Contrastive Instance Association for 4D Panoptic Segmentation using Sequences of 3D LiDAR Scans

Rodrigo Marcuzzi Lucas Nunes Louis Wiesmann Ignacio Vizzo Jens Behley Cyrill Stachniss

Abstract—Scene understanding is critical for autonomous navigation in dynamic environments. Perception tasks in this domain like segmentation and tracking are usually tackled individually. In this paper, we address the problem of 4D panoptic segmentation using LiDAR scans, which requires to assign to each 3D point in a temporal sequence of scans a semantic class and for each object a temporally consistent instance ID. We propose a novel approach that builds on top of an arbitrary single-scan panoptic segmentation network and extends it to the temporal domain by associating instances across time. We propose a contrastive aggregation network that leverages the point-wise features from the panoptic network. It generates an embedding space in which encodings of the same instance at different timesteps lie close together and far from encodings belonging to other instances. The training is inspired by contrastive learning techniques for self-supervised metric learning. Our association module combines appearance and motion cues to associate instances across scans, allowing us to perform temporal perception. We evaluate our proposed method on the SemanticKITTI benchmark and achieve state-of-the-art results even without relying on pose information.

Index Terms-Semantic Scene Understanding, Deep Learning Methods

I. INTRODUCTION

N the context of mobile robotics and self-driving cars, spatial-temporal semantic scene understanding is critical since it is necessary to segment and identify the other agents in the scene but also understand how they behave over time. Different tasks related to dynamic scene understanding, i.e., object detection, semantic segmentation, instance segmentation, and multi-object tracking, are often tackled separately.

In this paper, we address the problem of 4D panoptic segmentation using LiDAR scans recorded by an outdoor platform, see also [1]. This requires to assign to each 3D point in a temporal sequence of scans a semantic class and a temporally consistent instance ID, as illustrated in Fig. 1.

In panoptic segmentation for LiDAR scans, state-of-theart methods [17], [12], [27] rely on their strong 3D feature extractors. We argue that these features can also be leveraged to generate temporally consistent instance-wise embeddings in order to extend the segmentation to the temporal domain by means of the tracking-by-detection paradigm: identify objects in individual 3D scans and then perform temporal instance associations via feature similarity.

This work has been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy, EXC-2070-390732324-PhenoRob. by the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017008 (Harmony). All authors are with the University of Bonn, Germany.

Digital Object Identifier (DOI): see top of this page.



Fig. 1: Our method allows to perform semantic segmentation (top) and temporally consistent instance segmentation (bottom) of 3D LiDAR scans. The different instance colors depict different IDs.

The main contribution of this paper is a novel method based on a sparse convolutional neural network, which takes 3D point cloud representation of LiDAR scans as input and outputs, for each point, a semantic label and a temporally consistent instance ID over the whole sequence. We build on top of a single-scan panoptic segmentation network to obtain the semantic and instance predictions as well as pointwise features. Our contrastive aggregation network takes these features as input to generate instance-wise embeddings that are consistent over time. It learns through a contrastive approach using as positive samples the same instance at several timesteps and as negative samples different instances in the same and other scans. Finally, our association module combines appearance and motion cues to perform the instance associations and generate consistent instance IDs over time. In sum, we propose an approach that is able to perform 4D panoptic segmentation of LiDAR scans using a frozen panoptic segmentation backbone and associating instances across time via feature similarity. We achieve state-of-the-art performance in 4D panoptic segmentation even without sensor pose estimates provided by a SLAM approach.

The implementation of our approach is publicly available at https://github.com/PRBonn/contrastive_association.

Manuscript received: Sep 9, 2021; Revised: Nov 24, 2021; Accepted: Dec 23, 2021. This paper was recommended for publication by Editor Markus Vincze upon evaluation of the Associate Editor and Reviewers' comments.

II. RELATED WORK

Our work targets outdoor environments, for tasks such as, perception in autonomous driving, for which the point clouds are extensively used [15].

Panoptic Segmentation on Point Clouds. To unify semantic and instance segmentation, Kirillov et al. [23] propose panoptic segmentation for images and Milioto et al. [27] extend it to LiDAR point clouds. This task boils down to performing semantic segmentation of stuff classes and instance segmentation for thing classes. To deal with 3D data, some approaches project it into a 2D representation like range images [27] or bird's-eye-view [38] and use 2D convolutional networks. Another alternative is to divide the space into 3D partitions to maintain the geometric relations between the data points and apply 3D convolutional networks. Gasperini et al. [12] use point convolutions [32] to improve the results but require a higher computational cost and applying hard downsampling. Recently, Hong et al. [17] use sparse convolutions [14] and improve results by voxelizing the space using cylindrical coordinates to match the distance-dependent density of the LiDAR point clouds.

Multi-Object Tracking. Multi-object tracking in the vision domain is usually solved using the tracking-by-detection paradigm [30], [36] formulated as a data association task consisting of two steps: obtaining object detections in the current frame and associating them across time. The performance of the task relies on the quality of the appearance and motion models [6]. When using deep learning to tackle this task, the focus of the research is on learning these models and their effective combination. Some works [36] only rely on a motion model and others [11], [31] use metric learning to generate instance embeddings for data association. They provide positive and negative samples to enforce that positive pairs of detections have more similar features than negative pairs. Dong et al. [11] use a triplet loss with one positive and one negative sample. Hermans et al. [16] use hard batch triplet loss to mine a hard negative sample which is more beneficial for learning. Son et al. [31] add an extra positive sample to increase temporal consistency. We propose to leverage instance labels to train our contrastive aggregation network with several positive and negative samples to compute temporally consistent instance features and perform associations via feature similarity.

Segmentation and Tracking. There have been a few works to unify tracking and segmentation tasks. Kim et al. [22] extend panoptic segmentation [23] from 2D images to the video domain focusing on short and sparsely labeled video snippets in an offline setting. Voigtlaender et al. [35] extend the multi-object tracking task with segmentation. Aygun et al. [1] propose to use as input 4D volumes by aggregating LiDAR scans and cluster instances across time and space to perform instance segmentation and tracking simultaneously. Recently, Hurtado et al. [18] proposed a new architecture that includes a tracking head to perform the instance associations across time. Similarly, we add instance associations to a panoptic segmentation network using the extracted features from the encoder-decoder but we do not retrain it. This allows our method to leverage features from arbitrary panoptic backbones. Furthermore, we train our approach using contrastive learning to leverage several positive and negative samples instead of just one.

Contrastive Learning. Contrastive learning for selfsupervised representation-learning [33], [8], [7] became popular due to the performance improvement on image classification tasks. In this setup, a data point is used as an anchor, positive samples are augmented views of it and the negative samples are the augmented views of other data points in the batch. The loss seeks to compare the generated encodings for each sample in an unsupervised way without labels. The goal is to learn a generic feature representation in which embeddings from positive samples lie close together and far from embeddings of negative samples. After this pretraining, the network is fine-tuned to solve a downstream task. Due to the availability of instance labels, we adopt the supervised contrastive loss [21] to use several positive along with many negative samples. However, the positive samples come from other timesteps of the same anchor instance and are not generated through augmentations of it.

In sum, our method is based on a panoptic segmentation backbone followed by temporal instance association via feature similarity learned using positive and negative samples.

III. OUR APPROACH

The goal of our approach is to achieve 4D panoptic segmentation of LiDAR scans, which means tackling simultaneously semantic and instance segmentation in the *3D spatial* and *1D temporal* domain. We achieve this by predicting, for each 3D point in the LiDAR scan, a semantic label, and for each object, a temporally consistent instance ID. To address this task, we follow prior work [36], [18] and use the tracking-by-detection paradigm [30] to identify objects in single 3D scans and then perform temporal associations via feature similarity.

Fig. 2 gives an overview of our approach. We adopt the point cloud representation of LiDAR scans, which represents each point by its 3D coordinates. We use a frozen panoptic segmentation network (backbone), to obtain semantic predictions, instance predictions, and features for every point in the scan. Using the instance predictions, we select the points and features belonging to each instance and apply our contrastive aggregation network (CA-Net) to produce instance-wise features. Finally, our association module assigns instance IDs, which are consistent throughout the whole temporal sequence, by combining appearance information from the instance-wise features and motion information using a constant velocity motion model. Combining this with the semantic predictions, we obtain 4D panoptic segmentation.

To be able to learn descriptive features, the CA-Net is trained with the supervised contrastive loss [21]. Using the instance labels, we identify the same object over time and use the same instance in different scans as positive samples. The negative samples are all other instances in the same and other scans in the training batch.



Fig. 2: Overview of our method. Given the current 3D LiDAR scan, we use a panoptic segmentation backbone B to obtain semantic classes, instance IDs, and features for each point. We select the points and features for each instance using the IDs and input them into the CA-Net to obtain instance-wise features. These features are used to perform instance associations using our association module to assign temporally consistent instance IDs. Combining this with the semantic predictions, we obtain 4D panoptic segmentation.

A. Panoptic Segmentation Backbone

The first element of our approach is a frozen panoptic segmentation network to obtain semantic predictions, instance predictions, and point-wise features. To achieve panoptic segmentation, bottom-up methods [27], [12], [38], [17] use a shared feature extractor to obtain point-wise features based on convolutions and apply task-specific heads to obtain semantic and instance features. The instance features are clustered in a post-processing step to produce instances.

Given a point cloud $\mathcal{P} = \{\mathbf{p}_1^C, \dots, \mathbf{p}_N^C\}$ with point coordinates $\mathbf{p}_i^C \in \mathbb{R}^3$, we apply the backbone B and obtain point-wise semantic classes $\mathcal{P}^S = \{p_1^S, \dots, p_N^S\}$, where $p_i^S \in \{1, \dots, n_{\text{classes}}\}$ and point-wise instance IDs $\mathcal{P}^I = \{p_1^I, \dots, p_N^I\}$ with $p_i^I \in \mathbb{N}$ from the task specific heads. From the feature extractor, we obtain the point-wise features $\mathcal{P}^F = \{\mathbf{p}_1^F, \dots, \mathbf{p}_N^F\}$, where $\mathbf{p}_i^F \in \mathbb{R}^{D_B}$ with feature dimension D_B . We use the ID of each point to select for each instance $j \in \{1, \dots, M\}$, its points $I_j^P = \{p_i^C \in \mathcal{P} \mid p_i^I = j\}$ and its point-wise features I_j^F , which we use as input of our CA-Net.

In this paper, we use DS-Net [17] as backbone B. Since we are not modifying nor retraining the backbone and only use it to get predictions and features, our approach can potentially be applied to other panoptic segmentation networks.

B. Contrastive Aggregation Network

Our contrastive aggregation network (CA-Net) generates temporally consistent appearance features for instance associations. Given the input points I_j^P and point-wise features I_j^F for all M instances in the current point cloud \mathcal{P} , the output of the network are instance-wise features $I^F = {\mathbf{f}_1, \ldots, \mathbf{f}_M}, \mathbf{f}_j \in \mathbb{R}^{D_A}$ with feature dimension D_A , i.e., a single feature vector for each instance in the scan.

The point-wise features from the backbone capture more general and high-level information, not specific to the instances. We are only interested in encoding information related to the individual instances. To learn from the shape of the instances and the relations between their points, we first apply three convolutional blocks with stride 2 to reduce the spatial dimension while increasing the feature dimension. This increases progressively the receptive field and allows us to capture low-resolution features to get information about the whole instance. After applying the convolutions, a pooling layer computes a single feature vector from all the features belonging to the instance points I_j^P , which summarizes the information of each point in a single instance feature. This is followed by two linear blocks and a projection head to obtain the final instance-wise feature \mathbf{f}_j . The goal is to project the intermediate feature representation into an embedding space, where embeddings belonging to the same instance across time have a high similarity and embeddings belonging to different instances have low similarity.

Due to the inherent sparsity of the data, we use sparse convolutions [9] to process the voxelized scan as it leads to better performance and reduces computational cost [14].

C. Association Module

After computing instance-wise embeddings I^F using our CA-Net, we maintain consistent instance-wise IDs by associating the instances $\{I_1 \ldots I_M\}$ in the current scan with the corresponding instances $\{I_1 \ldots I_K\}$ in the previous ones. To achieve this, we follow a similar procedure as previous works [36], [6]. We compute a cost matrix $\mathbf{C} \in \mathbb{R}^{M \times K}$ between all M instances in the current scan and all K instances identified in previous scans, by means of the similarity between their features. Then, the problem can be formulated as a bipartite graph matching problem that can be solved using the Hungarian method [24] to determine the pairs of associations between current and previous instances.

Tracking instances in consecutive frames is only part of the problem. We need to manage tracked objects that might leave the scene, newly detected objects that are not explained by any trajectory and re-identify objects, which have been occluded or missed in intermediate frames. To handle new targets, we add detected objects in the current frame only if the feature similarity with any of the already active instances is lower than a threshold T_{new} . For re-identification of objects, we store the deactivated trajectories for a fixed number of scans n_{old} and compare with the deactivated targets. We re-identify objects if the similarity is higher than a threshold T_{old} .

To add information about the movement of the targets, we apply a constant velocity assumption for all objects by adding a constant velocity motion model independent of the sensor ego-motion. This means that the model does not estimate the ego-motion but the motion of all the objects in the scene. We fuse the appearance and motion information in the cost matrix as a linear combination of the feature cost and the center distance between current instances and the predicted positions of the previous instances. Each entry of the cost matrix C is an association cost between the new instance m and the previous instance k:

$$\mathbf{C}_{m,k} = \alpha_f \cdot (1 - sim(\mathbf{f}_m, \mathbf{f}_k)) + \alpha_d \cdot \|\mathbf{c}_m - \mathbf{c}_k\|, \quad (1)$$

where \mathbf{f}_m is the feature and $\mathbf{c}_m \in \mathbb{R}^3$ the center coordinates of instance m, $sim(\cdot)$ is the cosine similarity function $sim(\underline{\mathbf{f}}_m, \underline{\mathbf{f}}_k) = \underline{\mathbf{f}}_m^\top \underline{\mathbf{f}}_k / \|\underline{\mathbf{f}}_m\| \|\underline{\mathbf{f}}_k\|$, and α_f , $\alpha_d \in \mathbb{R}$ are importance weights for the individual feature and distance costs. For re-identification, the motion model is applied continuously also to the deactivated targets (not associated with any other instance) to estimate their position in case of occlussions or missing detections.

D. Input Features

The point-wise features \mathcal{P}^F from the panoptic segmentation backbone are used by the task-specific heads to perform segmentation of the points. They are similar for instances belonging to the same semantic class and thus cannot be directly used to perform instance associations for objects of the same semantic class.

Inspired by the recent findings in natural language processing [34], we seek to enhance the point-wise features with spatial knowledge using positional encodings. We compute for each instance point $\mathbf{p}_l \in I_i^P$ a fixed positional encoding $\mathbf{e}_l \in \mathbb{R}^{D_B}$. We generate for each point a Fourier feature position encoding [20] with the same dimension as the pointwise features. We select a maximum frequency max_{freq} and sample, for each coordinate, from a series of sines and cosine functions with frequencies evenly spaced in $[0, \max_{\text{freq}}]$ on a logarithmic scale. Then, we use a padding of zeros to reach the same dimension as the point feature. To get the final input to the network, the feature and the positional encoding are combined by performing the addition of both. Due to their similarity, using the point-wise features as the only input to our CA-Net does not provide enough information to learn distinct instance-wise features to do the associations. By adding the positional encodings as extra information, we exploit the spatial relations between the instances and create more distinct features, and help the network learn better instance-wise features for this task.

E. Instance Point Extraction

As the input of the CA-Net, we select from all the points and features (P, P^F) in the current point cloud \mathcal{P} , the ones belonging to the different instances (I^P, I^F) . During training, we rely on the instance labels $\hat{\mathcal{P}}^I$ to group the points into instances and select their corresponding features. At inference time the instance points are selected using the predictions P^I from the backbone. These predictions are not perfect due to problems like wrong semantic predictions or errors in the clustering algorithm used to separate instances in the backbone after the instance head. The inputs for the CA-Net are then different at train and test time.

We can circumvent this by applying augmentations to the point instances to imitate what may happen during inference. We design specific augmentations to deal with the problems commonly observed at test time, mainly splitted instances and incomplete instances. Compared to other kinds of augmentations applied to the full point cloud, in our case we implement instance-wise augmentations since these are the input to our network.

The problem of splitted instances is due to the imperfect clustering applied in the backbone to obtain the instance predictions. It can wrongly cluster the points into instances, dividing one instance into two smaller ones, as shown in Fig. 3. To mimic this, we create an augmentation, which we call *split*, to separate the points on each side of an imaginary plane to split the instance. We sample a random unit normal vector $\mathbf{n} \in \mathbb{R}^3$, $\|\mathbf{n}\| = 1$ and a random instance point $\mathbf{p}_l \in I_j^P$ to generate a randomly oriented plane with Hessian normal form $\mathbf{n}^\top \mathbf{p}_l = d$, where d is the distance from the plane to the origin. The remaining points after the augmentation are the query points \mathbf{q}_l which lie in the half-space of the normal:

$$I_{aug} = \left\{ \mathbf{q}_l \in I_j^P \mid \mathbf{n}^\top \mathbf{q}_l^C - d > 0 \right\},\tag{2}$$

which leads to splitted instances.

The incomplete instances are caused by the wrong semantic class predicted for some instance points, usually at the borders of the instances. Their semantic class assigns them to the background, and they are not considered as part of the instance, as can be seen in Fig. 3. To address it, we create an augmentation, which we call *contour*, to discard points in the contour of the instances. We first normalize the point coordinates to the range [-1,1], generate a random maximum coordinate value $\gamma \in \mathbb{R}$ and save the indices of the points with smaller absolute coordinates that γ . We use these indices to select the instance points (with unnormalized coordinates) to keep after the augmentation, i.e.,

$$I_{aug} = \left\{ \mathbf{p}_l \in I_j^P \mid |\tilde{x}_l| < \gamma \land |\tilde{y}_l| < \gamma \land |\tilde{z}_l| < \gamma \right\}, \quad (3)$$

where $\tilde{x}_l, \tilde{y}_l, \tilde{z}_l$ are the normalized coordinate components of l^{th} point.

We illustrate our augmentations applied to instances in Fig. 3. It is important to notice that the augmentations do not change the position of any point but rather drop some of them. As an extra augmentation, we randomly drop cuboids of points belonging to the instance [37]. We discuss the influence of these augmentations in Sec. V-C.

F. Pose Information

Both, the positional encoding, which adds spatial knowledge to the computed features, and the constant velocity motion model, rely on the positions of the instances in the current scan, i.e., positions in a local coordinates frame. As these are local coordinates, the positions are not consistent for scans at different timesteps and the ego-motion of the sensor must be compensated.

By adding the sensor pose estimates using a SLAM approach [5], we can improve the instance associations by leveraging this extra information. We use the pose estimates to transform the positions of previously observed objects into the current local coordinates frame. This way, objects have



Fig. 3: Problems in the instance predictions using the backbone. Incomplete, splitted and perfect instance (top). Proposed *split* augmentation (left) and *contour* augmentation (right). The gray points are discarded.

consistent positions over time and the constant velocity motion model only predicts the motion of the other objects as the ego-motion is compensated. We recompute the features of the previously observed instances by updating their positional encoding using the consistent positions in the current coordinate frame. We obtain more similar features for the same instance at different points in time, which improves the instance associations as illustrated in Sec. IV-B

G. Contrastive Training

For the multi-object tracking problem, we use the instance head to get the predicted instances $\{I_1, \ldots, I_M\}$ in the current point cloud \mathcal{P} and associate them with previous instances across time. Several works [18], [16], [11], [31] use metric learning to learn instance representations by comparing the embedding of one anchor object with one or a few positive and negative samples. Contrastive learning is used in selfsupervised representation learning [33], [8] to train a network, which is later fine-tuned on a downstream task. The main idea of those approaches is to use data augmentation to generate two versions of one anchor sample and train the network to learn to pull the embeddings of these samples (positives) together and push them away from all other samples (negatives).

Given a batch of samples \mathcal{B} , the loss function seeks to discriminate between the positive pairs (augmented versions of the anchor) and the negatives (augmented versions of different anchors). In the self-supervised contrastive loss InfoNCE [33], an encoder is used to obtain the feature vectors \mathbf{z}_j for each augmented sample j. Let r(j) be the index of the other augmented sample from the same anchor j and A(j) the set of all indices in the batch except j. Then, the loss function takes the form:

$$\mathcal{L}_{self} = -\sum_{j \in \mathcal{B}} \log \frac{\exp\left(\mathbf{z}_{j}^{\top} \mathbf{z}_{r(j)} / \tau\right)}{\sum_{a \in A(j)} \exp\left(\mathbf{z}_{j}^{\top} \mathbf{z}_{a} / \tau\right)},$$
(4)

where τ is a temperature parameter.

In our setup, the samples are all instances in the batch and their corresponding feature vectors $\{\mathbf{f}_1, \ldots, \mathbf{f}_b\}$ are computed using our CA-Net. We want to enforce that features of the



Fig. 4: In our contrastive setup, for each instance, positives samples are the same instance in different scans (green) in a sequence, and negatives samples are all the other instances (red) in all scans and all other sequences in the batch.

same instance are consistent. The appearance of the instances changes over time as the viewpoint changes due to the instance's and the sensor's motion. As the samples for the same instance are different across time but depict the same object, we can use this as an implicit augmentation. To obtain positive samples, we select the same instance in different scans over time instead of using one instance as an anchor and generating augmented versions of it.

The appearance can, however, change significantly in scans temporally far from each other. We define a temporal window of scans $\Delta \in \mathbb{N}$, in which we consider instances similar enough to perform the associations and from which the positive samples are drawn. As Δ is the number of scans from which we extract instances, it is the maximum number of positive samples to consider in the loss function. Using the labels, we select a set of positive samples P(j) considering the instances with the same ID in consecutive scans in the temporal window. As negative samples, we use all other instances in the batch, as depicted in Fig. 4. To learn from many positive in addition to the many negative samples, we use the supervised contrastive loss [21]:

$$\mathcal{L}_{sup} = -\sum_{j \in \mathcal{B}} \frac{1}{|P(j)|} \sum_{p \in P(j)} \log \frac{\exp\left(\mathbf{f}_{j}^{\top} \mathbf{f}_{p} / \tau\right)}{\sum_{a \in A(j)} \exp\left(\mathbf{f}_{j}^{\top} \mathbf{f}_{a} / \tau\right)}, \quad (5)$$

where $\mathbf{f}_j, \mathbf{f}_p, \mathbf{f}_a \in \mathbb{R}^{D_A}$ are respectively the feature for the anchor instance, the feature for the positive samples and, the feature for all the other instances in the batch.

We are leveraging the properties of the supervised contrastive loss for our task. First, we can use an arbitrary number of positives which are determined by the temporal window. Second, we increase the number of negatives compared to other losses by considering all other instances in the same and other scans. Third, we benefit from the intrinsic ability of the loss to perform hard positive/negative mining and avoid the need for explicit hard negative mining.

Method	LSTQ	Sassoc	Scls	IoUSt	IoU Th
RangeNet++[28] + PP + MOT	35.52	24.06	52.43	64.52	35.82
KPConv [32] + PP + MOT	38.01	25.86	55.86	66.90	47.66
RangeNet++[28] + PP + SFP	34.91	23.25	52.43	64.52	35.82
KPConv [32] + PP + SFP	38.53	26.58	55.86	66.90	47.66
4DPLS[1]	56.89	56.36	57.43	66.86	51.64
Ours (without pose estimates)	60.04	59.49	60.60	66.88	51.98
Ours (with pose estimates[5])	63.11	65.71	60.60*	66.88*	51.98*

TABLE I: 4D panoptic segmentation on SemanticKITTI test set. MOT [36]; SFP - scene flow based propagation [29]; PP - Point-Pillars [25]. Numbers with * denote the same segmentation results. Adding pose information to our single-scan panoptic backbone does not influence the segmentation performance.

H. Implementation Details

We build on top of DS-Net [17] as panoptic segmentation backbone. The feature extractor provides point-wise feature vectors $\mathbf{p}_i^F \in \mathbb{R}^{D_B}$, $D_B = 128$. For the instance clustering, we use mean shift [10].

For our CA-Net, the convolutional blocks are 3D sparse convolutions [9], followed by a batch normalization layer [19] and leaky ReLU [26] as activation layer. The sparse linear blocks consist of a linear layer followed by a batch normalization layer and a leaky ReLU as activation layer. The final projection head is a linear layer that projects the intermediate instance-wise embeddings into a feature space of dimension $D_A = 1024$ in which we use feature similarities to associate instances across time.

We keep deactivated targets for $n_{old} = 8$ frames and use different thresholds to handle new targets and re-identification for the appearance and the motion model. For the feature similarity, we use $T_{new_f} = T_{old_f} = 0.7$ and for the center distance, we use $T_{new_d} = T_{old_d} = 2$. The weights for the feature cost and the center distance in Eq. (1) are $\alpha_f = 0.4$ and $\alpha_d = 0.7$. In the loss function, see Eq. (5) we use $\tau = 0.1$. At each step, the positive and negative samples are selected from a sequence of scans of random length $\Delta \in [2, 5]$. This way, at each step, the loss considers samples from sequences of variable length, which provides a different number of instances.

IV. EXPERIMENTAL EVALUATION

We present our experiments to show the capabilities of our method and to support our statement that our approach achieves state-of-the-art performance in 4D panoptic segmentation of LiDAR scans. Furthermore, we show that using a frozen panoptic segmentation backbone and associating instances across time via feature similarity allows us to outperform previous methods that strongly rely on ego-motion estimates provided by SLAM techniques. Moreover, we illustrate that the performance of our approach increases when including the pose information.

A. Experimental Setup

We evaluate our method on SemanticKITTI [3], [2], which consist of 22 sequences from the KITTI odometry dataset [13]. Sequences 00 to 10 are used for training, leaving sequence 08 as validation set and the test set consists of sequences



Fig. 5: Histograms of similarities between instances features in a sequence of scans. Without positional encoding (top), the histogram shows a high density at high similarities. Hence, there is a large number of instances with similar features, which makes it difficult to identify the same instance in different scans as different other instances in the sequence have similar features. With the positional encoding (bottom), most of the feature similarities are in the middle range and only a few features have high similarity. Therefore, features from different instances are more distinct, which makes them better suited for associating instances.

11 to 21. The provided annotations are point-wise semantic and instance labels [4]. To evaluate our performance, we use the LiDAR Segmentation and Tracking Quality (LTSQ) metric $LSTQ = \sqrt{S_{cls} \times S_{assoc}}$ [1]. It consists of two terms, the classification score S_{cls} and the association score S_{assoc} . Since our panoptic segmentation backbone is frozen and we use its single-scan predictions, our results do not change the segmentation results represented by the S_{cls} term. Thus, we focus instead on the S_{assoc} term to evaluate the quality of our associations.

B. 4D Panoptic Segmentation Results

The first experiment evaluates the performance of our approach on the SemanticKITTI test set and supports the claim that we achieve state-of-the-art results on the 4D panoptic segmentation task. In this experiment, we compare our approach with the recent and high-performing method by Aygun et al. [1] and baselines using semantic segmentation networks [32], [28] and combining the predictions from the PointPillars (PP) object detector [25] with a constant velocity motion model [36] or using scene flow propagation (SFP) [29]. Note that the approach by Aygun et al. [1] and the baselines rely on the sensor poses to either aggregate scans or transform them into a global coordinates frame where objects have consistent positions and are easier to associate. For these approaches, this knowledge is a pre-requisite.

We outperform all previous methods *without* relying on the pose estimations from a SLAM approach. See Tab. I. Both experiments using our method show the same segmentation performance S_{cls} , IoU^{St} , and IoU^{Th} since we use the semantic predictions from the same single-scan backbone and the pose information does not modify semantic segmentation results of *a single scan*. More detailed results are shown in Tab. I.

instance feature		max	average	~	
backbone features	positional encoding	pooling	pooling	CA-Net	
\checkmark		15.5	31.8	59.9	
	\checkmark	32.1	52.2	58.4	
\checkmark	\checkmark	36.9	59.5	70.4	

TABLE II: Association performance S_{assoc} using different instancewise features for association in SemanticKITTI validation set.

V. ABLATION STUDIES

A. Positional Encoding

First, we analyze the influence of adding the positional encoding to the point-wise features.

We show the similarity between point-wise features from the backbone. To this end, we generate instance-wise features f_i by averaging the point-wise features for each instance:

$$\mathbf{f}_j = \frac{1}{|I_j^F|} \sum_{\mathbf{p}_l \in I_j^P} \mathbf{p}_l^F,\tag{6}$$

and compute their similarities.

In a sequence of scans, there is a large number of highly similar instance features, which makes it hard to distinguish them, as shown Fig. 5 (top).

To illustrate how the spatial encoding helps, we add the positional encoding generated from the point coordinates as explained in III-D, to each point-wise feature and then average them to get a instance feature:

$$\mathbf{f}_{j} = \frac{1}{|I_{j}^{F}|} \sum_{\mathbf{p}_{l} \in I_{j}^{P}} \mathbf{p}_{l}^{F} + \mathbf{e}_{l}, \tag{7}$$

where $\mathbf{p}_l^F \in \mathbb{R}^{D_B}$ is the point-wise feature from the backbone and $\mathbf{e}_l \in \mathbb{R}^{D_B}$ is the positional encoding for the l^{th} instance point. As can be seen in Fig. 5 (bottom), when adding the positional encoding, the amount of similar instance-wise features is visibly reduced. Thus, instances of different objects have a large distance in the feature space.

B. Feature Design

In the next experiment, we have a closer look at our feature design decisions. We generate instance-wise feature vectors and associate the instances using only their features similarity, without relying on any other information to better show the performance. Tab. II shows the association performance of the different feature options, namely pooling the features and positional encodings and using them as input to the CA-Net to obtain the instance-wise features.

As reflected in the experiments, the information about the position of the instances plays a crucial role in the association process. However, the features from the backbone allow us to exploit some extra information but, as discussed in Sec. III-D, they are similar for objects of the same semantic class and adding a positional encoding improves the performance. Lastly, adding the positional encoding to each point $\mathbf{p}_l + \mathbf{e}_l$, we obtain even better results. This shows that both parts are important and that leveraging the point-wise features from the backbone improves the instance association performance.

augmentations			~
split	contour	cuboids	$S_{ m assoc}$
			49.9
\checkmark			60.9
	\checkmark		54.6
		\checkmark	52.6
\checkmark	\checkmark		62.3
\checkmark	\checkmark	\checkmark	63.6
i	nstance prediction	18	50.2

TABLE III: Influence of the selection of input points and the different augmentations in the association quality S_{assoc} .

#	feature	encoding	motion model	poses	$S_{\rm assoc}$
А	\checkmark				59.9
В	\checkmark	\checkmark			70.4
С	\checkmark	\checkmark		\checkmark	71.2
D			\checkmark		68.0
Е	\checkmark	\checkmark	\checkmark		71.7
F	\checkmark	\checkmark	\checkmark	\checkmark	72.9

TABLE IV: Influence of the different components of the approach in S_{assoc} on SemanticKITTI validation set.

C. Instance Augmentations

In this experiment, we show that using instance labels to select point-wise features can lead to suboptimal results and how our proposed augmentations improve this. For this experiment, we fix the temporal window $\Delta = 3$, do not use the sensor poses nor motion model, and perform the associations using only feature similarity.

As shown in Tab. III, training using the labels to select the instance points gives the worst results and using the predictions only helps to increase S_{assoc} by 0.3 percent points. Each individual augmentation outperforms training with the labels or the predictions. Particularly, the *split* augmentation has the strongest influence. Combining all three augmentations results in a 13.7 percent points improvement compared to the training using only the labels.

D. Method Components

Tab. IV shows the influence of different components on on the performance in terms of S_{assoc} on the validation set.

In (A), we only use the point-wise features from the backbone as input to our CA-Net and perform the associations via instance feature similarity. Since no spatial information is included, instances at different positions in the scan can be wrongly associated. In (B), we add the positional encoding to the input features, see Sec. III-D, which adds spatial information, visibly improving the results. In (C), we add the SLAM poses [5] to update the previous instance positional encoding, see Sec. III-F. This way, objects have consistent positions and are easier to associate. In (D), we use only the constant velocity motion model to perform the associations. This result is better than (A) and shows that motion and appearance models are necessary to achieve good association results. In (E), we add the motion model to (B). We combine cues from the appearance and motion models, improving the results obtained when the models are used separately. In (F), we add the sensor ego-motion poses estimates and obtain the best performance.

VI. CONCLUSION

We presented a novel approach to perform 4D panoptic segmentation on 3D LiDAR scans. We identify objects in the current scene and associate them with previously seen targets across time combining appearance and motion information. We show that it is possible to leverage the point-wise features from a single feature extractor to tackle concurrently the tasks of semantic and instance segmentation over time. Furthermore, generating a descriptive appearance model improves the instance association and allows our approach to outperform previous methods strongly relying on ego-motion estimates. We evaluated the different parts of our approach and provided comparisons to other existing techniques. The experiments suggest that we are able to learn a descriptive representation for each instance that in turn allows us to perform temporal associations and extend a panoptic segmentation backbone to perform 4D panoptic segmentation.

REFERENCES

- M. Aygun, A. Osep, M. Weber, M. Maximov, C. Stachniss, J. Behley, and L. Leal-Taixe. 4D Panoptic LiDAR Segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [2] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, J. Gall, and C. Stachniss. Towards 3d lidar-based semantic scene understanding of 3d point cloud sequences: The semantickitti dataset. *Intl. Journal of Robotics Research (IJRR)*, 40(8-9):959–967, 2021.
- [3] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV), 2019.
- [4] J. Behley, A. Milioto, and C. Stachniss. A Benchmark for LiDARbased Panoptic Segmentation based on KITTI. In Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA), 2021.
- [5] J. Behley and C. Stachniss. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In Proc. of Robotics: Science and Systems (RSS), 2018.
- [6] P. Bergmann, T. Meinhardt, and L. Leal-Taixe. Tracking Without Bells and Whistles. In Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV), 2019.
- [7] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9912–9924, 2020.
- [8] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2020.
- [9] C. Choy, J. Gwak, and S. Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2019.
- [10] D. Comaniciu and P. Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Trans. on Pattern Analalysis and Machine Intelligence (TPAMI)*, 24:603–619, 2002.
- [11] X. Dong and J. Shen. Triplet Loss in Siamese Network for Object Tracking. In Proc. of the Europ. Conf. on Computer Vision (ECCV), 2018.
- [12] S. Gasperini, M.N. Mahani, A. Marcos-Ramiro, N. Navab, and F. Tombari. Panoster: End-To-End Panoptic Segmentation of LiDAR Point Clouds. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):3216– 3223, 2021.
- [13] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2012.
- [14] B. Graham, M. Engelcke, and L. van der Maaten. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2018.
- [15] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun. Deep Learning for 3D Point Clouds: A Survey. *IEEE Trans. on Pattern Analalysis and Machine Intelligence (TPAMI)*, 2020.

- [16] A. Hermans, L. Beyer, and B. Leibe. In Defense of the Triplet Loss for Person Re-Identification. arXiv preprint:1703.07737, 2017.
- [17] F. Hong, H. Zhou, X. Zhu, H. Li, and Z. Liu. LiDAR-Based Panoptic Segmentation via Dynamic Shifting Network. In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2021.
- [18] J. Hurtado, R. Mohan, W. Burgard, and A. Valada. MOPT: Multi-Object Panoptic Tracking. arXiv preprint:2004.08189, 2020.
- [19] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint, abs/1502.03167, 2015.
- [20] A. Jaegle, F. Gimeno, A. Brock, A. Zisserman, O. Vinyals, and J. Carreira. Perceiver: General Perception with Iterative Attention. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2021.
- [21] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan. Supervised Contrastive Learning. In Proc. of the Conference on Neural Information Processing Systems (NeurIPS), volume 33, pages 18661–18673, 2020.
- [22] D. Kim, S. Woo, J. Lee, and I.S. Kweon. Video Panoptic Segmentation. In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2020.
- [23] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic Segmentation. In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2019.
- [24] H. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.
- [25] A. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. PointPillars: Fast Encoders for Object Detection From Point Clouds. In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2019.
- [26] A.L. Maas, A.Y. Hannun, and A.Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning* for Audio, Speech and Language Processing, 2013.
- [27] A. Milioto, J. Behley, C. McCool, and C. Stachniss. LiDAR Panoptic Segmentation for Autonomous Driving. In Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), 2020.
- [28] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss. RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. In *Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2019.
- [29] H. Mittal, B. Okorn, and D. Held. Just Go With the Flow: Self-Supervised Scene Flow Estimation. In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2020.
- [30] Reid, Donald B. An algorithm for tracking multiple targets. *IEEE Trans.* on Automatic Control, 1979.
- [31] J. Son, M. Baek, M. Cho, and B. Han. Multi-Object Tracking With Quadruplet Convolutional Neural Networks. In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2017.
- [32] H. Thomas, C. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas. KPConv: Flexible and Deformable Convolution for Point Clouds. In Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV), 2019.
- [33] A. van den Oord, Y. Li, and O. Vinyals. Representation Learning with Contrastive Predictive Coding. arXiv preprint:1807.03748, 2019.
- [34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need. Proc. of the Conference on Neural Information Processing Systems (NeurIPS), 2017.
- [35] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B.B.G. Sekar, A. Geiger, and B. Leibe. MOTS: Multi-Object Tracking and Segmentation. In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2019.
- [36] X. Weng, J. Wang, D. Held, and K. Kitani. 3D Multi-Object Tracking A Baseline and New Evaluation Metrics. In Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), 2020.
- [37] Z. Zhang, R. Girdhar, A. Joulin, and I. Misra. Self-Supervised Pretraining of 3D Features on any Point-Cloud. arXiv preprint:2101.02691, 2021.
- [38] Z. Zhou, Y. Zhang, and H. Foroosh. Panoptic-PolarNet: Proposal-Free LiDAR Point Cloud Panoptic Segmentation. In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2021.