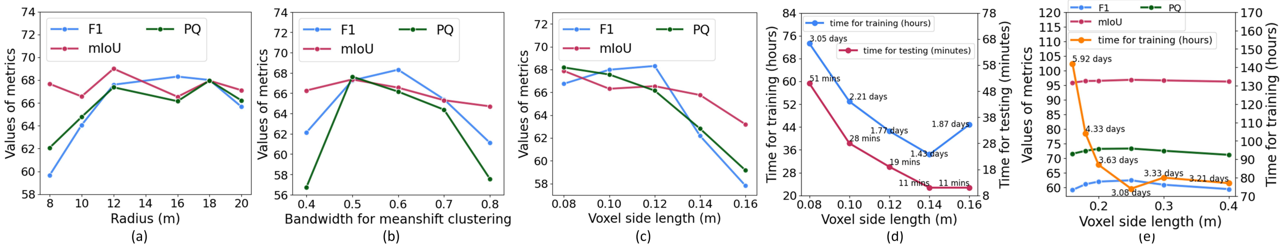

Figure 2. Ablation studies. Plots (a)-(d) refer to NPM3D, (e) refers to FOR-instance.

Setting ID	Instance segmentation					Semantic seg.	Panoptic seg.		
	mCov	mWCov	mPrec	mRec	F1	mIoU	RQ	SQ	PQ
I (5d embed)	<u>65.2</u>	<u>68.6</u>	61.5	<u>73.7</u>	67.1	75.0	77.9	<u>87.6</u>	68.1
II (3d offset)	62.5	65.9	40.5	71.8	51.8	73.0	66.9	85.5	56.9
III (3d offset + 3d raw)	59.6	63.1	<u>75.3</u>	68.6	71.8	72.8	80.8	86.6	69.9
IV (5d embed + 3d offset)	63.8	67.4	74.9	73.6	**74.3**	**75.8**	<u>82.6</u>	87.5	**72.1**
V (5d embed + 3d offset + 3d raw)	63.2	66.7	74.0	72.7	73.3	73.8	82.1	87.2	71.4

Table 3. Quantitative results on NPM3D for the five settings listed in Table 2. All values are percentages [%].

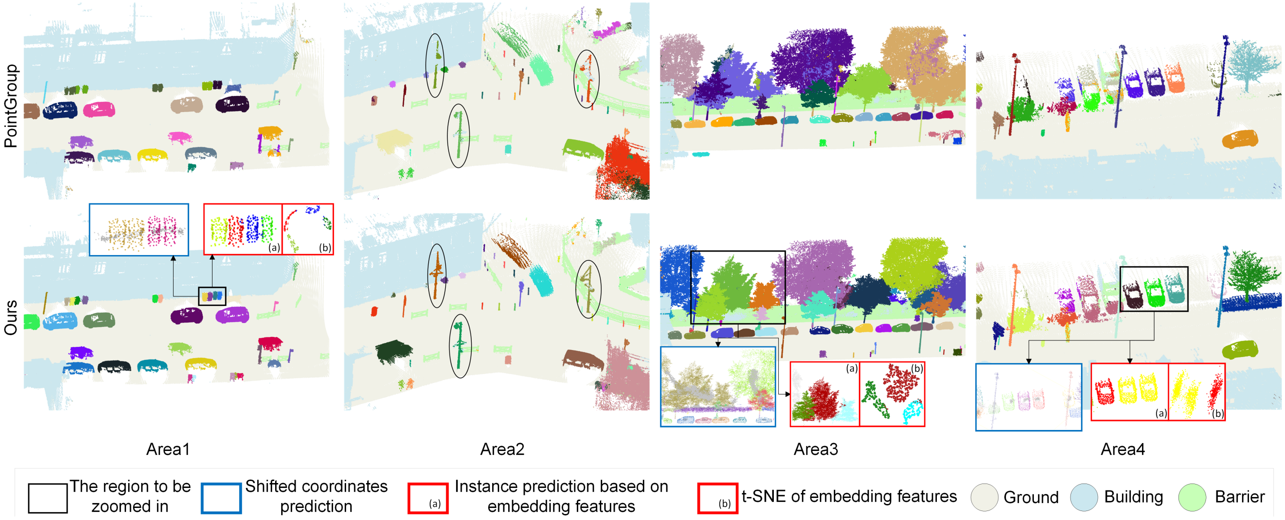
PointGroup
Ours
(a)
(b)
Area1
Area2
Area3
Area4
The region to be zoomed in
Shifted coordinates prediction
Instance prediction based on embedding features
t-SNE of embedding features
Ground
Building
Barrier

Figure 3. Example segmentations from the NPM3D dataset. See section 3.3 of the text for a discussion of these results. Light gray, light blue and light green colours denote uncountable "stuff" classes, saturated random colours denote instances.

instance segmentation. The centroid offset prediction fails to separate them, whereas discriminative embedding succeeds, as can be seen in the t-SNE projection of the 5D embedding space. Area 2 highlights a case where PointGroup suffers from its hard assertion that only points from the same semantic category can be clustered together. This over-reliance on the category labels means that instance segmentation cannot correct semantic segmentation errors, as on the streetlights marked by a black circle. Area 3 shows an example for the particularly challenging category of trees, which have large shape variability and are not delimited by well-defined surfaces. When they are located close to each other, the centroid method becomes unreliable, whereas they can still be separated in the discriminative embedding. Area 4 illustrates the opposite case, where the embedding features are unable to separate two cars, which sometimes occurs especially when there are many instances in close proximity. But since the cars are well enough separated, the centroid offsets are correctly predicted for most of their points and rectify the mistake.

3.4 Evaluation on FOR-instance

The FOR-instance dataset defines a canonical train/test split, to which we adhere. Within the training portion, we randomly set aside 25% of the data files as our validation set to monitor generalisation and hyper-parameters. As for NPM3D, we concatenate the results of all test sets and compute the performance metrics from that overall segmentation result.

Ablation of voxel size. Unsurprisingly, the rather simple segmentation between tree and non-tree points is hardly affected by the voxel grid filtering. But also instance segmentation performance is remarkably stable across a wide range of voxel sizes, Figure 2e. It reaches its maximum for voxels with side length 20 to 25 cm, but even at 40 cm the panoptic quality PQ drops less than 2.5 percent points under the maximum. Also very small voxels degrade performance only a little (likely because of diminished spatial context information, due to empty voxels), but significantly increase the training time. From our results, we do not see a reason to decrease the voxel size below $20{\times}20{\times}20\,\text{cm}^3$ for this application.

Ablation of cylinder radius. As shown in Table 4, expanding

Radius (m)	Instance segmentation [%]					Semantic seg. [%]	Panoptic segmentation [%]			Training time
	mCov	mWCov	mPrec	mRec	F1	mIoU	RQ	SQ	PQ	
4	65.2	78.1	58.4	65.9	61.9	96.5	81.0	88.9	73.2	3.6 days
8	68.7	81.0	69.2	68.7	**68.9**	**97.2**	84.5	90.6	**77.3**	7.8 days

Table 4. Ablation of cylinder radius for FOR-instance data.

	Instance segmentation					Semantic seg.	Panoptic segmentation		
	mCov	mWCov	mPrec	mRec	F1	mIoU	RQ	SQ	PQ
PointGroup	49.0	46.9	54.8	48.2	51.3	97.0	75.6	87.7	68.3
Setting IV (5D embed + 3D offset)	68.7	81.0	69.2	68.7	**68.9**	**97.2**	84.5	90.6	**77.3**

Table 5. Panoptic segmentation results on FOR-instance data. All values are percentages [%].

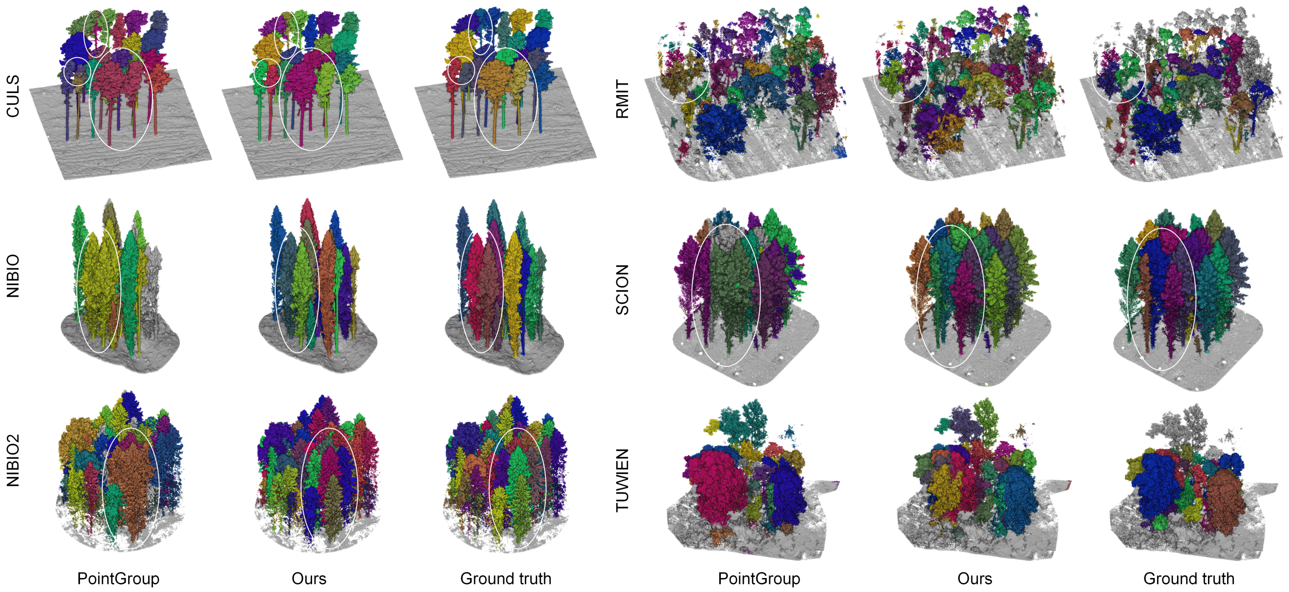

Figure 4. Example segmentations from six different regions within the FOR-instance dataset. Gray denotes non-tree points, other colours have been chosen randomly and indicate different instances.

the radius of the input blocks from 4 m to 8 m improves all performance metrics. The main reasons is that the bigger radius increases the chance of covering trees completely with a single block, leading to better instance segmentation. For forestry applications we therefore recommend to use rather large neighbourhoods, despite the significantly longer training time.

We also compare our preferred setting, with embedding and offset branches, block radius 8 m and voxel size 20 cm, to our implementation of the PointGroup method, see Table 5. We observe a marked improvement of all metrics with our proposed version, with over 17 percent points difference in F1-score. It appears that in the forest setting, where object centroids are hard to estimate and object boundaries are diffuse, the discriminative embedding has a clear advantage over clustering methods that operate in 3D geometric object space.

Figure 4 shows example results from different locations in the FOR-instance dataset. We note that both tested methods produce surprisingly compelling instance segmentations in most cases, across a range of forest characteristics. Still, our mixed clustering approach consistently yields results on par or better than PointGroup, see differences marked with white ellipses. FOR-instance was released only recently, and we are not aware of any other published results on the dataset. From the user perspective, we note that our pipeline achieves satisfactory instance segmentation without region-specific parameter tuning or post-processing, challenges commonly reported in the context of tree segmentation, e.g., (Wang et al., 2021, Chang et al., 2022, Wilkes et al., 2022).

4. CONCLUSION

We have studied the bottom-up approach to panoptic segmentation of outdoor 3D point clouds. We found that the bottleneck is the correct clustering of points into instances, and have constructed a pipeline with two complementary segmentation branches: one that is based on 3D centroid prediction and is well-suited for well-separated, compact objects; and a second one that is based on a discriminative embedding of the 3D points and better handles (nearly) contiguous objects and fuzzy object borders. In experiments on two different datasets, a contemporary panoptic segmentation pipeline with a carefully designed instance clustering stage was able to reach F1-scores >74% for objects in an urban mapping context and, remarkably, F1-scores >68% for trees in dense forest plots.

REFERENCES

Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S., 2016. 3D semantic parsing of large-scale indoor spaces. *IEEE/CVF Conference on Computer Vision and Pattern Recognition.*

Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., Gall, J., 2019. SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences. *IEEE/CVF International Conference on Computer Vision*.

Calders, K., Adams, J., Armston, J., Bartholomeus, H., Bauwens, S., Bentley, L. P., Chave, J., Danson, F. M. et al., 2020. Terrestrial Laser Scanning in Forest Ecology: Expanding the Horizon. *Remote Sensing of Environment*, 251, 112102.

Chang, L., Fan, H., Zhu, N., Dong, Z., 2022. A Two-Stage Approach for Individual Tree Segmentation From TLS Point Clouds. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15, 8682–8693.

Chaton, T., Chaulet, N., Horache, S., Landrieu, L., 2020. Torch-Points3D: A modular multi-task framework for reproducible deep learning on 3D point clouds. *International Conference on 3D Vision*.

Chen, C., Yang, B., 2016. Dynamic Occlusion Detection and Inpainting of In Situ Captured Terrestrial Laser Scanning Point Clouds Sequence. *ISPRS Journal of Photogrammetry and Remote Sensing*, 119, 90–107.

Chen, M., Hu, Q., Yu, Z., Thomas, H., Feng, A., Hou, Y., McCullough, K., Ren, F., Soibelman, L., 2022. STPLS3D: A large-scale synthetic and real aerial photogrammetry 3D point cloud dataset. *British Machine Vision Conference*.

Chen, S., Fang, J., Zhang, Q., Liu, W., Wang, X., 2021. Hierarchical aggregation for 3D instance segmentation. *IEEE/CVF International Conference on Computer Vision*.

Cheng, B., Misra, I., Schwing, A. G., Kirillov, A., Girdhar, R., 2022. Masked-attention mask transformer for universal image segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Choy, C., Gwak, J., Savarese, S., 2019. 4D spatio-temporal convnets: Minkowski convolutional neural networks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., Nießner, M., 2017. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. *IEEE/CVF Computer Vision and Pattern Recognition*.

De Brabandere, B., Neven, D., Van Gool, L., 2017. Semantic instance segmentation for autonomous driving. *IEEE Conference on Computer Vision and Pattern Recognition Workshops*.

Engelmann, F., Bokeloh, M., Fathi, A., Leibe, B., Nießner, M., 2020. 3D-MPA: Multi proposal aggregation for 3D semantic instance segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Fong, W. K., Mohan, R., Hurtado, J. V., Zhou, L., Caesar, H., Beijbom, O., Valada, A., 2022. Panoptic nuScenes: A Large-Scale Benchmark for LiDAR Panoptic Segmentation and Tracking. *IEEE Robotics and Automation Letters*, 7(2), 3795–3802.

Han, L., Zheng, T., Xu, L., Fang, L., 2020. OccuSeg: Occupancy-aware 3D instance segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

He, K., Gkioxari, G., Dollár, P., Girshick, R. B., 2017. Mask R-CNN. *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Jiang, L., Zhao, H., Shi, S., Liu, S., Fu, C.-W., Jia, J., 2020. PointGroup: Dual-set point grouping for 3D instance segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Kirillov, A., He, K., Girshick, R. B., Rother, C., Dollár, P., 2019. Panoptic segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Kolodiazhnyi, M., Rukhovich, D., Vorontsova, A., Konushin, A., 2023. Top-Down Beats Bottom-Up in 3D Instance Segmentation. *arXiv preprint arXiv:2302.02871*.

Lahoud, J., Ghanem, B., Pollefeys, M., Oswald, M., 2019. 3D instance segmentation via multi-task metric learning. *IEEE/CVF International Conference on Computer Vision*.

Liang, Z., Li, Z., Xu, S., Tan, M., Jia, K., 2021. Instance segmentation in 3D scenes using semantic superpoint tree networks. *IEEE/CVF International Conference on Computer Vision*.

Liu, J., He, T., Yang, H., Su, R., Tian, J., Wu, J., Guo, H., Xu, K., Ouyang, W., 2022. 3D-QueryIS: A Query-based Framework for 3D Instance Segmentation. *arXiv preprint arXiv:2211.09375*.

Roynard, X., Deschaud, J.-E., Goulette, F., 2018. Paris-Lille-3D: A Large and High-Quality Ground-Truth Urban Point Cloud Dataset for Automatic Segmentation and Classification. *International Journal of Robotics Research*, 37(6), 545–557.

Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., Leibe, B., 2023. Mask3D for 3D semantic instance segmentation. *International Conference on Robotics and Automation*.

Sun, J., Qing, C., Tan, J., Xu, X., 2023. Superpoint transformer for 3D scene instance segmentation. *Association for the Advancement of Artificial Intelligence*.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L., 2019. KPConv: Flexible and deformable convolution for point clouds. *IEEE/CVF International Conference on Computer Vision*.

Vu, T., Kim, K., Luu, T. M., Nguyen, T., Yoo, C. D., 2022. Soft-Group for 3D instance segmentation on point clouds. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Wang, D., Liang, X., Mofack, G. I., Martin-Ducup, O., 2021. Individual tree extraction from terrestrial laser scanning data via graph pathing. *Forest Ecosystems*, 8, 67.

Wang, X., Liu, S., Shen, X., Shen, C., Jia, J., 2019. Associatively segmenting instances and semantics in point clouds. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Wilkes, P., Disney, M., Armston, J., Bartholomeus, H., Bentley, L., Brede, B., Burt, A., Calders, K. et al., 2022. TLS2trees: a scalable tree segmentation pipeline for TLS data. *bioRxiv preprint bioArxiv:2022.12.07.518693*.

Xiang, B., Yue, Y., Peters, T., Schindler, K., 2023. A Review of Panoptic Segmentation for Mobile Mapping Point Clouds. *arXiv preprint arXiv:2304.13980*.

Yang, B., Wang, J., Clark, R., Hu, Q., Wang, S., Markham, A., Trigoni, N., 2019. Learning object bounding boxes for 3D instance segmentation on point clouds. *Advances in Neural Information Processing Systems*.

Zhang, F., Guan, C., Fang, J., Bai, S., Yang, R., Torr, P. H. S., Prisacariu, V., 2020. Instance segmentation of LiDAR point clouds. *IEEE International Conference on Robotics and Automation*.

Zhang, W., Pang, J., Chen, K., Loy, C. C., 2021. K-Net: Towards unified image segmentation. *Advances in Neural Information Processing Systems*.

Zhao, Y., Zhang, X., Huang, X., 2021. A technical survey and evaluation of traditional point cloud clustering methods for LiDAR panoptic segmentation. *IEEE/CVF International Conference on Computer Vision Workshops*.

Zhong, M., Chen, X., Chen, X., Zeng, G., Wang, Y., 2022. MaskGroup: Hierarchical point grouping and masking for 3D instance segmentation. *International Conference on Multimedia and Expo*.

Zhou, Z., Zhang, Y., Foroosh, H., 2021. Panoptic-PolarNet: Proposal-free LiDAR point cloud panoptic segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.