Simpler Is Better: Revisiting Doppler Velocity for Enhanced Moving Object Tracking with FMCW LiDAR

Yubin Zeng†, Tao Wu†*, Shouzheng Qi, Junxiang Li, Xingyu Duan and Youjin Yu

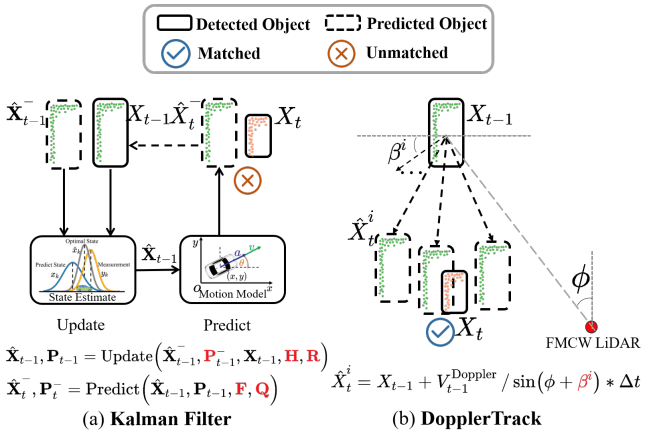
***Abstract*— Real-time and accurate perception of dynamic objects is crucial for autonomous driving. To better capture the motion information of objects, some methods now employ 4D Doppler point clouds collected by frequency-modulated continuous-wave (FMCW) LiDAR to enhance the detection and tracking of moving objects. Compared to standard time-of-flight (ToF) LiDAR, FMCW LiDAR can provide the relative radial velocity of each point through the Doppler effect, offering a more detailed understanding of an object's motion state. However, despite the proven efficacy of these methods, ablation studies reveal that the direct contribution of Doppler velocity to tracking is limited, with performance gains often resulting from improved object recognition and labeling accuracy.**

Revisiting the role of Doppler velocity, this study proposes DopplerTrack, a simple yet effective learning-free tracking method tailored for FMCW LiDAR. DopplerTrack harnesses Doppler velocity for efficient point cloud preprocessing and object detection with O(N) complexity. Furthermore, by exploring the potential motion directions of objects, it reconstructs the full velocity vector, enabling more direct and precise motion prediction. Extensive experiments on four datasets demonstrate that DopplerTrack outperforms existing learning-free and learning-based methods, achieving state-of-the-art tracking performance with strong generalization across diverse scenarios. Moreover, DopplerTrack runs efficiently at 120 Hz on a mobile CPU, making it highly practical for real-world deployment. Code and datasets will be released at https://github.com/12w2/DopplerTrack.

I. INTRODUCTION

The capability to accurately perceive moving objects in real time is fundamental for autonomous vehicles to perform tasks such as obstacle avoidance [1], [2], scene understanding [3], and path planning [4], [5].

To obtain accurate spatial information on objects, state-of-the-art 3D Multi-Object Tracking (MOT) methods primarily rely on the three-dimensional point clouds provided by TOF LiDAR as input [6], [7], [8]. These methods use the Kalman filter (KF) [6] or velocity predictions [7] by detectors from multi-frame data to predict and update the states of objects. To more directly capture the motion information of objects, some methods have begun to use FMCW LiDAR [9] [10], which can provide 4D Doppler point clouds, for research on the detection and tracking of moving objects. Compared to standard TOF LiDAR, FMCW LiDAR provides additional Doppler velocity information to the point clouds, which should be beneficial for improving the motion prediction and enhancing the tracking performance of objects. However, surprisingly, although these efforts have made significant progress in objects recognition [10] and semi-automatic labeling [9], ablation experiments demonstrate that the Doppler velocity information has minimal impact on tracking performance. This finding prompts us to reflect: What is the actual value of Doppler velocity information in the field of object tracking?



(a) **Kalman Filter** (b) **DopplerTrack**

Fig. 1. **Comparison of DopplerTrack and Kalman filter in object tracking, where hyperparameters are highlighted in red.** Kalman filter (a) relies on multiple hyperparameters for modeling during prediction and update processes, including the motion model F, process noise covariance Q, measurement noise covariance R, measurement model H, and initial covariance P_0. When the object's observation noise or motion model of deviates from these predefined hyperparameters, the accuracy of state estimation may degrade, potentially leading to tracking failures. In contrast, DopplerTrack (b) postulates potential motion directions β of objects and analytically reconstructs complete velocity vectors by resolving Doppler velocities along these motion directions. It then employs a constant velocity model to generate the predicted candidates. By directly deriving state information from measurements, DopplerTrack simplifies the tracking process and enhances robustness.

Bearing such objectives, we revisit the utilization of Doppler velocity in tracking and analyze its limitations in existing methods. These methods [9], [10] typically involve directly incorporating Doppler velocity information into algorithms previously designed for 3D point cloud-based MOT, including Kalman filter and 4D Panoptic LiDAR Segmentation (4D-PLS) [11]. For the Kalman filter, it refines the state of an object by constructing a motion model using the detection results of the object. Although the new Doppler velocity measurement is sufficiently accurate, they are inevitably affected by the noise in other measurements, the selection of motion model, and parameter tuning within the complex Kalman filter system. In 4D-PLS, it constructs a 4D volume by jointing multi-frame point cloud segmentation results to the current frame using odometry and directly

†Yubin Zeng and Tao Wu contributed equally to this work
All authors are with College of Intelligence Science and Technology, National University of Defense Technology, China. T. Wu is the corresponding author. E-mail: wutao@nudt.edu.cn

outputs tracking results through a network. While deep neural networks are powerful enough to implicitly learn the information of Doppler velocity, the concatenation by odometry disrupts the Doppler velocity information from historical frames. Specifically, the Doppler velocity is merely the component of the point velocity along the laser beam, not the true velocity vector. When transforming the pose of historical frame objects through odometry, the inability to know the object's true velocity direction prevents the transformation of the Doppler velocity into the current coordinate system, thereby losing the capability to implicitly enhance motion estimation through Doppler velocity.

To leverage Doppler velocity for enhanced moving object tracking, we propose DopplerTrack, a simple yet effective method tailored for FMCW LiDAR. Our approach is driven by a key insight: while measured Doppler velocities may be incomplete, they are sufficiently accurate. By reconstructing the full velocity vector, we enable stable tracking of moving objects using a simple constant velocity model. Building on this understanding, we generate a set of search objects based on the possible motion directions of each object and impose Doppler velocity constraints for robust association. Furthermore, we exploit Doppler velocity to optimize preprocessing and detection, achieving O(N) complexity, which enhances both the robustness and computational efficiency of the tracking pipeline.

Extensive experiments on four challenging datasets demonstrate that DopplerTrack achieves state-of-the-art tracking accuracy while running efficiently at 120 Hz on a mobile CPU. By eliminating the need for complex motion models and extensive parameter tuning, our approach significantly outperforms existing methods while maintaining strong generalization across diverse scenarios. These results reaffirm the principle that **simpler is better**, highlighting the practical value of Doppler velocity in moving object tracking.

II. RELATED WORKS

A. 3D Multi-Object Tracking

Most 3D MOT methods [12], [6], [13], [14], [15] follow the tracking-by-detection (TBD) framework, which divides the tracker into three steps: (1) point cloud reception and preprocessing, (2) 3D object detection, and (3) motion prediction, data association, and lifecycle management. Relying on powerful 3D object detectors [16], [17], [18], the TBD framework has achieved state-of-the-art performance on multiple datasets, with most methods focusing on the tracking component. SimpleTrack [13] has achieved impressive results by analyzing the failure modes of existing methods and making strategic improvements. Poly-MOT [6] achieves state-of-the-art results by designing specialized tracking modules for different object categories and refining motion estimation. As 3D object detection performance may degrade under domain shifts [19], some methods have explored MOT based on segmentation results [11], [20]. Among them, 4D-PLS [11] constructs a 4D volume by concatenating multi-frame segmentation results and using deep learning to directly output tracking results, achieving remarkably impactful outcomes. Unlike these approaches, our method removes dependence on object detectors and deep learning models by directly leveraging Doppler velocity for object segmentation and motion estimation.

B. FMCW LIDAR Tracking

FMCW LIDAR is a newly developed type of ranging sensor that measures the radial velocity of each return by exploiting the Doppler effect. Due to the maturity of perception algorithms for TOF LIDAR, most perception research conducted on FMCW LIDAR is based on the algorithms developed for TOF LIDAR. Guo et al. [21] proposed a point cloud region growing clustering and velocity estimation algorithm based on Doppler velocity, and conducted simulation experiments on CARLA [22]. Peng et al. [10] integrated Doppler velocity into the dimensions of an SVM classifier to improve the accuracy of pedestrian classification and enhance tracking performance. A groundbreaking study [9], the first to apply deep learning to FMCW LIDAR data, utilized Doppler velocity as an indicator of the object's forward-moving speed. It employed heuristic tracking to generate more robust point cloud labels and then applied the 4D-PLS [11] for multi-object tracking. While these methods have achieved commendable results in dynamic-static separation, object identification, and semi-automatic labeling, they have not fully leveraged the considerable potential of Doppler velocity in the core tracking algorithms. Unlike these methods, which treat Doppler velocity as supplementary information, our approach leverages it as the core of the tracking pipeline. Specifically, we reconstruct the full velocity vector from Doppler measurements, enabling robust motion estimation without complex priors.

III. PROPOSED METHOD

A. Overview

DopplerTrack is a universal framework that operates without data training, employing 4D Doppler point clouds as input to detect and track moving objects. As seen in Fig. 2, in our pipeline, we first apply a pre-processing module (c.f. Sec. III-B) to segment dynamic point clouds. Subsequently, in the object detection module (c.f. Sec. III-C), we detect each moving object through clustering. Finally, in the object tracking module (c.f. Sec. III-D), we accomplish motion prediction, data association and trajectory management for the objects. Specifically, for each object in the tracking result T_{t-1} of the previous frame, we generate a series of potential candidate object groups according to the possible motion directions and accomplish matching with the current frame's detection results D_t through data association.

B. Pre-processing Module

On receiving a new 4D Doppler point cloud $\mathbf{P_t}$, the framework initially requires the rapid and accurate segmentation of dynamic point clouds, which is crucial for ensuring the real-time capability of the system and the precision of subsequent detection modules. Existing approaches often rely on ground estimation [9] or region-growing algorithms

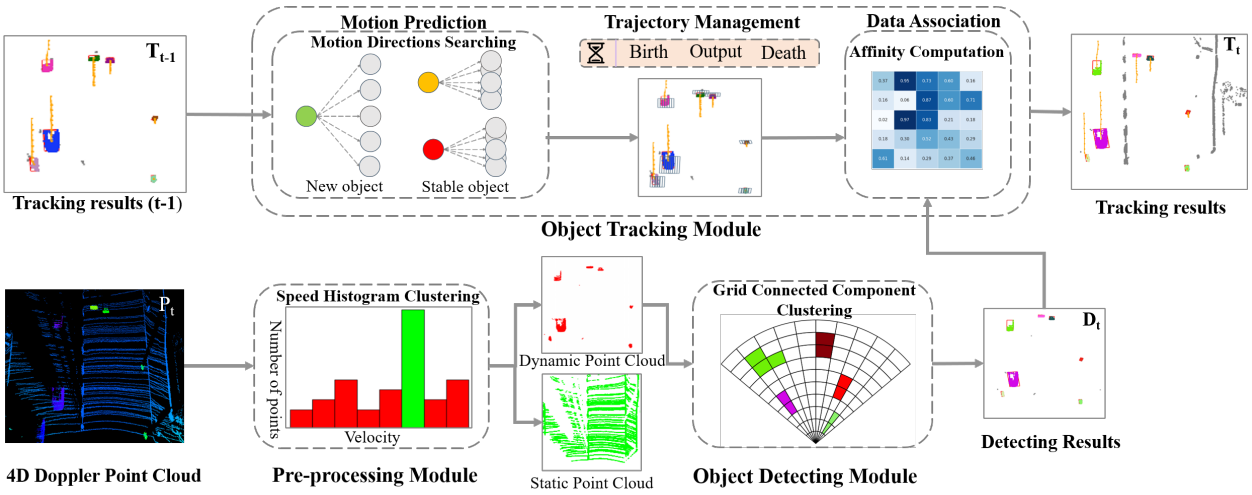


Fig. 2. The pipeline of DopplerTrack. Note that point clouds are shown in the bird's eye view for visualization.

Algorithm 1 GCC-Cluster

Initialize: Label $\leftarrow 1$
$p \leftarrow$ convert-to-spherical(point_cloud)
Initialize hash-table, grid-velocities, and grid-counts
for each point in p **do**
  $\rho_{\text{index}}, \theta_{\text{index}} \leftarrow$ compute-grid-index(point, $\Delta\rho, \Delta\theta$)
  Update hash-table, grid-velocities, and grid-counts using $(\rho_{\text{index}}, \theta_{\text{index}})$ and point.velocity
end for
Compute average velocities for each grid
$L \leftarrow$ zeros(size of hash-table)
for each grid in hash-table **do**
  if L(grid) $= 0$ **then**
    LABELCOMPONENTBFS(grid, Label, hash-table, L, grid-avg-velocities)
    Label $\leftarrow$ Label + 1
  end if
end for
procedure LABELCOMPONENTBFS(grid, Label, hash-table, L, grid-avg-velocities)
  queue.push(grid)
  while queue is not empty **do**
    Propagate labels in BFS manner based on velocity threshold and unlabeled neighbors
  end while
end procedure

[21] to separate static and dynamic points. While these methods achieve reasonable accuracy, they still rely on 3D spatial point cloud search, potentially limiting computational efficiency. To address these limitations, we developed a bucket sorting partition algorithm based on the Doppler velocity dimension. This algorithm relies solely on the single velocity dimension of the point cloud and operates with a time complexity of O(N). It ensures superior speed while maintaining the accuracy of the algorithm. Specifically, we first reverse the Doppler velocity V_{Doppler} of the point cloud to the opposite direction of the vehicle's travel direction (negative X-axis) to obtain V'_{Doppler}. Following the principles of Doppler velocity measurement, all static point clouds should have a consistent and unique V'_{Doppler}. Consequently, the V'_{Doppler} for any dynamic object, except those moving tangentially to the LiDAR's line of sight, will differ from that of static objects. The computation of V'_{Doppler} is as follows:

$$V'_{\text{Doppler}} = V_{\text{Doppler}} / \sin(\theta) / \cos(\phi), \tag{1}$$

where θ and ϕ represent the polar and azimuthal angles of the measurement point in the spherical coordinate system, respectively.

Subsequently, we perform bucket sorting on all points' V'_{Doppler}, where the bucket size is determined by the measured noise, with the largest bucket representing the static point cloud. To mitigate the noise in point cloud scanning and the influence of bucket division on the sorting algorithm, we select a portion of the point cloud situated directly in front of the vehicle from the largest bucket. We calculated their average velocity $\overline{V}$ and applied a preset threshold ΔV to filter the point cloud's V'_{Doppler}. Points within $\overline{V} \pm \Delta V$ are identified as static. The remaining points were designated as the dynamic point cloud, which was our primary focus.

C. Object Detection Module

In most TBD methods [13], [6], it is typically required to first obtain accurate object 3D bounding boxes and category attributes across consecutive frames to form continuous object trajectories. However, for tracking tasks, neither category information nor complete 3D bounding boxes are essential, and acquiring this information from sparse point clouds is challenging. Consequently, in our approach, we employ clustering algorithms to identify these point clouds as simple cluster objects and conduct subsequent tracking tasks based on their minimal bounding boxes.

Previous methods [23], [24] employ voxel-based clustering strategy, where the 3D space is discretized into predefined

volumetric grids. However, since our approach has already eliminated most static interferences in the preprocessing module, we adopt a more efficient strategy by projecting the point cloud onto a 2D grid and performing clustering via a connected components method (GCC-Cluster).

Algorithm 1 outlines our adopted GCC-Cluster, which operates in linear time O(N). To address the issue of varying point densities—where points near the sensor are significantly denser than those farther away—we first transform the point cloud from the Cartesian coordinate system to a polar grid representation for grid-based processing. Subsequently, we apply a Breadth-First Search (BFS)-based labeling procedure, where neighboring grid cells are assigned to the same cluster based on their Doppler velocity consistency. Finally, we generate a minimum bounding box for each identified cluster, which serves as the final detection output.

D. Object Tracking Module

Following the classical object tracking paradigms [12], [13], our tracking module consists of three key components: motion prediction, data association, and trajectory management. We focus on redesigning motion prediction and data association, while trajectory management follows standard count-based rules [12], [25].

Motion Prediction. Tracking moving objects in FMCW LiDAR presents a unique challenge: while the measured Doppler velocities provide instantaneous motion cues, they represent only the radial component of the object's velocity rather than the full motion vector. However, despite this incompleteness, these measurements are both highly accurate and directly observable, making them a reliable foundation for motion prediction.Motivated by this insight, we hypothesize potential motion directions and reconstruct the complete velocity vectors by resolving Doppler velocities along these directions. Specifically, for newly detected objects, we adopt a broader search range to account for all possible motions, while for stable objects, we narrow the search space around the prior motion direction. Once the velocity is estimated, we employ a constant velocity model to generate a set of predicted object positions. As shown in Fig. 1, we process the Doppler velocity based on the search direction β to obtain the object's motion velocity at time $t-1$. Utilizing this velocity, we employ a constant velocity model to generate a series of candidate object boxes $\hat{X}_t^i$. The calculation formula is as follows:

$$\hat{X}_t^i = X_{t-1} + V_{t-1}^{Doppler} / \sin(\phi + \beta^i) \times \Delta t, \tag{2}$$

where ϕ is the object's azimuth, β^i is the potential movement direction of the object being searched, X_{t-1} is the object's position in the previous frame, and X_t^i is the position of the ith object obtained through the search. $V_{t-1}^{Doppler}$ is the Doppler velocity of the box in the previous frame, obtained by calculating the average Doppler velocity of all points within the box.

Data Association. In this section, We establish a similarity metric for associating objects across consecutive frames, constructing an affinity matrix solved via the Hungarian algorithm. The metric design accounts for two key challenges: (1) Cluster-based detections, where object proposals lack well-defined 3D bounding boxes, causing significant shape variations across frames; and (2) Many-to-one matching, where an object in the current frame may correspond to multiple candidates from the previous frame.

To address these issues, we designed a composite metric L_{all}, consisting of an overlap metric L_{overlap} and a Doppler velocity difference metric L_{Doppler}. The calculation formula for the composite metric L_{all} is as follows:

$$L_{\text{all}} = L_{\text{overlap}} + L_{\text{Doppler}}. \tag{3}$$

Considering that the optimal case under object shape variations occurs when the bounding box is fully enclosed, meaning the minimum bounding box equals the union, we adopt GIoU [26] to measure overlap. The calculation formula provided as follows:

$$L_{\text{overlap}} = 1 - \frac{I_{B_i,B_j}}{U_{B_i,B_j}} + \frac{C_{B_i,B_j} - U_{B_i,B_j}}{C_{B_i,B_j}}, \tag{4}$$

where I_{B_i,B_j}, U_{B_i,B_j} and C_{B_i,B_j} are respectively the intersection, union, and minimum bounding box between box B_i and B_j.

To enhance robustness, we introduce a Doppler velocity difference metric based on motion consistency. Since an object's velocity remains relatively stable across frames, the Doppler velocity difference should be minimal. The calculation formula provided as follows:

$$L_{\text{Doppler}} = \frac{|B_i^{\text{Doppler}} - B_j^{\text{Doppler}}|}{\max(|B_i^{\text{Doppler}}|, |B_j^{\text{Doppler}}|)}, \tag{5}$$

where B_i^{Doppler} and B_j^{Doppler} represent the average Doppler velocity values of all points within the boxes B_i and B_j, respectively.

Trajectory Management. This component manages object birth, output, and death using a simple counting-based strategy. Unmatched detections D_{unmatch} are treated as potential new objects but withheld from output until consistently matched for N_{birth} frames, at which point they are confirmed as new trajectories T_{new}. For death, trajectories are retained for up to max-age frames before deletion, ensuring robustness against temporary occlusions and missed detections.

IV. EXPERIMENTS

A. Dataset

As shown in Fig. 3, we evaluated our algorithm on four diverse and challenging datasets: **Straight**, **Intersection**, **Aqcar**, and **Aeva-car**. The first three datasets were collected using our Aqronos FMCW LIDAR, which has a field of view (FOV) of 100°×20° (H×V), a maximum measurement range of 200 m, and velocity measurement accuracy of 5 cm/s. The fourth dataset, **Aeva-car**, was collected from the bridge02 sequence of the public dataset HeLiPR [27] using Aeva Aeries II FMCW LIDAR, which has a FOV of 120°×19.2° (H×V) and a maximum measurement

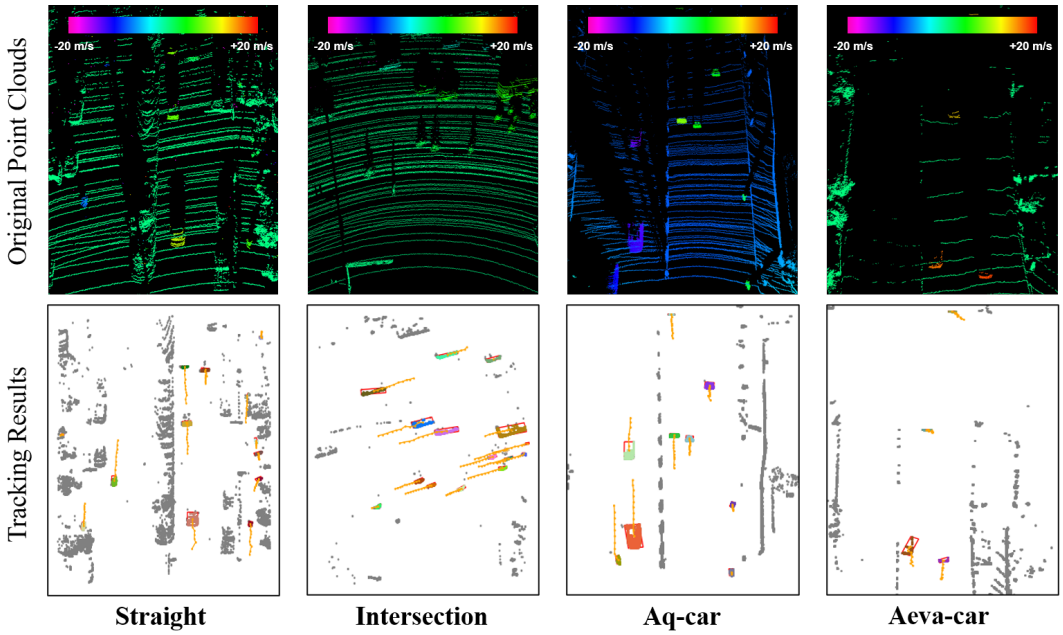


Fig. 3. Qualitative results of DopplerTrack: The original point clouds are colored according to the Doppler velocities measured by the FMCW LIDAR sensor. In the corresponding bird's-eye view, the detected moving objects and their trajectories over ten frames are delineated, employing varied colors to differentiate among distinct object clusters while ground points are omitted to emphasize the objects.

range of 200 m. We annotated all dynamic objects in the datasets, with longitudinal annotations ranging from 0-100m and lateral annotations from -20m to 20m. The annotations consist of 14,247 moving 3D bounding boxes with tracking IDs, including 7,824 cars, 4,799 bicycles, 448 trucks, and 1,176 pedestrian labels. Detailed information about the four datasets is as follows:

Straight. Collected straight road data from 7.8 meters high traffic monitoring. The dataset includes 409 frames featuring numerous pedestrians and bicycles, some of which are obscured by trees.

Intersection. Collected intersection data from 7.3 meters high traffic monitoring, it includes a significant number of laterally crossing objects, totaling 400 frames. Due to the Doppler velocity measurement being zero when the object is tangential to the laser beam, this presents a significant challenge to our algorithm.

Aq-car. Collected from a moving vehicle, it includes dense bi-directional traffic flows, totaling 350 frames.

Aeva-car. Collected from a moving vehicle, this dataset has sparser point clouds compared to the others, totaling 750 frames.

B. Evaluation Setup

Baselines. We established three baselines for comparison: two are learning-based methods, where the detectors utilize PointPillars [16] and CenterPoint [17], and the tracker uniformly adopts the detector from SimpleTrack [13]. The other is a learning-free method, where the detector employs the current state-of-the-art unsupervised clustering method ElC [23], and the tracker uses KF-speed [10]. For the learning-based methods, we reproduced the detection results on our dataset using the open-source code library OpenPCDet [28]. We split the dataset into training and testing sets in an 80% and 20% ratio, where the training set was used for training deep learning models, and the testing set was used to evaluate all experiments. The dataset includes four categories: car, bicycle, truck, and pedestrian. Notably, since only dynamic objects were annotated, the input point cloud data and labels for deep learning are limited to 4D dynamic point clouds and objects. Moreover, we adjusted the heights of point clouds from two traffic monitoring datasets, **Straight** and **Intersection**, to a vehicle-mounted perspective (1.7m). These adjustments help to improve the performance and accuracy of learning-based models.

Evaluation Metrics. To quantify our performance, we use the classical MOTA [29] and the IDF1 [30] as the primary metrics for evaluation. To adapt these evaluation metrics to our cluster-based object detections, we follow the point-based IoU computation method from RaTrack[31] , where the IoU is calculated by counting the number of intersected and union points between the ground truth and predicted objects. In all experiments, the point-based IoU threshold is set to 0.4.

C. Implementation Details

Our detection and tracking algorithm is implemented in C++ under Intel® 13900HX without the use of any GPU. Compared to previous learning-free methods [10], our method is streamlined and not overly sensitive to hyper-parameter tuning. Therefore, we did not meticulously adjust all hyper-parameters for the dataset. Specifically, slight variations exist for different LIDARs, but parameters remain consistent for datasets from the same LIDAR. Detailed

TABLE I

PERFORMANCE OF DOPPLERTRACK AND BASELINES ON FOUR DATASETS. ALL EXPERIMENTS WERE CONDUCTED ON MOVING OBJECTS.

Method		Straight		Intersection		Aq-car		Aeva-car	
Tracker	**Detector**	MOTA↑	IDF1↑	MOTA↑	IDF1↑	MOTA↑	IDF1↑	MOTA↑	IDF1↑
KF-Speed [10]	ElC [23]	21.0	59.6	49.8	70.5	63.9	81.0	47.8	29.2
SimpleTrack [13]	PointPillars [16]	48.6	66.0	59.9	78.1	64.4	80.2	83.4	91.4
SimpleTrack [13]	CenterPoint [17]	64.1	80.6	50.4	75.5	69.1	82.0	78.1	88.9
DopplerTrack	**GCC-Cluster (ours)**	**71.6**	**80.9**	**78.9**	**86.4**	**91.8**	**94.6**	**93.6**	**95.2**

TABLE II

PER-FRAME AVERAGE RUNTIME (MS) OF DIFFERENT MODULES IN DOPPLERTRACK

Module	**Pre-processing**	**Detection**	**Tracking**	**Total**
Time (ms)	3.22	1.15	3.95	8.32

hyper-parameters for each module are as follows:

Pre-processing Module. Due to the larger measurement noise of the Aqronos FMCW LIDAR compared to the Aeva Aeries II FMCW LIDAR, we set the bucket size to 0.8 for the former and 0.4 for the latter, with ΔV uniformly set to 0.8. For static-viewpoint datasets, a simple background subtraction was applied to reduce background noise.

Object Detection Module. Due to the higher point cloud density of the Aqronos FMCW LiDAR compared to the Aeva II FMCW LiDAR, we set the former's $\Delta\rho$ and $\Delta\theta$ to 0.2 and 0.4. For the latter, these parameters are set to 0.4 and 0.6.The speed threshold in BFS is uniformly set to 0.5.

Object Tracking Module. In motion prediction, we set the search number of candidates to 12, generating uniformly distributed search directions. The search range is set to 170 degrees for newborn objects and 20 degrees for stable objects. In trajectory management, we uniformly set both max-age and N_{birth} to 3.

D. Results

We evaluate DopplerTrack and the baseline methods across four datasets and compare their performance on key metrics, as summarized in Table I. DopplerTrack demonstrates a notable performance gain over all baselines, achieving MOTA improvements of 7.5%, 19.0%, 22.7%, and 10.2% on the **Straight**, **Intersection**, **Aq-car**, and **Aeva-car** datasets, respectively. Furthermore, with a more direct and reliable motion prediction module, DopplerTrack consistently outperforms all baselines on the IDF1 metric, underscoring its capability to maintain object identity over time. We also show some qualitative results of DopplerTrack in Fig. 3.

In addition to its accuracy, DopplerTrack achieves high computational efficiency, making it well-suited for real-time applications. As shown in Table II, we provide a breakdown of the average runtime of each module on a real-world driving sequence (∼340s). The entire pipeline operates within 8.32 ms per frame, ensuring low latency while maintaining strong tracking performance. These results indicate that DopplerTrack improves tracking accuracy while remaining computationally efficient, making it suitable for deployment in real-world autonomous driving scenarios.

E. Ablation Study

To validate the effectiveness of our object detection and tracking modules, we conduct an ablation study to analyze their impact. The results are summarized in Table III.

By replacing our detection module (c.f. Sec. III-C) with the state-of-the-art unsupervised detector ElC [23], we observe a degradation in MOTA across all datasets: 7.6% on **Straight**, 3.3% on **Intersection**, 0.9% on **Aq-car,** and 2.9% on **Aeva-car**. Similarly, substituting tracking module (c.f. Sec. III-D) with SimpleTrack results in a more pronounced performance drop: 4.3% on **Straight**, 12.7% on **Intersection**, 11.4% on **Aq-car**, and 12.6% on **Aeva-car**. These results suggest that the primary source of performance improvement is attributed to the tracking module, while the detection module demonstrates remarkable generalization across diverse scenarios.

Further analysis reveals that in the **Intersection** dataset, severe occlusions cause significant deformations in object bounding boxes. In the **Aq-car** and **Aeva-car** dataset, the high-speed motion of the ego vehicle introduces considerable variations in the relative motion patterns of objects. These factors induce discrepancies between the actual object dynamics and the kinematic assumptions of Kalman filtering, compromising the robustness of traditional trackers. In contrast, DopplerTrack directly exploits Doppler velocity to constrain object motion, effectively mitigating the accumulation of model prediction errors. This enables it to maintain stable tracking performance even in complex scenarios with heavy occlusions and high-speed motion. These findings further validate the effectiveness of our direct motion prediction strategy leveraging Doppler velocity cues.

F. Sensitivity Analysis

Impact of IoU Threshold. We analyze the impact of different point-based IoU thresholds on the evaluation results. As shown in Fig. 4, DopplerTrack achieves the best performance across all thresholds on the **Aq-car** dataset with over 20% improvement, which benefits from the cluster-based detector.

TABLE III
ABLATION STUDY RESULTS FOR DOPPLERTRACK.

Method	Straight		Intersection		Aq-car		Aeva-car	
	MOTA↑	IDF1↑	MOTA↑	IDF1↑	MOTA↑	IDF1↑	MOTA↑	IDF1↑
Replace detector with EIC [23]	64.0	73.3	75.6	**86.6**	90.9	91.9	90.7	90.0
Replace tracker with SimpleTrack [13]	67.3	77.3	66.2	82.7	80.4	87.7	81.0	88.2
DopplerTrack	**71.6**	**80.9**	**78.9**	86.4	**91.8**	**94.6**	**93.6**	**95.2**

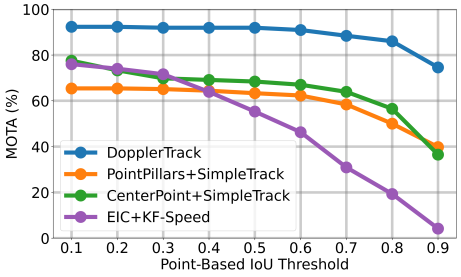


Fig. 4. Comparison of DopplerTrack and baselines across different point-based IoU thresholds on the **Aq-car** dataset.

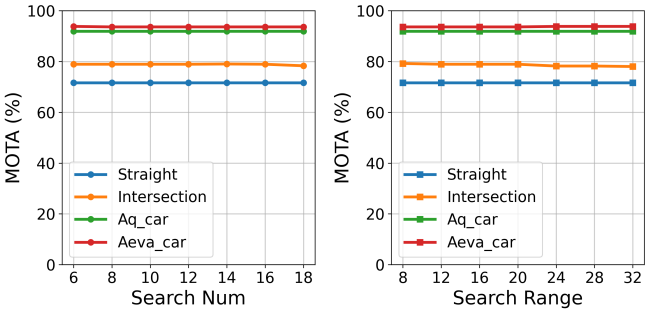


Fig. 5. The impact of search num and search range on DopplerTrack's performance across four datasets.

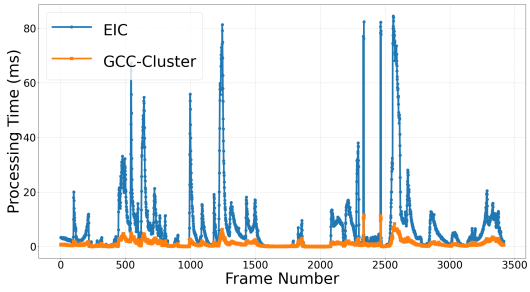


Fig. 6. Runtime performance comparison of GCC-Cluster and EIC [23] on a real-world driving sequence.

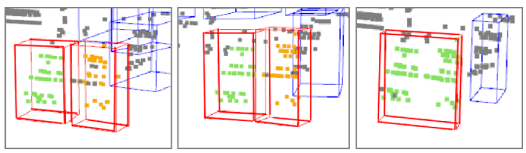
Impact of Search Parameters. We also analyze the impact of different search parameters in the motion prediction component of the object tracking module, including the search number of candidates and the search range of stable objects. As shown in Fig. 5, the results across four datasets indicate that search parameters have minimal influence on performance, demonstrating the robustness of our motion prediction component. In contrast to Kalman filtering, which requires tuning complex hyperparameters [6], DopplerTrack operates effectively with little to no parameter tuning, further reinforcing the core principle of DopplerTrack: **Simpler Is Better**.

Runtime Analysis of the Detector. We evaluate the runtime performance of GCC-Cluster and EIC [23] on a real-world driving sequence. As shown in Fig. 6, GCC-Cluster exhibits significantly more stable runtime performance across diverse scenarios compared to EIC [23]. This highlights the strong adaptability of our approach in dynamic, real-world environment, ensuring that moving objects are detected with minimal latency at all times.

G. Limitations

Although we achieved a lead over the baseline methods across four datasets, the improvement on the **Straight** dataset was not as expected. We analyzed the detailed data in Table IV and found that our method had more than doubled the number of missed detections (FN) compared to the second-best baseline. As shown in Fig. 7, we conducted a visual analysis of the failure cases. In comparison with the baseline method, our detection algorithm failed to separate two adjacent pedestrians, erroneously perceiving them as a single object. Despite incorporating a Doppler velocity difference constraint into our clustering algorithm, it remains limited when objects are in close proximity and move at consistent speeds.

In real-world applications, it is uncommon for objects to consistently exhibit strict parallelism and uniform velocity. Hence, despite our algorithm's constrained performance in such conditions, this shortcoming does not substantially affect its efficacy in broader applications.



(a) Ground Truth (b) CenterPoint (c) GCC-Cluster

Fig. 7. Visualize our method and the baseline method on failure cases.

TABLE IV

DETAILED PERFORMANCE ON THE **STRAIGHT** DATASET

Method	Detector	MOTA↑	FP↓	FN↓	IDs↓
SimpleTrack [13]	CenterPoint [17]	64.1	234	**217**	**5**
DopplerTrack	**GCC-Cluster**	**71.6**	**38**	316	7

V. CONCLUSIONS

This work revisits Doppler velocity for moving object tracking and shows that simpler is better. We propose DopplerTrack, a learning-free method that reconstructs full velocity vectors from Doppler measurements and achieves state-of-the-art performance without training. It remains robust to parameter variations and runs in real time at 120 Hz on a mobile CPU, making it practical for deployment. These results highlight the power of simple yet effective solutions.

REFERENCES

[1] Yingbing Chen, Ren Xin, Jie Cheng, Qingwen Zhang, Xiaodong Mei, Ming Liu, and Lujia Wang. Efficient speed planning for autonomous driving in dynamic environment with interaction point model. *IEEE Robotics and Automation Letters*, 7(4):11839–11846, 2022.

[2] Anjian Li, Liting Sun, Wei Zhan, Masayoshi Tomizuka, and Mo Chen. Prediction-based reachability for collision avoidance in autonomous driving. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7908–7914. IEEE, 2021.

[3] Zhengxia Zou, Rusheng Zhang, Shengyin Shen, Gaurav Pandey, Punarjay Chakravarty, Armin Parchami, and Henry X Liu. Real-time full-stack traffic scene perception for autonomous driving with roadside cameras. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 890–896. IEEE, 2022.

[4] Chuan-Che Lee and Kai-Tai Song. Path re-planning design of a cobot in a dynamic environment based on current obstacle configuration. *IEEE Robotics and Automation Letters*, 8(3):1183–1190, 2023.

[5] Zuxin Liu, Baiming Chen, Hongyi Zhou, Guru Koushik, Martial Hebert, and Ding Zhao. Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11748–11754. IEEE, 2020.

[6] Xiaoyu Li, Tao Xie, Dedong Liu, Jinghan Gao, Kun Dai, Zhiqiang Jiang, Lijun Zhao, and Ke Wang. Poly-mot: A polyhedral framework for 3d multi-object tracking. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9391–9398. IEEE, 2023.

[7] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Voxelnext: Fully sparse voxelnet for 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21674–21683, 2023.

[8] Yilun Chen, Zhiding Yu, Yukang Chen, Shiyi Lan, Anima Anandkumar, Jiaya Jia, and Jose M Alvarez. Focalformer3d: focusing on hard instance for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8394–8405, 2023.

[9] Yi Gu, Hongzhi Cheng, Kafeng Wang, Dejing Dou, Chengzhong Xu, and Hui Kong. Learning moving-object tracking with fmcw lidar. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3747–3753. IEEE, 2022.

[10] Xiaoyi Peng and Jie Shan. Detection and tracking of pedestrians using doppler lidar. *Remote Sensing*, 13(15):2952, 2021.

[11] Mehmet Aygun, Aljosa Osep, Mark Weber, Maxim Maximov, Cyrill Stachniss, Jens Behley, and Laura Leal-Taixé. 4d panoptic lidar segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5527–5537, 2021.

[12] Xinshuo Weng, Jianren Wang, David Held, and Kris Kitani. 3d multi-object tracking: A baseline and new evaluation metrics. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10359–10366. IEEE, 2020.

[13] Ziqi Pang, Zhichao Li, and Naiyan Wang. Simpletrack: Understanding and rethinking 3d multi-object tracking. In *European Conference on Computer Vision*, pages 680–696. Springer, 2022.

[14] Nuri Benbarka, Jona Schröder, and Andreas Zell. Score refinement for confidence-based 3d multi-object tracking. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8083–8090. IEEE, 2021.

[15] Johannes Pöschmann, Tim Pfeifer, and Peter Protzel. Factor graph based 3d multi-object tracking in point clouds. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10343–10350. IEEE, 2020.

[16] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019.

[17] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793, 2021.

[18] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Largekernel3d: Scaling up kernels in 3d sparse cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13488–13498, 2023.

[19] Yan Wang, Xiangyu Chen, Yurong You, Li Erran Li, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. Train in germany, test in the usa: Making 3d object detectors generalize. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11713–11723, 2020.

[20] Rodrigo Marcuzzi, Lucas Nunes, Louis Wiesmann, Elias Marks, Jens Behley, and Cyrill Stachniss. Mask4d: End-to-end mask-based 4d panoptic segmentation for lidar sequences. *IEEE Robotics and Automation Letters*, 2023.

[21] Mian Guo, Kai Zhong, and Xiaozhi Wang. Doppler velocity-based algorithm for clustering and velocity estimation of moving objects. In *2022 7th International Conference on Automation, Control and Robotics Engineering (CACRE)*, pages 216–222. IEEE, 2022.

[22] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

[23] Wenbang Deng, Kaihong Huang, Qinghua Yu, Huimin Lu, Zhiqiang Zheng, and Xieyuanli Chen. Elc-ois: Ellipsoidal clustering for open-world instance segmentation on lidar data. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7606–7613. IEEE, 2023.

[24] Seungcheol Park, Shuyu Wang, Hunjung Lim, and U Kang. Curved-voxel clustering for accurate segmentation of 3d lidar point clouds with real-time performance. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6459–6464. IEEE, 2019.

[25] Hsu-kuang Chiu, Chien-Yi Wang, Min-Hung Chen, and Stephen F Smith. Probabilistic 3d multi-object cooperative tracking for autonomous driving via differentiable multi-sensor kalman filter. *arXiv preprint arXiv:2309.14655*, 2023.

[26] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019.

[27] Minwoo Jung, Wooseong Yang, Dongjae Lee, Hyeonjae Gil, Giseop Kim, and Ayoung Kim. Helipr: Heterogeneous lidar dataset for inter-lidar place recognition under spatiotemporal variations. *The International Journal of Robotics Research*, 43, 04 2024.

[28] OD Team et al. Openpcdet: An open-source toolbox for 3d object detection from point clouds, 2020.

[29] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008.

[30] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision*, pages 17–35. Springer, 2016.

[31] Zhijun Pan, Fangqiang Ding, Hantao Zhong, and Chris Xiaoxuan Lu. Ratrack: moving object detection and tracking with 4d radar point cloud. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4480–4487. IEEE, 2024.