Learning Moving-Object Tracking with FMCW LiDAR

Yi Gu[1], Hongzhi Cheng[1], Kafeng Wang[2], Dejing Dou[3], Chengzhong Xu[1*] and Hui Kong[1*]

***Abstract*— In this paper, we propose a learning-based moving-object tracking method utilizing our newly developed LiDAR sensor, Frequency Modulated Continuous Wave (FMCW) LiDAR. Compared with most existing commercial LiDAR sensors, our FMCW LiDAR can provide additional Doppler velocity information to each 3D point of the point clouds. Benefiting from this, we can generate instance labels as ground truth in a semi-automatic manner. Given the labels, we propose a contrastive learning framework, which pulls together the features from the same instance in embedding space and pushes apart the features from different instances, to improve the tracking quality. Extensive experiments are conducted on our recorded driving data, and the results show that our method outperforms the baseline methods by a large margin.**

I. INTRODUCTION

Multi-Object Tracking (MOT) is a significant module in modern autonomous driving pipelines. As one of the key tasks of MOT, moving-object tracking can strongly affect the local planning strategy of the ego-vehicle. With this consideration, we only focus on tracking moving objects in this work. In the literature, most state-of-the-art (SOTA) MOT algorithms are based on deep learning techniques, and require detection or segmentation results. For example, the recent MOT methods [1], [2] require to detect objects first, and then associate detections across frames. The training processes of these methods need the bounding box labels for the moving targets in each LiDAR frame. The 4D Panoptic LiDAR Segmentation (4D-PLS) [3] is based on a tracking-by-segmentation paradigm and has shown remarkable performance in the LiDAR-based MOT task. However, the labeling of the segmentation task is more expensive, even with many advanced tools [4].

Basically, these methods require a large amount of manual annotations to train a deep neural network in a supervised way, and may not generalize well in scenes beyond the training set. Typically, there are two strategies to overcome this problem. One strategy is to investigate how to utilize data efficiently or how to train a model with less supervision, such as self-supervised learning [5], [6], weakly supervised learning [7], few-shot learning [8], etc.

The other potential strategy to solve this problem is from sensor innovation or fusion perspective, since it can usually provide extra information acting as a kind of self-supervision during model training. Our work belongs to this route, where we utilize our newly developed FMCW LiDAR to track moving objects in a deep learning paradigm. Our target is to accomplish this task with as little manual-labeling effort as possible.

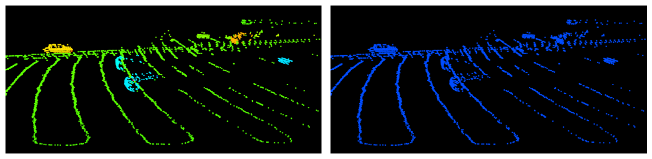
(a) Point clouds with velocity (b) Original 3D point clouds.

Fig. 1: Visual comparison of the FMCW LiDAR measurements and the range-only measurements. We use the HSI color model to visualize the velocity. Hue values in (a) are arranged by the Doppler velocity. A lower hue indicates faster speed towards the sensor.

The major difference of FMCW LiDAR from most existing LiDAR sensors lies in its supplement of speed information in addition to the 3D coordinates of the point cloud. Its basic principle is based on the Doppler effect that can provide the measurement of relative velocity between each sensed 3D world point to the FMCW LiDAR sensor along the radial direction. Figure 1 shows a certain frame of FMCW LiDAR point cloud with and without velocity information, respectively.

In contrast to the above-mentioned methods, the advantages of using the FMCW LiDAR are reflected in two aspects. The first one is to reduce the cost of data labeling, and the other is to provide the motion cue for data association.

In our work, we are devoted to fully utilizing these two advantages of the FMCW LiDAR. The first stage of our method is data labeling, a semi-automatic annotation process. Specifically, we initially separate static points from dynamic ones with velocity information. Within each LiDAR frame, a clustering algorithm [9] is applied to the moving points to get each individual moving cluster. Then data association is carried out on the moving clusters with estimated ego-motion and position (velocity) compensation.

As shown in Figure 2a, with the estimated ego-motion information (by a LiDAR SLAM algorithm for our case), the segmented moving clusters across multiple frames can be aligned to a reference frame (set as the current frame). As shown in Figure 2b, based on position compensation with velocity information on each cluster, we can further align each cluster so that the clusters corresponding to the same object are aggregated, where the initial data association can be achieved based on the distance between the centers of the aggregated clusters. It is observed that data association

The Doppler LiDAR data were provided by Guangshao Technology.

*Corresponding author.

[1]Yi Gu, Hongzhi Cheng, Hui Kong and Chengzhong Xu are with Faculty of Science and Technology, University of Macau, Taipa, Macau, China. {yc17463; hongzhicheng; huikong; czxu}@um.edu.mo

[2]Kafeng Wang is with Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences and University of Chinese Academy of Sciences. kf.wang@siat.ac.cn

[3]Dejing Dou is with Baidu Research.

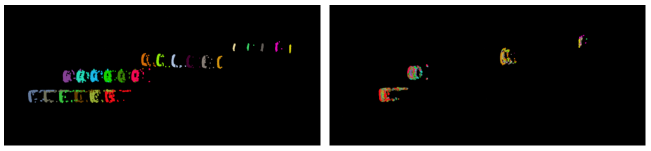

Fig. 2: The illustration of the position compensation. (a) is the result of clustering in each frame and aligned in the current frame. (b) is the result of the position compensation according to the velocity of each cluster.

can be established for most cases in this way. But the density parameter of clustering and the distance threshold need to be tuned scene-by-scene. We annotate our dataset by manually tuning these hyper-parameters. After that, each point is assigned an instance ID without semantic class label. Note that the cost of this semi-automatic annotation process is far less than the per-point annotations and bounding box annotations.

With the above ground-truth labels, we can train an end-to-end model to track the moving objects. Generally, it is not robust to track objects with only instance-level supervision. We propose a contrastive learning framework to make the features belonging to the same instance close and the features from different instances apart. With these high-quality features, we can track the instances across time simply by calculating the association probabilities of their points.

In summary, our main contributions are of two folds:

- We present the first deep learning based method on moving-object tracking with the FMCW LiDAR. We propose a contrastive learning framework to make full use of the instance-only supervision.
- Our semi-automatic annotation method with FMCW LiDAR can significantly reduce the labeling costs, which can motivate more related works based on newly developed sensors.

II. RELATED WORK

A. FMCW LiDAR

FMCW LiDAR is a newly developed type of ranging sensors and was first researched in wind measurements [10]. Compared with other popular LiDAR schemes, the scheme of FMCW LiDAR makes it able to measure the relative velocity (Doppler velocity) of each scanned point with respect to the sensor along the radial direction by the Doppler effect [11]. Since this kind of LiDAR sensors is seldom available, there are few related works specifically on perception. Recently, DICP [11] presented a novel method for point cloud registration by using Aevas Aeries I FMCW LiDAR. Guo et al. [12] proposed a clustering and velocity estimation method based on Doppler velocity with CARLA [13] simulation. Jie Shan et al. [14], [15] proposed detection and tracking methods based on hand-crafted features with Kalman filter to estimate the state of moving objects by using Blackmore FMCW LiDAR. Their methods are not based on deep learning, simple and efficient, but in low accuracy.

B. 3D Multi-Object Tracking

The dominant methods in this field exploit the tracking-by-detection paradigm, which detects objects first, then associates detections across time. The association step is usually formulated as a bipartite matching problem, where the key to matching relies on the appearance and motion models. Most of the LiDAR-based tracking methods hinge on the motion for association due to the lack of the texture cue. For example, CenterPoint [1] achieved this by estimating the velocity of each object. Other methods [16], [2] exploited appearance information from camera to enhance the association quality. SimTrack [17] avoided heuristic matching step by joint detection and tracking in an end-to-end manner. It detects objects in bird-eye-view and predicts the offset of each object directly. Its association step can be finished with a simple lookup operation. However, all these methods require bounding box annotations for detection.

MOTS [18] proposed the multi-object tracking and segmentation task for the first time and MOTP [19] extended it to panoptic segmentation and 3D domain. 4D-PLS [3] explored tracking-by-segmentation paradigm, which can associate objects at point level. The core of 4D-PLS is the density-based clustering, which benefits from their semantic features. Our method can be sorted into this category. But in our dataset, there are only instance annotations without semantic labeling, which is more challenging than previous works.

C. Contrastive Learning

Contrastive learning based methods have dominated self-pretraining in vision in recent years [5], [20]. The main idea of contrastive learning is the following: pull together an anchor and a "positive" sample in embedding space, and push apart the anchor from many "negative" samples [21]. In the self-supervised learning domain, there are no labels available. The positive samples are usually drawn from data augmentations, and the negative pairs are formed by the anchor and remaining samples in the mini-batch. When labels are available, a similar idea can also be applied in the supervised learning scheme with different loss functions, such as Triplet Loss [22], [23], N-pairs Loss [24] and Supervised Contrastive Loss [21]. Recently, Marcuzzi et al. [25] proposed to use contrastive learning to make instance association, which is similar to our work. The difference is that we only utilize the instance information, while their method also requires semantic information.

III. METHOD

The goal of our approach is to track the moving objects across different times. To achieve this, we first annotate our data by Heuristic Tracking (See Sec. III-A), which is a semi-automatic process and can provide us high-quality labeled data as ground truth. Thereafter, we train a deep neural network to achieve better performance (See Sec. III-B). Due to lack of the supervision from semantic class labels, we utilize the contrastive learning method to improve the

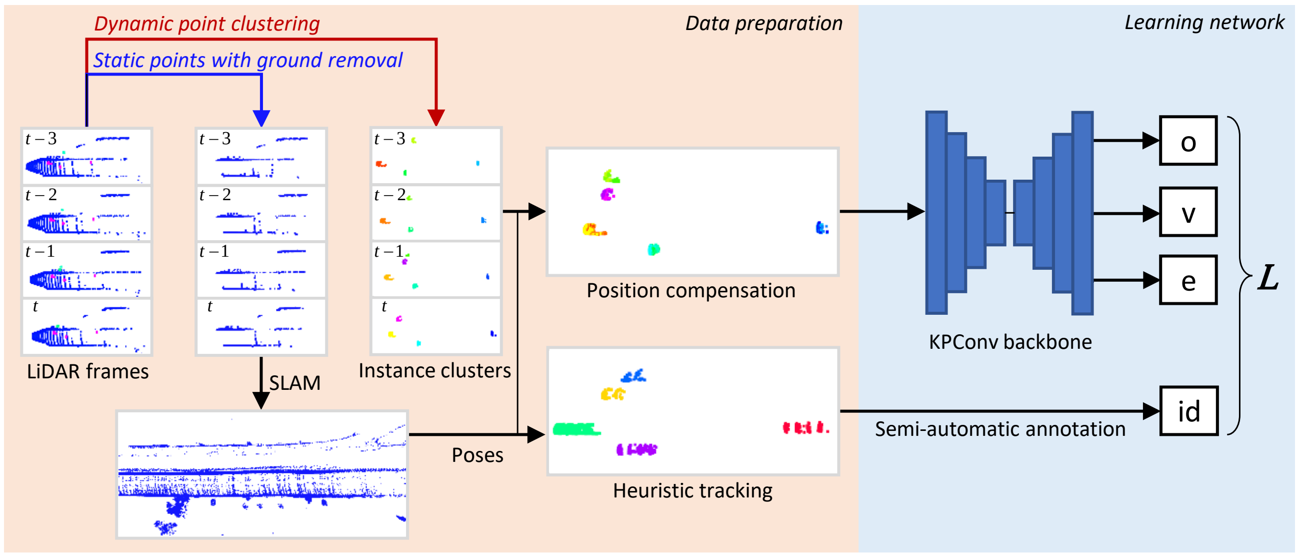

Fig. 3: An overview of our moving-object tracking pipeline (window size $\tau = 4$ for illustration).

effectiveness of the association (See Sec. III-C). Figure 3 is the overview of our pipeline.

A. Heuristic Tracking

We first filter out the abnormal points such as geometry outliers and those with unrealistic velocity values. Then we separate dynamic and static points apart according to the ego velocity V_{car}. As this value is not recorded in our dataset, we estimate it by the mean velocity of the ground points in front view, noted as V_g, which is theoretically equal to $-V_{car}$. The ground points are extracted via the RANSAC [26] algorithm. We treat the points whose velocities are within $V_g \pm V_m$ as static points, where V_m is a constant value and is set to 0.2 m/s in our experiments. Then the remaining points are dynamic points. Henceforth, we only focus on these moving points.

Since there are no bounding boxes nor instance segmentation masks, we need to define the concept of the object first. We apply DBSCAN [9] to cluster the moving points so that each cluster can be treated as an object. Though most of our cases can be successfully clustered, there are still some objects that failed to be separated out, which need to be further tuned by adjusting the density parameter of DBSCAN.

To track across the frames, we compensate the position of each cluster with the mean velocity of its own points. More specifically, for a window of τ frames at times $\{max(0, t + 1 - \tau), ..., t\}$, we first align them to the current frame t according to the ego-motion estimation results provided by a SLAM approach [27]. As we operate in consecutive frames and the window time duration in our experiments is less than 1 s, we can treat the motion of each object as uniform motion during this period. In this way, we can compensate the position of each cluster by computing the relative displacement and translating the corresponding points to simplify the association, as shown in Figure 2. Ideally, the points from the same object instance can be aggregated together so that we can do association by nearest neighbor query, accompanied by a distance threshold, d_N. That is to say, if two clusters at different times are nearest neighbors after shifting and the distance between them is within the threshold d_N, we consider them as the same instance. To make this operation more robust, we represent the position of each cluster by its centroid in the bird-eye-view to eliminate the height error.

B. 4D-PLS Review

We annotate our data based on Heuristic Tracking (See IV-B.2 for details). With the obtained ground-truth labels, we are able to train a neural network to further improve the tracking quality. We build up our model based on the 4D-PLS [3] architecture. We will review the 4D-PLS first in this subsection and introduce our modifications in the next one.

The input of the 4D-PLS is a 4D volume, which contains consecutive frames of point clouds in a time window, aligned to the current frame. The main component of the model is the density-based clustering by the following equation,

$$\hat{p}_{ij} = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma_i|^{\frac{1}{2}}} exp(-\frac{1}{2}(e_i - e_j)^T \Sigma_i^{-1}(e_i - e_j)), \quad (1)$$

where $\hat{p}_{ij}$ is the probability of the two points p_i and p_j belonging to the same instance. $e_i \in \mathbb{R}^D$ is the embedding of the p_i. Note that in this formulation, p_i should be the instance center approximation. So Σ_i is the variance matrix of p_i.

Therefore, the outputs of the 4D-PLS [3] are semantic class, objectness, variance and point embedding. Here the objectness is a scalar used to assess how close each point is to its instance center.

C. Contrastive Moving-Object Tracking

There are two drawbacks if we directly use the original 4D-PLS [3]. The first one lies in the window size limitation. Longer temporal windows can aggregate more information, while the Gaussian assumption is only valid for shorter temporal windows in 4D-PLS. Besides, since we discard the static points and only focus on the moving points, a small window size is not efficient. Thanks to the superiority of velocity information, this problem can be solved simply by the position compensation described before. After that, the Gaussian assumption can be valid for longer temporal windows. We validate this conclusion in our ablation studies.

Another problem is that our dataset has no bounding box labels nor semantic labels needed in 4D-PLS, and only the above obtained instance IDs can be used for supervised learning. It is hard to associate the points at different times without a strong semantic cue. We tried to apply 4D-PLS with only instance level supervision, where the objectness loss and instance loss are defined as:

$$L_{obj} = \sum_{i=1}^{N} (\hat{o}_i - o_i)^2, \quad \hat{o}_i, o_i \in [0,1] \tag{2}$$

$$L_{ins} = \sum_{j=1}^{K} \sum_{i=1}^{N} (\hat{p}_{ij} - p_{ij})^2, \quad p_{ij} = \begin{cases} 1, & \text{if } p_i \in I_j; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

In the above losses, notations are defined as follows: N is the number of points; K is the number of instances in the current time window; I_j represents the point set of the j-th instance; $\hat{o}_i$ is the approximation to the instance objectness o_i after position compensation. Note that since the clusters from the same instance have been aggregated, we obtain the ground-truth objectness according to the instance center, while in 4D-PLS, it is calculated for each cluster.

This method does not work well in our experiments (See Table II and Figure 4 for better comparison). Due to the lack of ground-truth class labels, we cannot rely on semantic information to obtain the class related features, which are considerably helpful for the association step.

Besides the motion cue, the appearance of the instance is also an important cue that can be explored further [17], [2]. It means that the clusters from the same instance should be similar, while from different instances should be distinct. Inspired by this hint, we propose to train a class-agnostic model in a contrastive learning paradigm [5], [21]. More specifically, we make use of the supervised contrastive learning [21] loss to further utilize the instance ID information. We represent each instance at each timestamp, i.e., each cluster, by its global feature z_i. Following Pointnet [28], we obtain this global feature z_i by applying a symmetric function on all embeddings of its points:

$$z_i = f(x_{i1}, ..., x_{in}) \approx g(e_{i1}, ..., e_{in}), \tag{4}$$

where x_{ij} is the j-th point of the cluster i and e_{ij} is its corresponding feature embedding. We define that the positive samples are clusters from the same instance and the negative samples are those from different instances.

After that, we can use the following expression to compute the loss:

$$L_{SC} = \sum_{i \in N} \frac{-1}{|P(i)|} \sum_{p \in P(i)} log \frac{exp(z_i \cdot z_p/\tau)}{\sum\limits_{a \in A(i)} exp(z_i \cdot z_a/\tau)}, \tag{5}$$

where N is the number of clusters. $A(i)$ is the index set of all the clusters within the window except i. $P(i)$ is the index set of the clusters from the same instance as i.

As for inference, we follow the 4D-PLS to associate points in two stages. For the first stage, we select the point p_i which has the highest objectness score as the instance center proxy. Then, we do the density-based clustering for all candidate points. If $p_{ij} > p_{threshold}$, we assign it to the instance of p_i and remove it from candidate points. These steps are repeated until all points are clustered. For the second stage, we perform cross-volume association greedily based on the overlap score. When the overlap is below a threshold, we assign a new ID.

IV. Experiments

A. Dataset

We utilize the latest Guangshao 2.0 FMCW LiDAR to collect our data. The horizontal field-of-view of our LiDAR is 37.5°, and the vertical field-of-view is 16.7°. The maximum perceived range is 300 m. The precision of the velocity measurement is 5 cm/s. The sampling rate is 10 Hz. The data were collected in seven different scenes, 5 from highway and 2 from urban scenes. Each scene contains 300-500 frames after sampling and all data contain 3200 frames in total. We use 2400 scannings as the training set and the left as the test set.

B. Implementation Details

1) Ego-motion Estimation: We first filter out the outlier points as mentioned in section III-A. In addition, we also remove the ground points estimated by RANSAC [26], since the points with special pattern scanned on the ground may lead to bad registration results. We use the modified LOAM [27] to get the ego-pose information. Specifically, we only use the static points to do the registration. We initialize the translation between the two frames by ego-velocity compensation. Other more robust registration methods, such as DICP [11], can be explored in future work. Figure 5a shows the local map built by our FMCW LiDAR data.

2) Semi-Automatic Annotation: We use DBSCAN [9] to cluster the moving points. The default density parameter is set as 1.2, and the minimum number of points in each cluster is 20. If there are less than 20 points in a cluster, we filter out these points. We double-check the results by both nearest neighbor query and cluster again after position compensation. After that, we manually check the correctness. For those failing and wrongly labeled data, we re-label them by hand.

3) Network Detail and Loss functions: Since the position of each cluster has been compensated, our method is not sensitive to the window size τ. We validate this by choosing different window sizes (4, 6, 8, and 10) in our experiments.

We use KPConv [29] as our backbone and the complete loss function is set as:

$$L = L_{SC} + \lambda_{ins} L_{ins} + \lambda_{var} L_{var} + L_{obj} + L_{reg}, \tag{6}$$

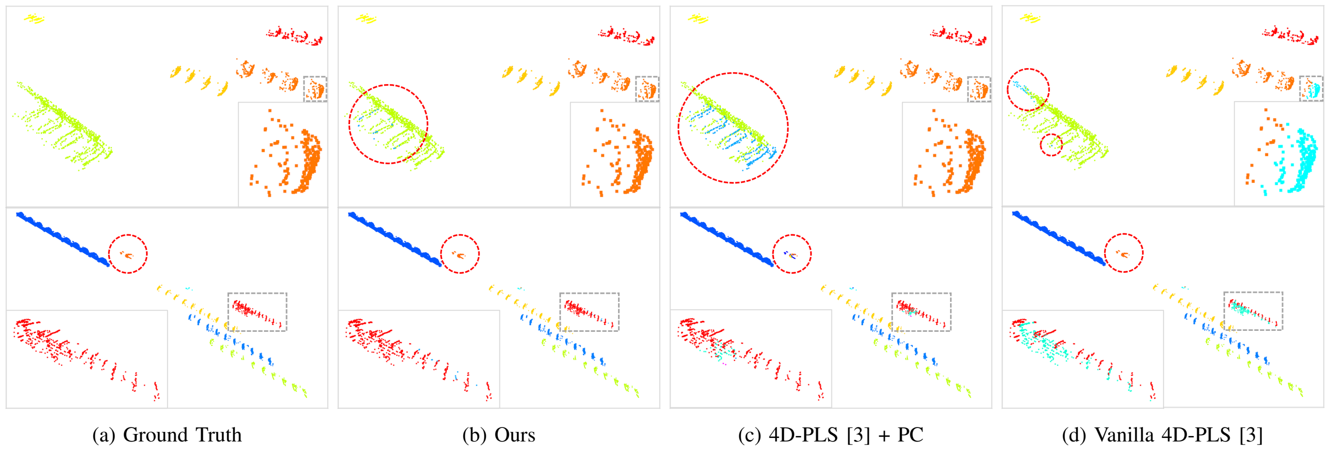

Fig. 4: Qualitative comparison of our method with baseline methods. We show the tracking results for 4 frames (top row) and 10 frames (bottom row). PC means Position Compensation. The areas with significant differences are enlarged for better comparison.

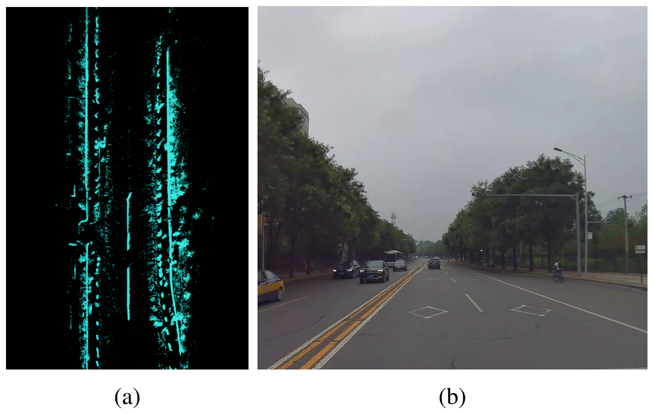

Fig. 5: (a) is the local map by SLAM [27] with only static non-ground points reserved. (b) is the corresponding image in front view. Note that the dynamic cars are filtered out in static map.

where L_{reg} is the regression loss from KPConv [29], and L_{var} is the variance smoothness loss similar to 4D-PLS [3]. We set the λ_{ins} and λ_{var} as 0.01 in our experiments. And as the KPConv has provided the per-point embeddings, we can extract instance features by directly using a max-pooling operation, which is better than the other symmetric functions.

We find that $p_{threshold} = 0.4$ can get the best performance. The overlap threshold is set as 0.8, which has a trivial effect on the final results. Note that these hyper-parameters are not changed after configuration, while the hyper-parameters in Heuristic Tracking need to be tuned scene-by-scene.

TABLE I: Comparison of our method with baseline methods. PC means Position Compensation.

Method	AS	MOTSA	MOTSP	SMOTSA
Heuristic Tracking	54.18	82.55	84.15	77.24
Vanilla 4D-PLS [3]	59.55	80.68	84.87	76.03
4D-PLS [3] + PC	84.07	88.42	87.02	85.97
Ours	**87.89**	**89.44**	**89.07**	**89.04**

C. Evaluation Metrics

Since we have no class labels, we choose the following metrics [18], [3] which are only related to instance IDs, to make a fair comparison with the baseline methods: multi-object tracking and segmentation accuracy (MOTSA), multi-object tracking and segmentation precision (MOTSP), soft multi-object tracking and segmentation accuracy (SMOTSA) and association score (AS). In all of these metrics, larger values indicate better performance.

We follow the MOTS [18] to define the set of True Positives (TP), False Positives (FP), False Negatives (FN) and ID switch (IDS) to compute the MOTSA, MOTSP and SMOTSA. For AS, we follow the 4D-PLS [3] to define the TP, FP, and FN, which can be seen as a soft version of that in MOTS. In consideration that Association Score can evaluate the association in space and time in a more unified manner, we treat AS as the main evaluation indicator.

D. Results

We compare our method with two baselines: heuristic tracking and 4D-PLS [3]. We reimplement the 4D-PLS on our dataset, without semantic output. Since 4D-PLS is sensitive to the window size, we also add position compensation to 4D-PLS to compare with our method. As for heuristic tracking, we follow our semi-automatic annotation process without human interactions. We test several sets of hyper-parameters and choose the results with the best performance. Table I shows the results of our method and baseline methods. Obviously, our method achieves the best results and generalizes well in most cases. Figure 4 is the qualitative comparison of our method with baselines. Our method is more consistent in association step. It is worth mentioning that these results are not cherry-picked.

E. Ablation studies

The velocity of points within the same instance should be close, which is also a strong cue for the association. Thus, we conduct experiments on whether to input the velocity channel. The results in Table II show that there is a slight improvement with velocity as input.

Since our method has compensated for the position of each cluster, our model is not sensitive to the window size. But the window size cannot be too large, since the memory is limited and the uniform motion assumption cannot hold for large window sizes. We conduct ablation study on the window size, with $\tau = 4, 6, 8, 10$. Table III shows that there is no obvious difference in the performance.

TABLE II: Ablation study on input format

Method	AS	MOTSA	MOTSP	SMOTSA
xyz	87.03	**89.48**	88.94	88.95
xyz+v	**87.89**	89.44	**89.07**	**89.04**

TABLE III: Ablation study on window size

Method	AS	MOTSA	MOTSP	SMOTSA
4 scans	**87.89**	**89.44**	89.07	**89.04**
6 scans	86.25	89.14	89.02	88.94
8 scans	86.47	89.16	88.94	88.62
10 scans	86.54	89.31	**89.19**	89.02

V. CONCLUSIONS

In this paper, we present a method of moving-object tracking based on the latest FMCW LiDAR sensor. To the best of our knowledge, it is the first work ever conducted on moving-object tracking with the FMCW LiDAR in a learning scheme. We propose a contrastive learning framework to make full use of the instance-only supervision, which can significantly improve the tracking quality. Our whole pipeline requires far less annotation effort than most existing works, e.g., MOT and MOTS, which require bounding box labels or per-point semantic labels. The results show that our method can surpass the baseline methods by a large margin. We hope our work can motivate more revolutionary works based on the FMCW LiDAR.

REFERENCES

[1] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 11 784–11 793.

[2] Y. Zeng, C. Ma, M. Zhu, Z. Fan, and X. Yang, "Cross-modal 3d object detection and tracking for auto-driving," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3850–3857.

[3] M. Aygun, A. Osep, M. Weber, M. Maximov, C. Stachniss, J. Behley, and L. Leal-Taixé, "4d panoptic lidar segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5527–5537.

[4] D. Bloembergen and C. Eijgenstein, "Automatic labelling of urban point clouds using data fusion," *arXiv preprint arXiv:2108.13757*, 2021.

[5] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9729–9738.

[6] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," *arXiv preprint arXiv:2111.06377*, 2021.

[7] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National science review*, vol. 5, no. 1, pp. 44–53, 2018.

[8] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.

[9] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[10] M. L. Chanin, A. Garnier, A. Hauchecorne, and J. Porteneuve, "A doppler lidar for measuring winds in the middle atmosphere," *Geophysical Research Letters*, vol. 16, no. 11, pp. 1273–1276, 1989.

[11] B. Hexsel, H. Vhavle, and Y. Chen, "Dicp: Doppler iterative closest point algorithm," *arXiv preprint arXiv:2201.11944*, 2022.

[12] M. Guo, K. Zhong, and X. Wang, "Doppler velocity-based algorithm for clustering and velocity estimation of moving objects," *arXiv preprint arXiv:2112.12984*, 2021.

[13] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[14] Y. Ma, J. Anderson, S. Crouch, and J. Shan, "Moving object detection and tracking with doppler lidar," *Remote Sensing*, vol. 11, no. 10, p. 1154, 2019.

[15] X. Peng and J. Shan, "Detection and tracking of pedestrians using doppler lidar," *Remote Sensing*, vol. 13, no. 15, p. 2952, 2021.

[16] H.-k. Chiu, J. Li, R. Ambruş, and J. Bohg, "Probabilistic 3d multi-modal, multi-object tracking for autonomous driving," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 14 227–14 233.

[17] C. Luo, X. Yang, and A. Yuille, "Exploring simple 3d multi-object tracking for autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 488–10 497.

[18] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "Mots: Multi-object tracking and segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7942–7951.

[19] J. V. Hurtado, R. Mohan, W. Burgard, and A. Valada, "Mopt: Multi-object panoptic tracking," *arXiv preprint arXiv:2004.08189*, 2020.

[20] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[21] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 18 661–18 673, 2020.

[22] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification." *Journal of machine learning research*, vol. 10, no. 2, 2009.

[23] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[24] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," *Advances in neural information processing systems*, vol. 29, 2016.

[25] R. Marcuzzi, L. Nunes, L. Wiesmann, I. Vizzo, J. Behley, and C. Stachniss, "Contrastive instance association for 4d panoptic segmentation using sequences of 3d lidar scans," *IEEE Robotics and Automation Letters*, 2022.

[26] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[27] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[28] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[29] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6411–6420.