DeepEgo+: Unsynchronized Radar Sensor Fusion for Robust Vehicle Ego-Motion Estimation

Simin Zhu, *Graduate Student Member, IEEE*, Satish Ravindran, Lihui Chen, Alexander G. Yarovoy, *Fellow, IEEE*, and Francesco Fioranelli, *Senior Member, IEEE*

***Abstract*—This article studies the problem of estimating the 2-D motion state of a moving vehicle (ego motion) using millimeter-wave (mmWave) automotive radar sensors. Unlike prior single-radar or synchronized radar systems, the proposed approach (named DeepEgo+) can achieve sensor fusion and estimate ego motion using an unsynchronized radar sensor network. To achieve this goal, DeepEgo+ combines two neural network (NN)-based components (i.e., Module A for motion estimation and Module B for sensor fusion) with a decentralized processing architecture using the late fusion technique. Specifically, each radar sensor in the network has a Module A that processes its output and computes an initial motion estimate, while Module B fuses the initial estimates from all radar sensors and outputs the final estimate. This novel architecture and fusion scheme not only eliminates the synchronization requirement but also provides robustness and scalability to the system. To benchmark its performance, DeepEgo+ has been tested using a challenging real-world radar dataset, RadarScenes. The results show that DeepEgo+ provides significant performance advantages over recent state-of-the-art approaches in terms of estimation accuracy, long-term stability, and robustness against high outlier ratios and sensor failures. Furthermore, the influence of vehicle nonzero acceleration on ego-motion estimation is identified for the first time, and DeepEgo+ demonstrates the feasibility of compensating for its effect and further improving the estimation accuracy.**

***Index Terms*—Automotive radar, deep learning, ego-motion estimation, radar point cloud, radar sensor network.**

I. Introduction

OVER the past decade, millimeter-wave (mmWave) multiple-input–multiple-output (MIMO) radar has gradually become one of the indispensable sensors for self-driving vehicles and advanced driver-assistance systems (ADASs) [1]. Compared with other sensors, this type of radar has unparalleled advantages for automotive applications. For example, unlike optical sensors such as Lidar and camera, automotive radar is less sensitive to illumination and weather conditions [2]. In addition, it has lower power consumption, more affordable price, and is more compact than the "Lidar-like" scanning radar [3]. Furthermore, for detected objects, automotive radar can provide direct measurements of their range, azimuth angle, elevation angle, radial velocity, and radar cross section (RCS) [4]. Among many different radar applications and functionalities in the autonomous driving pipeline [5], vehicle ego-motion estimation and related self-localization are among the promising tasks that automotive radar can contribute to. Conventionally, vehicle motion information can be monitored through odometry sensors such as wheel encoders, inertial measurement units (IMUs), and global positioning systems (GPSs). However, these sensors are subject to a range of effects, including IMU drifting, wheel slippage, or multipath reflections [6], [7], [8]. Therefore, to compensate for the limitations of odometry sensors, many studies have developed ego-motion estimation methods based on other sensors such as cameras and Lidar [9]. Despite these sensors providing excellent localization performance, they are often very demanding on computing resources due to their high resolution [10], not to mention their critical shortcomings discussed earlier.

The above arguments make automotive radar a stronger candidate for vehicle ego-motion estimation. Radar-based methods usually estimate vehicle motion using the measured range, azimuth, and radial velocity of detected stationary objects (inliers). However, the performance of such methods is hampered by several shortcomings of radar sensors. For example, due to their low spatial resolution, radar point clouds are typically sparse and contain less geometric information about the sensed environment [11]. Therefore, directly matching consecutive radar point clouds (such as for localization methods developed for optical sensors [12]) rarely achieves good results, and customization and heuristics are often required [13]. Furthermore, the quality of radar data used for ego-motion estimation is susceptible to phenomena such as false alarms, multipath reflections, RCS fluctuations, and detection of moving objects in the scene [14]. These effects can produce a large number of detections that are not from stationary objects (i.e., outliers) and thus confuse motion estimation methods. To address this issue, recent studies [15], [16], [17] use random sampling techniques such as the random sample consensus (RANSAC) [18] or its variants [19], [20], [21]. However, these techniques are often iterative, inefficient, susceptible to high outlier ratios, and unable to distinguish slow-moving objects due to binary thresholding.

Recently, a neural network (NN)-based method, named DeepEgo [22], was proposed to deal with the above limitations. In addition, unlike other NN-based methods [23], [24], [25], DeepEgo can directly process multidimensional radar point clouds without introducing quantization errors.

Received 24 September 2024; revised 20 December 2024 and 20 February 2025; accepted 21 February 2025. Date of publication 26 February 2025; date of current version 13 March 2025. *(Corresponding author: Simin Zhu.)*

Simin Zhu, Alexander G. Yarovoy, and Francesco Fioranelli are with the Microwave Sensing, Signals and Systems (MS3) Group, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: s.zhu-2@tudelft.nl; a.yarovoy@tudelft.nl; f.fioranelli@tudelft.nl).

Satish Ravindran and Lihui Chen are with NXP Semiconductors, San Jose, CA 95134 USA (e-mail: satish.ravindran@nxp.com).

Digital Object Identifier 10.1109/TRS.2025.3546001

To benchmark, DeepEgo was evaluated using a challenging real-world radar dataset, RadarScenes [26], and the result demonstrates its superior performance compared to previous studies in terms of estimation accuracy and robustness. Even so, there are still some aspects of DeepEgo that can be further improved. First, DeepEgo is designed for single-radar scenarios, while today's cars are usually equipped with multiple unsynchronized radar sensors (i.e., forming a radar sensor network). It is, therefore, worthwhile to investigate what benefits sensor fusion can bring to ego-motion estimation and how fusion with unsynchronized radars can be achieved while still maintaining the advantages of DeepEgo. Second, even if DeepEgo proposed a novel loss function to supervise model training, it is not robust to low-quality training examples (e.g., radar point clouds associated with incorrect true vehicle motion or radar point clouds with almost no inliers). Last but not least, DeepEgo relies on radial velocity measurements of inliers to compute vehicle motion, but the impact of vehicle acceleration on radial velocity estimation has not been identified, let alone the relevant remedies for this.

To fill the above gaps, this study proposes a novel NN, named DeepEgo+, for vehicle ego-motion estimation using an unsynchronized radar sensor network. To the best of the authors' knowledge, compared with all previous studies such as [3], [20], [22], and [27], the proposed method provides the following advantages.

1) *Unsynchronization:* To work with unsynchronized radars, DeepEgo+ uses a decentralized signal processing architecture. The output of each radar node is first processed independently before implementing sensor fusion. To the best of the authors' knowledge, this is the first work in the field of radar-based ego-motion estimation working with fully unsynchronized[1] radar sensor networks.
2) *Robustness:* Several modifications to the original loss function of DeepEgo [22] are proposed to improve its robustness to low-quality training examples. In addition, the designed decentralized architecture with the late fusion approach [28] further enhances its robustness against situations such as sensor failure[2] or erroneous radar node output. Furthermore, DeepEgo+ shows significant improvement over previous studies in terms of robustness to data with high outlier ratios.
3) *Acceleration:* Although this is not one of the primary goals, DeepEgo+ demonstrates for the first time in the literature its ability to offset the impact of the vehicle's nonzero acceleration on ego-motion estimates after point cloud generation, and the normalized cross correlation (NCC) between estimation error and vehicle acceleration becomes 3.5 times smaller than the previous art, thanks to the proposed temporal NNs.

In addition to the above contributions, DeepEgo+ also inherits all the advantages of the previous DeepEgo model [22]. Furthermore, DeepEgo+ is tested using the challenging RadarScenes dataset [26], which is the same dataset used to measure the performance of DeepEgo and six other works [3], [25], [27], [29], [30], [31] selected from the literature.[3] Last but not least, a set of novel evaluation metrics and visualization tools are presented to gain a comprehensive understanding of the performance of DeepEgo+. Compared with previous works, DeepEgo+ shows a significant improvement in terms of estimation accuracy, long-term stability, and robustness against high outlier ratios and sensor failures.

The rest of this article is organized as follows. Section II presents an overview of existing research on the topic. Section III presents the detailed design of DeepEgo+. In Section IV, the performance of DeepEgo+ is measured using different evaluation metrics, and other approaches found in the literature are also tested as a comparison. Finally, Section V offers conclusions and outlines directions for future studies.

II. Related Works

This section provides an overview of the prior work in the area of vehicle localization with automotive radar and sensor fusion with automotive radar in Sections II-A and II-B, respectively. After that, a summary of the literature study is provided in Section II-C.

A. Vehicle Localization With Automotive Radar

In general, approaches for radar-based vehicle localization can be divided into two categories, namely scan-matching methods [3], [13], [30], [32], [33], [34], [35], [36] and instantaneous methods [22], [27], [37], [38], [39], [40], [41]. The main difference between these two types of methods lies in the characteristics of the radar data used for localization.

1) Scan-Matching Methods: The fundamental idea behind the scan-matching method is to analyze the relative Euclidean transformation between consecutive radar point clouds [32]. It is similar to the approaches that have been widely used for camera- and Lidar-based localization [42]. Scan-matching methods have shown promising performance on scanning radars [43]. However, due to low resolution and RCS fluctuation issues, adaptations become necessary for processing automotive radar data. For example, hard data associations between point clouds have been replaced by soft associations through normal distribution transform (NDT) [29], which represents each radar point or cluster as a Gaussian distribution [13], [32]. Measured radial velocity and returned power have been utilized to improve localization accuracy and mitigate the impact of false alarms [3], [30]. Moreover, a complex noise model has been added to the objective function [34], in order to reduce local maxima caused by the summing approximation [30]. Despite these efforts, scan-matching methods for automotive applications still have some limitations. First, scan-matching methods are usually based on optimizations, which requires good initialization and iterative processes. Second, due to the addition of heuristics, these methods often contain many manually tuned parameters that cannot be adjusted by themselves to adapt to complex driving scenarios. Last but not least, these methods can theoretically encounter "tunneling" effects when the radar is not facing forward. In this case, there are large, dense radar point clouds from nearby road curbs and vegetation that do not move between frames [22].

[1]Unsynchronized here means that the radar sensors in the network are not synchronized, i.e., the time intervals between individual radar outputs are uneven, and the radar transmitting and receiving operations are not ordered.

[2]Sensor failure is denoted as a condition when the radar sensor stops functioning properly or temporarily fails to provide accurate data.

[3]The DeepEgo paper [22] presents in more detail the performance of these six previous studies on the RadarScenes dataset.

2) Instantaneous Methods: Instantaneous methods exploit the fact that, at a given azimuth, the measured radial velocity of a stationary object can be determined by a linear transformation of the vehicle's own motion. Therefore, the main objective of these methods is to locate detections originated from stationary objects (i.e., inliers). After finding all inliers, the vehicle ego motion can be calculated by linear regression. Compared with scan-matching methods, instantaneous methods do not require multiple-radar frames (hence, "instantaneous"), avoid performing data association in sparse radar point clouds, and are less affected by RCS fluctuations. However, real-world radar data usually contain a large number of outliers, which makes locating inliers a nontrivial task. To address this issue, the RANSAC approach [18] was introduced to reject outliers before implementing regression [27], [37]. Powered by RANSAC, instantaneous methods have been the standard for ego-motion estimation in various radar applications for a decade [33], [40], [44], [45]. However, instantaneous methods with RANSAC still have some drawbacks. First, these methods can only use azimuth and radial velocity measurements to identify outliers, but previous studies [3], [22] have shown that other object characteristics (e.g., measured return power and range) can also help reject outliers. Second, these methods imply iterative procedures and some parameters that need to be adjusted according to the current outlier ratio, which makes real-time ego-motion estimation a challenging task. Third, they assume that inliers are in the majority, which makes them susceptible to scenarios with high outlier ratios. Finally, they use binary thresholding to determine inliers, which makes them vulnerable to slow-moving objects that may be misclassified as inliers. As an alternative to RANSAC-based methods, an NN-based instantaneous method named DeepEgo [22] was recently proposed.[4] Instead of using RANSAC, DeepEgo directly processes multidimensional radar point clouds using NNs to extract complex spatial features and estimate point weights, which are then used for weighted regression. Although DeepEgo can effectively address the above limitations of instantaneous methods, there are still several aspects that can be further improved. First, today's vehicles are often equipped with multiple radars that can provide different perspectives of the scene and potentially improve estimation accuracy and robustness [46]. However, DeepEgo only works for a single radar, and it is unclear how to fuse data from multiple-radar sensors, especially when the sensors are unsynchronized. Second, as mentioned earlier, the success of instantaneous methods relies on accurate measurements of the inliers' radial velocity. However, the nonzero acceleration of the vehicle can introduce inconsistent Doppler phase shifts, which leads to a wider Doppler frequency range and thus inaccurate radial velocity estimation [47]. Finally, DeepEgo proposes a novel loss function to supervise the network to learn how to assign weights to inliers and outliers. However, that loss function is not robust to the imperfections of real-world training data. For example, the training sample may be associated with a wrong ground truth, or the training sample may be in practice useless if no inliers are detected by the radar.

[4]As reported in [22], DeepEgo achieved a 51.2% improvement in translation velocity RMSE and a 49.8% improvement in rotational velocity RMSE, compared to the RANSAC-based instantaneous method.

B. Sensor Fusion With Automotive Radars

Perception sensors are key to achieve full automation in self-driving cars. As discussed in Section I, different sensors have different advantages and disadvantages. However, even for the same type of sensor, different configurations and mounting positions will also cause differences. Thus, the main objective of sensor fusion is to combine the data to overcome the limitations of individual, independent sensors [48]. To achieve sensor fusion, usually, two design choices need to be made, namely, fusion architecture and fusion technique.

1) Fusion Architecture: According to the sensor type, fusion architectures can be divided into heterogeneous sensor fusion (HTSF) [49], [50], [51], [52], [53], [54], [55] and homogeneous sensor fusion (HMSF) [37], [56], [57]. HMSF focuses on fusing the same type of sensors, while HTSF considers fusing radar with other sensor modalities such as IMU [25], [49], [51], [54], [55], camera [50], or Lidar [53]. Judging from the amount of literature, HTSF appears to have received more attention than HMSF. This is because HTSF can combine the advantages of different types of sensors and overcome their respective weaknesses. However, HMSF offers better scalability than HTSF, which makes it relatively straightforward to add more sensors to the system. In addition, by fusing multiple radars, HMSF improves the overall signal-to-noise ratio and uses redundancy to increase reliability. Furthermore, future autonomous vehicles are likely to have greater sensor redundancy, meaning there may be multiple sensors of the same and different types [58]. Therefore, it is also necessary to study HMSF and realize that future fusion architectures can be hybrid, where HMSF can first simplify the complexity of data integration and reduce computing power, and then HTSF combines the advantages of different modalities.

2) Fusion Technique: Fusion techniques determine at what point of a data processing chain the sensor data are fused. These techniques can generally be divided into early fusion [37], [57], halfway fusion [25], [50], and late fusion. Early fusion combines sensor data at a very early stage, such as [37] fusing multiple-radar point clouds to estimate the vehicle ego motion. Halfway fusion performs fusion at an intermediate stage, for example, work [25] uses different NNs to extract features from radar and odometry sensors separately and then fuses them. Late fusion combines the final output of each sensor after processing the data from each sensor separately. Therefore, it is clear that the later the fusion occurs, the less information is preserved. However, this also means that early and halfway fusion can be computationally expensive. Moreover, in order to fuse data at an early stage, early and halfway fusion usually requires good sensor synchronization [37] or restrictions on radar configuration and mounting location [56]. In contrast, late fusion is more robust to sensor failures than other fusion techniques because each sensor is processed independently, making it a preferred choice for safety-critical applications. Furthermore, the computational cost of late fusion techniques increases linearly with the number of sensors.

C. Summary

Based on the literature study, the following conclusions can be drawn. First, instantaneous methods show better potential

than scan-matching methods, especially the recently proposed DeepEgo [22]. However, to work with future sensor networks and handle complex driving scenarios and challenging radar data, DeepEgo+ combines loss function updates, decentralized architecture, and improved robustness to address the identified gaps. Second, although HTSF has received more attention than HMSF in the literature, future fusion architectures may require both HTSF and HMSF, which makes the current study meaningful. Finally, for driving safety and for unsynchronized radar sensor networks, it is better to use late fusion instead of other fusion techniques.

III. Methodology

This section presents the design of the proposed method DeepEgo+ for vehicle ego-motion estimation using multiple unsynchronized automotive radars.

A. Problem Formulation

This article focuses on the problem of estimating vehicle 2-D ego motion using unsynchronized radar sensors in real traffic scenarios. Given an unsynchronized radar sensor network mounted on a moving vehicle, assuming that each radar has a linear array and the vehicle is not sliding sideways, after signal preprocessing [59], each radar outputs a multidimensional radar point cloud $P_{t,i}^{L\times M}$, where t is the global timestamp, i is the radar index, L is the number of radar points, and M is the number of measured object features. Assuming that $P_{t,i}^{L\times M}$ contains no outliers, given the measured radial velocity $d_{t,i}^{l}$ and azimuth angle $\alpha_{t,i}^{l}$ (i.e., $M \geq 2$), the following equation holds:

$$\begin{bmatrix} -d_{t,i}^{1} \\ \cdots \\ -d_{t,i}^{L} \end{bmatrix} = \begin{bmatrix} \cos\left(\alpha_{t,i}^{1}\right) & \sin\left(\alpha_{t,i}^{1}\right) \\ \cdots & \cdots \\ \cos\left(\alpha_{t,i}^{L}\right) & \sin\left(\alpha_{t,i}^{L}\right) \end{bmatrix} \cdot \begin{bmatrix} v_{t,i}^{x,\mathrm{gt}} \\ v_{t,i}^{y,\mathrm{gt}} \end{bmatrix} \tag{1}$$

where $v_{t,i}^{x,\mathrm{gt}}$ and $v_{t,i}^{y,\mathrm{gt}}$ are the ground truth 2-D motion components of the radar in radar coordinates. Equation (1) can be simplified as

$$D_{t,i} = A_{t,i} \cdot V_{t,i}^{\mathrm{gt}}. \tag{2}$$

To estimate $V_{t,i}^{\mathrm{gt}}$, the least-squares method (LSQ) can be used

$$V_{t,i}^{\mathrm{est}} = \left(A_{t,i}^{T} A_{t,i}\right)^{-1} A_{t,i}^{T} D_{t,i}. \tag{3}$$

However, real-world radar data contain a large number of outliers that affect the LSQ solution. Conventional instantaneous methods use RANSAC to address this issue, while the recent work DeepEgo [22] suggests to use the weighted LSQ (w-LSQ)

$$V_{t,i}^{\mathrm{est}} = \begin{bmatrix} v_{t,i}^{x} \\ v_{t,i}^{y} \end{bmatrix} = \left(A_{t,i}^{T} W_{t,i}^{\mathrm{est}} A_{t,i}\right)^{-1} A_{t,i}^{T} W_{t,i}^{\mathrm{est}} D_{t,i} \tag{4}$$

$$W_{t,i}^{\mathrm{est}} = \begin{bmatrix} w_{t,i}^{1,\mathrm{est}} & 0 & 0 \\ 0 & \cdots & 0 \\ 0 & 0 & w_{t,i}^{L,\mathrm{est}} \end{bmatrix}. \tag{5}$$

$W_{t,i}^{\mathrm{est}}$ is a diagonal matrix with L weights estimated by DeepEgo. Given the radar motion, the vehicle ego motion $V_{t,\mathrm{car}}^{\mathrm{est}}$ can be computed by

$$V_{t,\mathrm{car}}^{\mathrm{est}} = \begin{bmatrix} v_{t}^{x} \\ \omega_{t} \end{bmatrix} = \begin{bmatrix} v_{t,i}^{x}\cos(\theta_i) - v_{t,i}^{y}\sin(\theta_i) + y_i\omega_t \\ \dfrac{1}{x_i}\left(v_{t,i}^{y}\cos\left(\theta_i\right) + v_{t,i}^{x}\sin\left(\theta_i\right)\right) \end{bmatrix} \tag{6}$$

where v_t^x is the vehicle translational velocity; ω_t is the rotational speed; and x_i, y_i, and θ_i are the mounting position and angle of the ith radar with respect to the center of the vehicle. Based on the sensor network, given T consecutive timestamps, (6) can deliver T vehicle velocity estimates

$$\left\{V_{1,\mathrm{car}}^{\mathrm{est}}, V_{2,\mathrm{car}}^{\mathrm{est}}, \ldots, V_{T,\mathrm{car}}^{\mathrm{est}}\right\}. \tag{7}$$

However, it is worth repeating that the above velocity estimates can be inaccurate due to the issues mentioned in Section II-A. To further improve this, as concluded in Section II-C, HMSF with the late fusion technique can be applied. Since the vehicle motion has temporal correlation across frames, the Kalman filter (KF) [60] can be used as one of the alternative solutions to fuse these motion estimates. Assuming a linear motion model and additive Gaussian perturbation, the process model and measurement model can be expressed as follows:

$$\begin{aligned} x_t &= F_t x_{t-1} + q_t, \quad q_t \sim N(0, Q_t) \\ y_t &= G_t x_t + r_t, \quad r_t \sim N(0, R_t) \end{aligned} \tag{8}$$

where x_t is the vehicle kinematic state, y_t is the velocity initial estimate $V_{t,\mathrm{car}}^{\mathrm{est}}$ from (7), F_t is the transition matrix, and G_t is the measurement matrix. q_t and r_t represent the process and measurement noise, respectively. In addition, q_t and r_t are assumed to be white, independent, and Gaussian distributed with covariance matrix Q_t and R_t. Therefore, according to [61], the vehicle motion can be computed through

$$\begin{aligned} \hat{x}_{t|t-1} &= F_t \hat{x}_{t-1|t-1} \\ H_{t,t-1} &= F_t H_{t-1,t-1} F_t^T + Q_t \\ K_t &= H_{t,t-1} G_t^T \left(G_t H_{t,t-1} G_t^T + R_t\right)^{-1} \\ \hat{x}_{t|t} &= \hat{x}_{t|t-1} + K_t\left(y_t - G_t \hat{x}_{t|t-1}\right) \\ H_{t,t} &= \left(I - K_t G_t\right) H_{t,t-1} \end{aligned} \tag{9}$$

where $\hat{x}_{t|t-1}$ is the predicted state estimate, $H_{t,t-1}$ is the predicted estimate covariance, K_t is the Kalman gain, $\hat{x}_{t|t}$ is the updated state estimate, and $H_{t,t}$ is the updated estimate covariance. In order to compute $\hat{x}_{t|t}$, it is essential to know F_t, G_t, Q_t, R_t, and y_t. While F_t and G_t can be determined based on the assumed motion model and timestamps, y_t is related to the point cloud weight $W_{t,i}^{\mathrm{est}}$ as in (4), and Q_t and R_t are unknown and often require adaptive adjustment.

B. Proposed Architecture Overview

Fig. 1 presents the architecture of DeepEgo+. It is a decentralized signal processing architecture with two NN-based modules: Module A for motion estimation and Module B for fusion. Module A is implemented at each radar node and, similar to DeepEgo [22], can directly process multidimensional radar point clouds and output initial motion estimates. However, with the proposed modifications, Module A addresses several limitations of DeepEgo. For example, with the new loss function, Module A is more robust to low-quality training examples in the dataset. Also, Module A employs additional NN layers to reduce the complexity of input features so that it can process data from all radars and share the same parameters among different nodes. This parameter sharing significantly reduces the number of training parameters and keeps the network size constant even if the number of radar

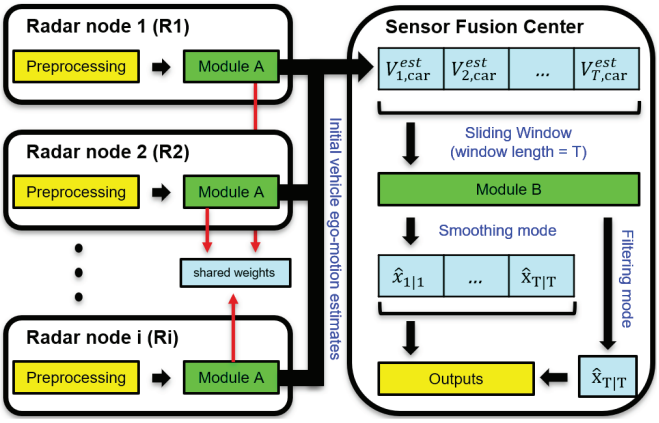


Fig. 1. Architecture of DeepEgo+. It is a decentralized processing architecture with two NN-based components, Modules A and B. Each radar has Module A that processes its output and calculates an initial velocity estimate. After collecting T initial estimates, with late fusion, Module B fuses them and outputs the update for multiple timestamps (smoothing mode), or the latest timestamp (filtering mode).

nodes increases. After collecting T initial estimates from Module A, Module B fuses them using the proposed NN-based KF. The NN-based KF mimics the procedure of the conventional KF to exploit the temporal correlation in vehicle motion and provide smoother estimates. The difference is that it avoids laborious parameter tuning and instead uses temporal NNs to adaptively estimate the Kalman gain. In addition, as shown in Section IV-G, the temporal NNs can learn to offset the effects of nonzero vehicle acceleration. Moreover, Module B can be used as a smoothing function that provides T velocity estimates or as a filter that updates the velocity based on the latest timestamp.

In summary, DeepEgo+ provides an end-to-end solution for ego-motion estimation using unsynchronized radar sensor networks. Through a special network design, it addresses the previous limitations mentioned in Section II. Furthermore, the proposed decentralized processing architecture helps reduce the computational overhead and improve the runtime performance by distributing the processing tasks across multiple-radar nodes. The remainder of this section presents its design details. Specifically, the modifications that upgrade DeepEgo [22], which works for single radar and single frame, to DeepEgo+, which works for multiradar and multiframe, will be presented.

C. Improved Loss Function

This section presents the proposed modifications for the loss function of DeepEgo. To guide model training, DeepEgo's loss function consists of three parts, namely the motion loss (ML), Doppler loss (DL), and sample weight (SW). The final loss function of DeepEgo ($\text{FLF}_{\text{DeepEgo}}$) is computed as

$$\text{FLF}_{\text{DeepEgo}} = (\text{ML} + \mu \cdot \text{DL}) \cdot \text{SW} \tag{10}$$

where ML measures the mean squared error (mse) between estimated motion and ground truth motion, DL teaches the model to separate outliers and inliers, μ is a predetermined weighting factor, and SW adjusts the final loss of each training example. The three parts can be expressed as follows:

$$\begin{aligned}\text{ML} &= \left[\text{mse}\left(V^{\text{gt}}_{1,i}, V^{\text{est0}}_{1,i}\right), \ldots, \text{mse}\left(V^{\text{gt}}_{T,i}, V^{\text{est}}_{T,i}\right)\right] \\ \text{DL} &= \left[\text{mse}\left(W^{\text{gt}}_{1,i}, W^{\text{est}}_{1,i}\right), \ldots, \text{mse}\left(W^{\text{gt}}_{T,i}, W^{\text{est}}_{T,i}\right)\right] \\ \text{SW} &= \left[\sum_{l=1}^{L}\left(w^{l,\text{gt}}_{1,i}\right), \ldots, \sum_{l=1}^{L}\left(w^{l,\text{gt}}_{T,i}\right)\right]\end{aligned} \tag{11}$$

where $w^{l,\text{gt}}_{t,i}$ is the ground truth point weight computed based on $V^{\text{gt}}_{t,i}$ [22]. Although DeepEgo's performance relative to previous works has demonstrated the effectiveness of its loss function, there are still several aspects that can be further improved. First, ML uses mse to measure error, which causes the model to overfit to large errors caused by extreme or special cases. Second, by simply summing the weights of all points, SW advocates focusing on training examples with low outlier ratios. In fact, examples with reasonably high outlier ratios are also important to consider, as traditional instantaneous methods tend to fail in these cases. Third, DL tries to minimize the squared distance between estimated point weights to all ground truth weights. Consequently, the model will also try to match detection points with smaller weights caused by slow-moving objects. This can bias the final wLSQ solution and also can lead to overfitting. Finally, $\text{FLF}_{\text{DeepEgo}}$ sums ML and DL with a weighting factor μ, which can penalize the case when DL is large but ML is small. However, due to the effect of vehicle acceleration (described in Section IV-G), after motion filtering and acceleration compensation, DL can remain large but ML can become smaller. To address the above issues, the following modifications are thus suggested.

1) The new ML (ML_{new}) uses the Huber loss [62] instead of mse, which is more robust to outlier scenarios.
2) The new SW (SW_{new}) calculates the mean of all inlier weights to represent the quality of inliers. In addition, to completely eliminate the influence of low-quality training examples (as shown in Fig. 2), SW_{new} is set to zero when the mean weight or the number of inliers is below a given threshold.
3) The new DL (DL_{new}) measures the mse between the K points with the largest weights in $W^{\text{est}}_{t,i}$ and their actual weights in $W^{\text{gt}}_{t,i}$. In addition, the w-LSQ solution, as shown in (4) and (5), is computed using the selected K points instead of all L points.
4) The final loss function of DeepEgo+ ($\text{FLF}_{\text{DeepEgo+}}$) is finally a multiplication of ML_{new}, DL_{new}, SW_{new}, and the weighting term μ is dropped

$$\text{FLF}_{\text{DeepEgo+}} = \text{ML}_{\text{new}} \cdot \text{DL}_{\text{new}} \cdot \text{SW}_{\text{new}}. \tag{12}$$

For simplicity, in the rest of this article, DeepEgo with the above proposed modifications on its loss function is named *Model V1*. Similar to DeepEgo, *Model V1* estimates ego motion based on input from the same single radar. However, a radar sensor network consists of multiple-radar sensors installed at different locations with different viewing angles. As illustrated in Fig. 1, to handle the complex input data features caused by different radar nodes, the next section proposes several modifications to the architecture and input structure of *Model V1*, beyond the aforementioned changes to the loss function.

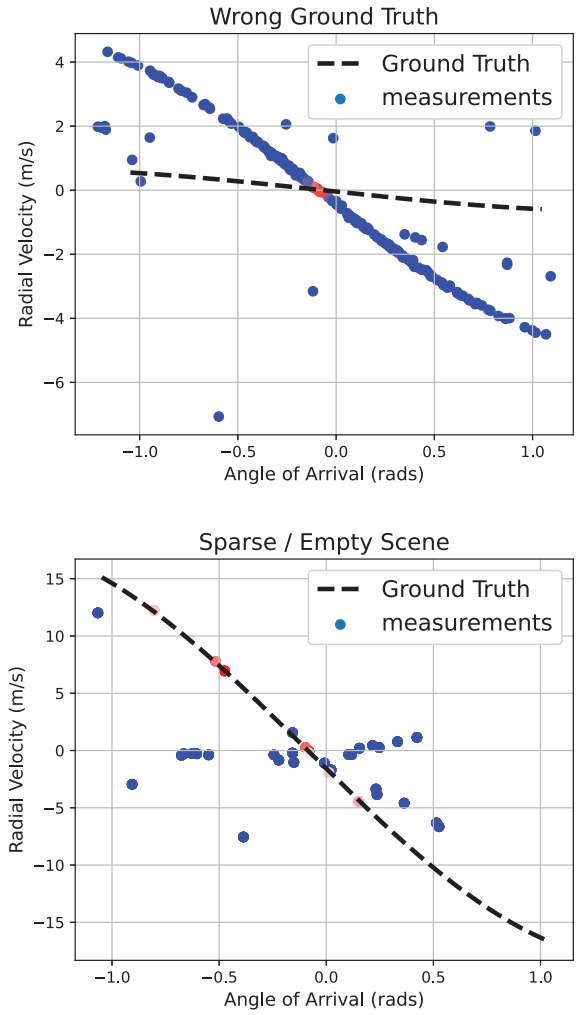


Fig. 2. Examples of imperfect real-world radar data from the RadarScenes dataset [63]. For both figures, radar measurements are denoted as colored dots (red inliers and blue outliers), and the black dashed line is computed based on the current ground truth vehicle motion and AoA measurements. The top figure shows what happens when the ground truth velocity is incorrect and the bottom figure provides an example of a sparse/empty radar point cloud.

D. Module A: Processing Multiradar Nodes

In our previous work, DeepEgo [22] is trained and tested separately and independently for each radar. Experimental results show that its performance degrades when using different radars for training and testing. In general, the more training data, the better performance a deep learning model should produce. However, further experiments indicate that when DeepEgo is trained and tested using data from the same radar sensor network, its performance decreases compared to the independently trained model. The reason behind this phenomenon is that as the training data increase, the feature complexity also increases. This is because each radar in the sensor network has a different installation location, which produces different feature patterns in the input radar point cloud.

To address this challenge, T-net [64] and a radar encoding step are added to *Model V1*, and the final NN is named Module A. Fig. 3 shows the architecture of Module A as well as explains the difference in the architecture between DeepEgo, *Model V1*, and Module A. T-net is an NN that

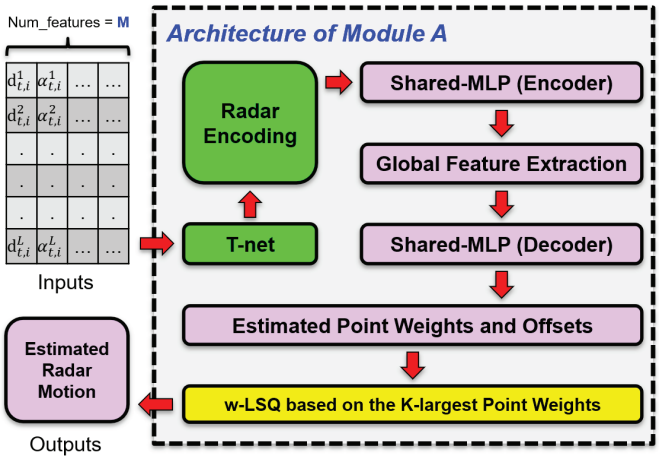


Fig. 3. Architecture of Module A. Purple blocks are included in DeepEgo [22]. As explained in Section III-C, *Model V1* computes the w-LSQ solution based on the K-largest point weights, with the relevant modification shown in yellow. Based on *Model V1*, Module A adds T-net [64] and a radar encoding step (denoted in green) to handle arbitrary radar inputs.

takes a multidimensional radar point cloud as input, outputs a matrix, and uses the matrix to perform an affine transformation on the input point cloud. This step can reduce the complexity of input features by aligning point clouds captured by different radars into a canonical space. Afterward, a radar encoding step is implemented to concatenate a predefined number (e.g., with two radars the numbers would be "0" and "1") with the output of the affine transformation. This radar encoding step can further help Module A distinguish among different feature patterns caused by different radars. As shown in the results in Section IV-C, Module A can improve performance even when the model is trained with more complex data. However, it is important to acknowledge that Module A can only work with known radar sensor networks. For testing using data from "unseen" radar nodes, further developments are required.

E. Module B: Processing Multiradar Frames

As illustrated in Fig. 1, the outputs of Module A are the vehicle motion estimates, arranged in time order. Although these motion estimates come from different radar nodes, they are all sampled from the continuous motion of the vehicle. Therefore, to exploit the advantage of multiple radars and the prior knowledge of the vehicle motion model, this work proposes to use a KF to fuse these initial estimates. Following (9) for the prediction and update steps of the KF, although most variables can be determined, the covariance matrices Q_t and R_t are unknown and usually need to be adaptively adjusted based on the certainty of the model predictions and measurements. Instead of manually tuning the covariance matrix, inspired by [65], this work proposes using NNs to estimate the Kalman gains. In this way, the knowledge of the underlying noise statistics is no longer required. In addition, based on the input, the estimated Kalman gain is able to automatically determine how much weight the filter gives to predictions (from the model) versus measurements (from the radar) when updating the state estimate.

Fig. 4 presents the architecture of the proposed NNs (named collectively as Module B). Module B consists of three NN-based functional components: 1) a shared multilayer

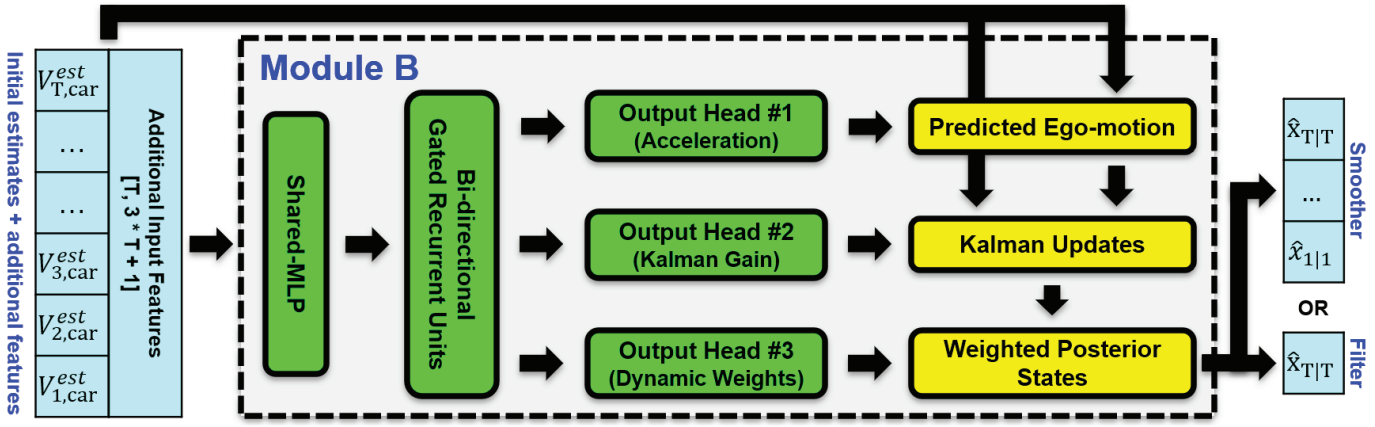


Fig. 4. Architecture of the proposed NNs for sensor fusion (named Module B in this work). Within Module B, only the green blocks are built by NNs. Module B takes the initial vehicle motion estimates provided by Module A as its input. Then, it uses bidirectional gated recurrent units (bi-GRUs) to extract temporal features from motion estimates. These features are further processed by three output heads, each of which estimates the matrices used to compute the Kalman prediction, Kalman update, and weighted output. Finally, Module B can also be used as a smoother to output T motion estimates or as a filter to update the most recent estimate.

perceptron (shared-MLP); 2) the bidirectional gated recurrent units (bi-GRUs); and 3) three output heads. The shared-MLP takes T initial vehicle motion estimates, T radar encoding labels, and three $T \times T$ feature matrices as its input. The initial vehicle motion estimate is provided by Module A and the radar encoding label is the same as described in Section III-D. Regarding the feature matrices, the first two are generated through the outer subtraction[5] for the vector of initial motion estimates and recorded timestamps. The third feature matrix is given by the division between the first two feature matrices and represents the estimated vehicle acceleration. The shared-MLP is used to preprocess the raw input data and transform it into a representation that is more suitable for the subsequent bi-GRUs. The bi-GRUs are recurrent NNs (RNNs) that can extract temporal features from continuous vehicle motions. Unlike traditional unidirectional RNNs, the bidirectional architecture allows leveraging information from previous motion states and upcoming motion states to make predictions about the current state. With this feature, Module B with sliding window input can be used not only as a filter but also as a smoother. The output of the bi-GRUs is further processed by three output heads, each consisting of two layers of the shared-MLPs. The first output head estimates a $T \times T$ acceleration matrix $\boldsymbol{\tau}$. The acceleration matrix has the same structure as the previously mentioned third feature matrix and contains the same type of information; it can be expressed as follows:

$$\boldsymbol{\tau} = \begin{bmatrix} \tau_{1\leftarrow 1} & \tau_{1\leftarrow 2} & \cdots & \tau_{1\leftarrow T} \\ \tau_{2\leftarrow 1} & \tau_{2\leftarrow 2} & \cdots & \tau_{2\leftarrow T} \\ \cdots & \cdots & \cdots & \cdots \\ \tau_{T\leftarrow 1} & \tau_{T\leftarrow 2} & \cdots & \tau_{T\leftarrow T} \end{bmatrix}. \tag{13}$$

Each element τ represents an estimated acceleration value, and the index notation (for example "$\kappa \leftarrow \lambda$") indicates that this element is used to make a prediction/update at timestamp κ using previous/future states from timestamp λ. With the same structure, the second output head estimates a $T \times T$ Kalman gain matrix, and the third outputs a $T \times T$ dynamic weight matrix. Therefore, following the yellow blocks in Fig. 4, the prediction step is achieved by multiplying the second feature matrix representing the time difference with the acceleration matrix $\boldsymbol{\tau}$ and then added with the initial motion estimates. Afterward, following the terminology used in KF theory, the innovation matrix is computed using the predicted motion states and the initial motion estimate as measurements. The innovation matrix is then multiplied with the Kalman gain matrix, outputting a correction matrix, which is then added to the predicted motion state forming an updated (posterior) state matrix. The updated state matrix has the same structure and formulation as the matrix in (13), where each row represents T potential updates for the timestamp κ. However, only one update is needed for each timestamp. To solve the data association problem, the dynamic weight matrix is multiplied with the updated state matrix. The weight matrix, whose rows sum to 1, acts as a soft data association, combining all potential updates via a weighted average. Finally, the output of Module B is a vector of T updated states, and the output can be used as a smoother or a filter.

[5]The outer subtraction is an operation that subtracts every element of a vector from every other element, resulting in a matrix with zeros on the diagonal.

F. Implementation Details

As explained in Section III-B, DeepEgo+ consists of two key components: Modules A and B. Module A is used to process multiple-radar nodes and compute initial motion estimates, and Module B fuses these initial estimates using NN-based KF. Their relevant implementation details are as follows.

1) Module A: First, the new DL and the new w-LSQ scheme are computed using the K-largest point weights. In this study, K is set to 224 based on empirical evaluation on the RadarScenes dataset [26]. Second, the T-net [64] used in Module A has three shared-MLPs as an encoder, one max-pooling layer, and another three shared-MLPs as a decoder for the output. Third, for a sensor network with I radars, the radar encoding mentioned in Sections III-D and III-E is a natural number ranging from 0 to $I - 1$. Fourth, the parameters of all purple blocks of Module A in Fig. 3 are presented in

the original DeepEgo paper [22]. Finally, as mentioned in Section III-C, SW_{new} is set to zero when the mean weight or the number of inliers is below 0.4 and 40, respectively.[6]

2) Module B: First, all shared-MLPs in this work have the same structure, which consists of a 1-D convolutional layer, a batch normalization layer, and a ReLU layer for nonlinearity. Second, to reduce overfitting, a dropout layer with a dropout rate of 0.2 was attached to the bi-GRUs. Third, for the three output heads in Module B, the top layer of the first head (acceleration) has no activation function (linear output), the top layer of the second (Kalman gain) uses the hyperbolic tangent activation function, and the third head (dynamic weights) applies the softmax activation function.

IV. Results and Discussion

This section presents the evaluation results of the proposed method DeepEgo+. Moreover, methods from the literature are evaluated and compared.

A. Radar Dataset and Evaluation Metrics

This study uses the RadarScenes [63] dataset for performance evaluation. In total, 64 scenes are selected for model training and testing. These scenes contain different driving scenarios and have an equivalent drive length of over 79 km. Each radar scene contains four sequences captured by four automotive radars, later denoted as "R_i," with $i \in [1, 4]$. Each radar sequence consists of multiple-radar frames, and each radar frame is represented by a 4-D radar point cloud, including angle of arrival (AoA), radial velocity, range, and returned power. Finally, it is worth mentioning that each radar frame is associated with a unique global timestamp t, and the order of firing of the four radars is not predetermined and hence random (i.e., unsynchronized sensor network). Since the selected dataset is the same as in our previous study DeepEgo [22], readers interested in a more detailed description of how the dataset is used are referred to that article. Moreover, readers with questions related to the structure and specifications of the dataset are suggested to consult [63]. In addition to the challenging radar dataset, this study also proposes to use a set of suitable metrics to evaluate the performance of the proposed method. They are given as follows.

1) *Root-Mean-Square Error (RMSE):* It is the square root of the mean of the square of all errors between ground truth and model predictions. It is a good metric for measuring estimation accuracy, but it can be very sensitive to "outliers." For example, comparing the RMSE of two ego-motion estimators, the outcome will be biased based on their performance on "bad" testing data. For instance, as illustrated in Fig. 2, bad testing data can be radar frames with incorrect ground truth or frames captured, resulting in empty/sparse scenes. Those scenarios can exist in real-world radar datasets. Although not common, when this occurs, the estimator under test can produce an excessively large RMSE. Therefore, when bad testing data are present, it becomes unfair to assess the performance of different approaches primarily based on the RMSE metric.

[6]Note that these two hyperparameters are determined empirically and they may change slightly if the model is trained with a different radar dataset.

2) *Saturated-RMSE (S-RMSE):* It is a truncated version of RMSE. This metric is inspired by the metric used for multiple-object tracking (MOT) [66]. In MOT, when the distance between the estimated and true object location exceeds a certain range, the object is classified as misdetected and assigned to a fixed error. Similarly, this idea can also be used to reduce the impact of bad testing data. Therefore, in this study, the S-RMSE is proposed and defined as follows:

$$\text{S-RMSE}(\mathbf{x}, \hat{\mathbf{x}}) = \sqrt{\frac{1}{T}\sum_{t=1}^{T} d_t^2} \tag{14}$$

where

$$d_t = \begin{cases} x_t - \hat{x}_t, & |x_t - \hat{x}_t| \le c \\ s, & |x_t - \hat{x}_t| > c \end{cases} \tag{15}$$

where x_t and $\hat{x}_t$ are the ground truth and estimated egomotion at timestamp t, respectively; T is the total number of timestamps (radar frames); c is the predefined range of considered errors; and s is the fixed error assigned when the error is larger than the predefined range. This article sets c and s to the same value. For measuring errors in translational velocity evaluation, both variables are set to 0.5 m/s, and for rotational velocity, they are set to 2.86°/s.

3) *Median Absolute Error (MedAE):* It is the median of the absolute values of all errors. The median operator makes it less sensitive to extreme outliers in the dataset. Moreover, since it takes the middle value of the error values, it highlights the central tendency of the errors.

4) *Mean Absolute Error (MAE):* It takes the mean of the absolute values of all errors. It directly shows the average magnitude of errors between estimation and ground truth. Also, it is less sensitive to outliers compared to RMSE because all errors are assigned with equal weight.

5) *Relative Trajectory Error (RTE)*: It measures the difference between the estimated trajectory and the ground truth trajectory. Therefore, it reflects the long-term stability of the method under test. Since this metric has been introduced in [22], readers are referred to its detailed description there. In this article, RTE_50 is used.

Apart from the challenging radar dataset and comprehensive evaluation metrics, this study also uses a strict testing approach, namely the "leave-one-out" (L1O) method. Specifically, if a scene is selected as the test scene, the radar sequences captured in that scene are not used for model training and validation and are therefore "unseen" to the trained model. This helps assess the generalization ability of the proposed network, as NNs may overfit the training and validation data.

B. Single Radar and Single Frame

As detailed in Section III, from the previous DeepEgo [22] to the current DeepEgo+, there are three intermediate steps. The first step is to modify its loss function and the wLSQ scheme (referred to as *Model V1*). Then, its network structure is modified in order to process data from multiple radars (using the previously defined *Module A*). Finally, to

TABLE I

PERFORMANCE COMPARISON BETWEEN DEEPEGO [22] AND THE PROPOSED MODEL V1, FOR SINGLE-RADAR SINGLE-FRAME SCENARIOS. THEIR PERFORMANCE IS MEASURED USING THE RADAR SEQUENCES FROM THE FIVE TESTING SCENES ALSO SELECTED IN [22]. WITH FOUR AUTOMOTIVE RADARS AND FIVE SCENES, THE METRICS OF EACH METHOD ARE COMPUTED BASED ON THE AVERAGE OF 20 RADAR SEQUENCES

	Translation Velocity (cm/s)				Rotational Velocity (deg/s)				RTE
Metrics	RMSE	S-RMSE	MedAE	MAE	RMSE	S-RMSE	MedAE	MAE	RTE_50
DeepEgo [22]	11.3	9.3	4.6	6.8	0.77	0.74	0.36	0.53	9.5
Model V1	11.6	9.1	4.4	6.6	0.72	0.67	0.32	0.48	8.5
Improvement(%)	-1.9%	+2.1%	+5.4%	+3.0%	+5.9%	+8.6%	+11.1%	+10%	+9.8%

TABLE II

PERFORMANCE COMPARISON BETWEEN DEEPEGO [22] AND MODULE A. FOR DEEPEGO, THE MODEL IS TRAINED AND TESTED SEPARATELY FOR EACH RADAR, WHILE MODULE A IS TRAINED AND TESTED USING DATA FROM ALL FOUR RADARS. THE METRIC S-RMSE IS USED TO MEASURE ERRORS IN TRANSLATIONAL VELOCITY ESTIMATION ("TRANSL. ERR" IN CENTIMETER/SECOND) AND ROTATIONAL VELOCITY ESTIMATION ("ROT. ERR" IN DEGREE/SECOND). THE FINAL RESULT IS AVERAGED OVER FIVE TESTING RADAR SEQUENCES FROM THE FIVE SELECTED SCENES

Metrics: S-RMSE		Tested using R1		Tested using R2		Tested using R3		Tested using R4	
		Transl. ERR	Rot. ERR	Transl. ERR	Rot. ERR	Transl. ERR	Rot. ERR	Transl. ERR	Rot. ERR
DeepEgo [22]	Trained using R1	9.5	0.64	30.4	2.37	43.1	2.62	43.4	2.04
	Trained using R2	41.4	2.27	9.2	0.88	37.0	2.50	42.6	2.32
	Trained using R3	44.3	2.27	36.5	2.57	8.7	0.81	37.9	2.12
	Trained using R4	44.3	1.99	43.1	2.60	33.1	2.41	9.9	0.63
Module A	Trained using All	9.2	0.62	8.9	0.77	8.5	0.71	9.3	0.59

fuse information from multiple radar nodes at multiple timestamps, an NN-based KF/smoother is proposed (the previously defined *Module B*). Although DeepEgo+ is the combination of Modules A and B, it is helpful to understand how model performance changes with these intermediate steps. Therefore, as an ablation study, the evaluation results of these steps are presented in the current and the following sections.

Table I presents the performance comparison between DeepEgo and Model V1. With the proposed modifications, Model V1 outperforms DeepEgo in almost all evaluation metrics, except the RMSE metric in translational velocity estimation. However, this outcome is expected since the proposed loss function (i.e., the Huber loss and SW_{new}) makes the model less affected by large errors caused by bad training data. In addition, the proposed S-RMSE metric reflects how these rare bad data in the test set can affect the final evaluation results. Compared to the RMSE metric, Model V1 scores better and has a higher percentage improvement in the S-RMSE metric for both translational and rotational ego-motion estimations.

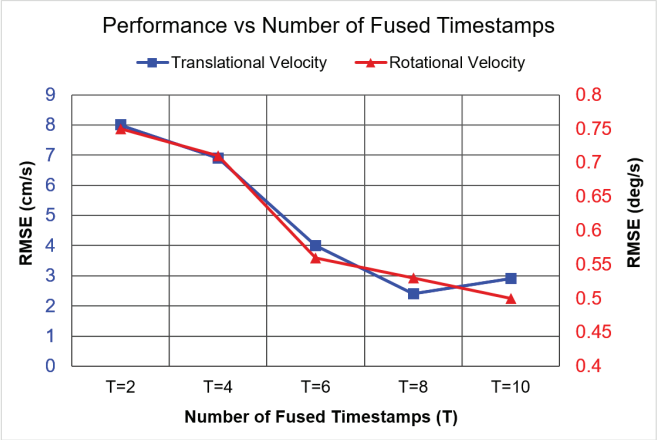


Fig. 5. Performance of DeepEgo+ under different numbers of fused timestamps (T). The left vertical axis shows the relationship between T and RMSE in translational velocity estimation, and the right axis provides the error in rotational velocity estimation.

C. Multiradar and Single Frame

This section presents the evaluation result when comparing between DeepEgo [22] and Module A. As shown in Table II, Module A performs better than DeepEgo in the S-RMSE metric. In addition, it is able to estimate the ego motion using data from four radars installed at different locations on the vehicle. It is worth mentioning that, due to different aspect angles, the relationship between ego motion and the measurements of radial velocity and AoA can vary significantly from radar to radar. Consequently, this diversity adds additional challenges to the NN model due to the increased complexity of input features, which leads to performance degradation when training and testing using data from radars mounted at different locations. However, with the proposed modifications, the model performance remains similar despite the increased complexity, and the model only needs to be trained once.

D. Multiradar and Multiframe

As previously depicted in Fig. 1, DeepEgo+ consists of two components, Modules A and B used jointly. Module A processes data coming from each different radar sequentially according to their global timestamps. It estimates the point weights and computes initial velocity estimates through wLSQ. After that, Module B takes multiple initial estimates as its input and updates the final estimation through smoothing or filtering operations. As the number of timestamps (or initial estimates) to Module B is adjustable, it becomes one of the critical hyperparameters. Clearly, this parameter determines the number of timestamps, denoted as T, to be fused, and also determines how much past and future information can be used for estimation. As shown in Fig. 5, the translation and rotational velocity estimation errors decrease as T increases. This is logical because the more information the model has

TABLE III

COMPREHENSIVE COMPARISON EVALUATED USING ALL 64 SCENES IN THE RADARSCENES DATASET [63] AND THE FIVE PROPOSED METRICS. THE PROPOSED METHOD DEEPEGO+ IS TESTED UNDER BOTH ITS FILTERING MODE AND SMOOTHING MODE. ALSO, THE PREVIOUS APPROACH DEEPEGO [22] AND THE PROPOSED MODULE A ARE ADDED FOR COMPARISON. THE "IMPROVEMENT" ROW IS COMPUTED BASED ON THE ORIGINAL METHOD IN [22] VERSUS THE PROPOSED METHOD DEEPEGO+ IN ITS SMOOTHING MODE

	Translation Velocity (cm/s)				Rotational Velocity (deg/s)				RTE
Metrics	RMSE	S-RMSE	MedAE	MAE	RMSE	S-RMSE	MedAE	MAE	RTE_50
DeepEgo [22]	11.5	9.0	4.7	6.7	0.85	0.74	0.40	0.56	11.1
Module A	10.6	8.6	4.4	6.3	0.73	0.66	0.34	0.48	9.9
DeepEgo+ [filtering]	6.1	4.2	2.0	3.0	0.51	0.49	0.23	0.34	6.4
DeepEgo+ [smoothing]	5.3	3.6	1.6	2.5	0.44	0.41	0.20	0.29	6.4
Improvement (%)	+53.9%	+60.0%	+66.0%	+62.7%	+48.2%	+44.6%	+50.0%	+48.2%	+41.4%

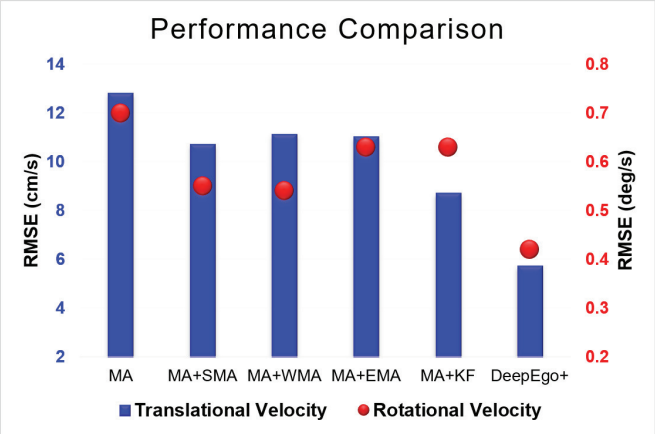


Fig. 6. Performance of the proposed approach DeepEgo+ and other fusion methods selected for comparison, including simple moving average (SMA), weighted moving average (WMA), exponential moving average (EMA), and KF. For a rigorous and fair comparison, these selected methods take the output of the proposed Module A (MA) and DeepEgo+ is set to its filtering mode. With five test scenes and four radars, the RMSE metric presented in the chart is measured and averaged over 20 testing radar sequences.

about the vehicle's motion, the more accurate estimates it can provide. However, it is not worth setting T to a too large value, as not only this will take up more memory space, but it will also have diminishing returns. Therefore, unless otherwise specified, based on these results, T is set to 8 in the following experiments.

As, to the best of our knowledge, there is currently no research on ego-motion estimation using unsynchronized radar sensor networks, in order to benchmark DeepEgo+, several traditional fusion methods are selected. Since these traditional fusion methods cannot directly process radar point clouds, the output of Module A is used as their input. Therefore, these selected methods are compared with the proposed Module B. As shown in Fig. 6, the proposed method outperforms the other approaches by a large margin, for both translational and rotational ego-motion estimation. Furthermore, unlike other methods, the proposed approach does not require laborious human engineering to adjust parameters after model training. Finally, as discussed later in Section IV-G, there is a fundamental reason/limitation why the proposed method outperforms other traditional approaches in dealing with nonzero acceleration, and to the best of our knowledge, there is no solution in the relevant literature so far to address this.

E. Performance on All Data

In order to provide a comprehensive evaluation, the performance of DeepEgo+ is measured with the L1O method based on the entire dataset, consisting of 64 scenes. The evaluation results are presented in Table III. First, it is worth mentioning that Module A is trained with all radars together, while DeepEgo [22] is trained independently and must be optimally tuned for each radar. Nevertheless, Module A outperforms DeepEgo in all presented evaluation metrics. Also, the proposed method performs better in its smoothing mode than in its filtering mode. This is reasonable since, in the smoothing mode, both past information and future information of vehicle motion are available.

While Table III provides exact numbers for the estimation errors, it is difficult to see how these errors are distributed and their physical interpretation. Fig. 7 shows the histogram of all estimation errors accumulated across all 64 testing scenes. It is shown that the proposed method achieves notably less error than DeepEgo [22], especially when the vehicle is turning or driving at a high speed.

While the previous results provide a comprehensive study of the performance of the proposed method, it is important to acknowledge that the performance comparison is in part limited, given the following reasons. First, to the best of our knowledge, there are no other works in the open literature that studies instantaneous ego-motion estimation using unsynchronized radar sensor networks. Second, the original method DeepEgo in [22] had already shown better performance than several alternative works, so their performance is not listed in this article. Third, other fusion methods shown in Fig. 6 do not score as well as DeepEgo+ due to their limitations, which will be further discussed in Section IV-G.

F. Effect of Sensor Failure

As described in Section III, this study proposes a novel decentralized NN architecture for unsynchronized radar sensor fusion. It first computes the initial vehicle motion estimate based on each radar's output and then fuses these initial estimates based on the order of timestamps. Therefore, due to the decentralized processing structure, the proposed method should be able to work with radar sensor networks that are smaller than the one used for training. It is clear that this feature increases the system's robustness against sensor failure. Thus, it is interesting to simulate sensor failure scenarios and to investigate how model performance deteriorates as the size of the radar sensor network decreases during testing,

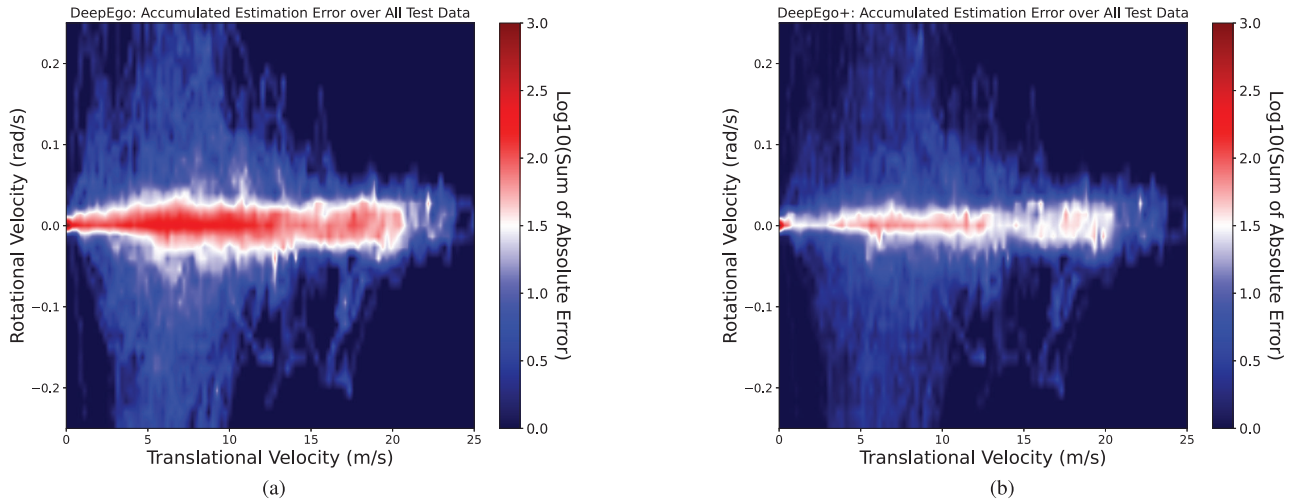


Fig. 7. Plots of the accumulated error from all 64 testing scenes. To plot this, the absolute errors between the ground truth ego motion and the model estimation are accumulated, and then, the base ten logarithm is applied for better visualization. (a) Result of the original DeepEgo [22]. (b) Result of the proposed method DeepEgo+.

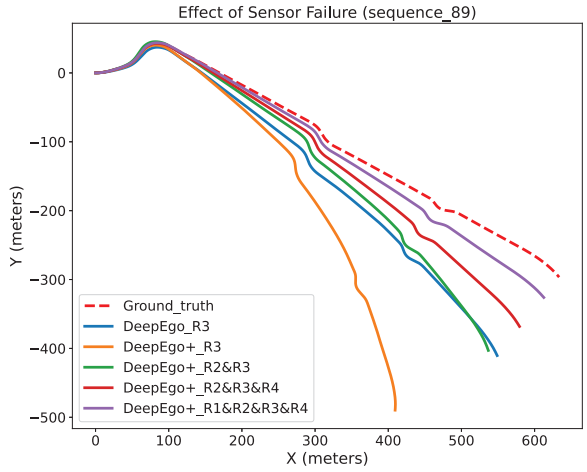


Fig. 8. Effect of radar sensor failure on the reconstructed vehicle trajectories. The proposed method DeepEgo+ is trained using all radars but tested with a varying number of missing/malfunctioning radars. The radar data used for testing do not contribute to model training and validation. The reconstructed trajectories are shown in solid lines, and the ground truth vehicle trajectory by the red dashed line.

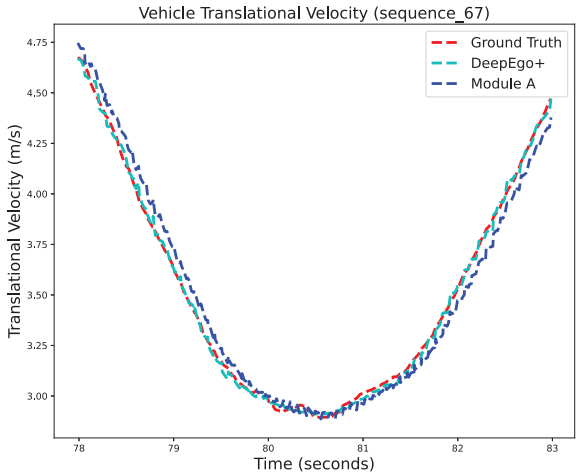
essentially assuming that data from some of the radars in the network are missing.

Fig. 8 presents an example of reconstructed vehicle trajectory, which visually illustrates the effect of sensor failure. In this test, after training on data from all radars, the model is tested using only data from the selected radar(s) with the assumption that some radars are malfunctioning, hence not providing any data. It can be seen that when all radars are working, the method provides the best estimation performance as the estimated trajectory approaches the ground truth trajectory. Even if half of the sensors fail, the proposed method still maintains good performance similar to the original DeepEgo [22]. However, it is noted that it is not recommended to intentionally apply the proposed architecture (DeepEgo+) to a too small radar sensor network compared to what was originally used for training. Although the proposed architecture provides robustness against sensor failure, it is best adapted to the size of the sensor network in the training data. As shown in the figure, when only one radar (e.g., "R_3") is working out of the original four used for training, the proposed method DeepEgo+ performs worse than DeepEgo [22].

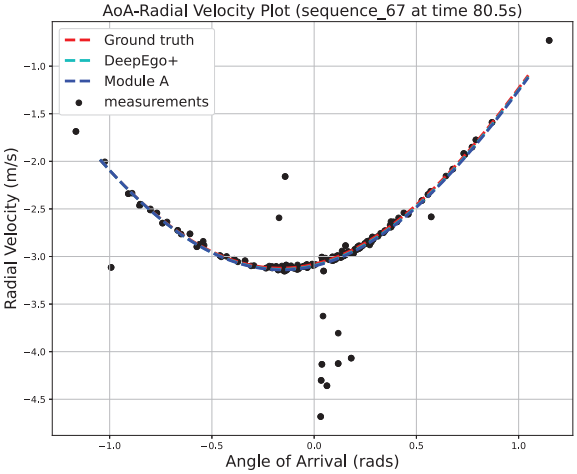
G. Vehicle Nonzero Acceleration

One of the key advantages of using radar for automotive applications is that it can directly provide radial velocity measurements of the detected objects. It is known that, for stationary objects, the measured radial velocities are related to aspect angles and the vehicle speed. However, if the speed of the vehicle is not constant, the Doppler frequency and the associated phase shift will vary with slow time (i.e., across radar chirps), and the estimated radial velocity will not match the vehicle speed, which can be instantaneously determined by the odometer sensor. Therefore, the vehicle speed can be overestimated or underestimated, depending on whether the vehicle is decelerating or accelerating. For better understanding, Fig. 9 shows the impact of vehicle nonzero acceleration on the estimated vehicle motion. It can be seen from Fig. 9(a) that the vehicle first decelerates, then maintains an almost constant speed, and finally accelerates. As expected, Module A overestimates the velocity when the vehicle decelerates, underestimates it when the vehicle accelerates, and overlaps with the ground truth only when the vehicle is at a constant speed. In contrast, the proposed DeepEgo+ can accurately estimate the velocity of the vehicle regardless of its motion state.

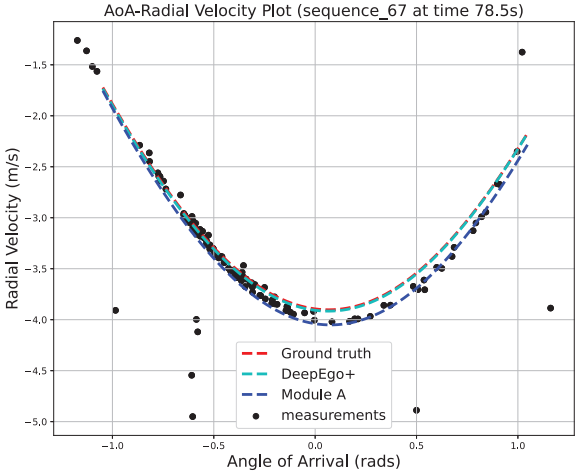
To understand the reasons for this, the images of Fig. 9(b) and (c) are helpful. They show two radar frames in AoA and radial velocity plots for the vehicle at constant speed and while decelerating, respectively. First, when the vehicle speed is constant, the radial velocities are not being overestimated or underestimated. As a result, the line representing the ground truth overlaps with the lines representing Module A and DeepEgo+, and they pass through detection points caused by stationary objects. As discussed in Section II, locating



(a) The vehicle's translational velocity over time.



(b) AoA-radial velocity plot (constant speed).



(c) AoA-radial velocity plot (decelerating).

Fig. 9. Impact of vehicle nonzero acceleration on ego-motion estimation. In this test, scene 67 ("sequence_67") of the RadarScenes dataset is used as test data. (a) Sequence of continuous vehicle motion in which the vehicle first slows down and then accelerates. (b) and (c) Relationship between AoA and radial velocity measurements at two selected timestamps; measurements are denoted as black dots. For all figures, the red, light-blue, and dark-blue dashed lines are computed based on the vehicle ground truth motion, the output of DeepEgo+, and the output of Module A, respectively.

the stationary points has been widely used by many radar-based ego-motion estimation methods, including the present work. However, as shown in Fig. 9(c), when the vehicle decelerates, the ground truth line does not match the measured radial velocity of the stationary object, and the radial velocity is underestimated. For methods that rely on accurate radial velocity measurements [22], [27], [30], this phenomenon can significantly affect their performance. For example, Module A and other instantaneous methods (mentioned in Section II-A) will never be able to estimate the true vehicle motion in Fig. 9(c) due to underestimation. In addition, it is clear that none of the filtering approaches evaluated in Fig. 6 is able to correct the estimates from Module A to the ground truth motion. However, since the magnitude of overestimation and underestimation is positively correlated with the magnitude of vehicle acceleration, DeepEgo+ is able to compensate for its effect by learning its patterns from consecutive radar frames using the proposed temporal NN within Module B. Note that the data used for evaluation in this section are not being used for model training and validation, which demonstrates the good generalization ability of DeepEgo+.

TABLE IV

NCC BETWEEN VEHICLE ACCELERATION AND MEASURED ERRORS IN VEHICLE TRANSLATIONAL VELOCITY ESTIMATION. NCC IS COMPUTED USING RADAR FRAMES FROM ALL 256 TESTING SCENES (INCL. FOUR RADARS). TO CONTROL FOR OTHER INDEPENDENT VARIABLES THAT MAY AFFECT THE MEASUREMENT ERROR, ONLY RADAR FRAMES WITH AN OUTLIER RATE LOWER THAN 40% WERE USED

Methods	RANSAC-based [27]	DeepEgo [22]	DeepEgo+
NCC	0.71	0.60	0.17

While the previous experiment has intuitively demonstrated the effectiveness of DeepEgo+ in addressing the impact of vehicle acceleration, Table IV presents a numerical relationship between velocity estimation error and vehicle acceleration

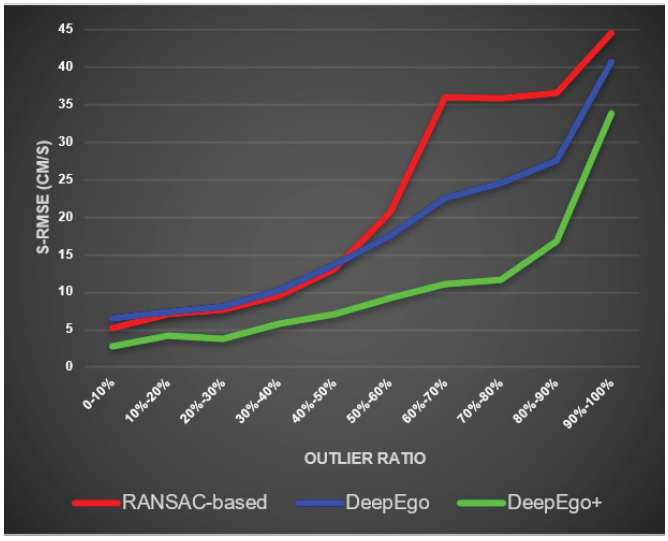


Fig. 10. Measured S-RMSE error in vehicle translational velocity estimation over different outlier ratios. All frames in the 64 test scenes are regrouped according to their outlier ratios. The red line represents the RANSAC-based instantaneous method [27], the previous approach DeepEgo [22] is represented by the blue line, and the proposed DeepEgo+ is denoted by the green line.

using the NCC. Based on the results, the RANSAC-based instantaneous method [27] and the previous approach DeepEgo [22] exhibit a strong correlation between acceleration and estimation error. Meanwhile, DeepEgo+ shows much weaker correlations than previous studies. Quantitatively, DeepEgo+ scores 3.5× better than DeepEgo.

Finally, it must be acknowledged that addressing the challenges induced by nonzero vehicle acceleration was not an explicit goal when developing the proposed approach. Also, to the best of our knowledge, none of the works in the literature mentioned in Section II have investigated this problem or proposed a solution. However, this is deemed to be an essential issue because, in realistic automotive scenarios, vehicles will have nonzero acceleration most of the times, especially in urban areas. Although the proposed method is very effective and addresses this challenge, the investigation can be continued in dedicated future work.

H. Robustness to Outliers

As discussed in Section II-A, one of the main challenges of instantaneous methods is to distinguish inliers from the input radar point cloud, while outliers caused by moving objects and false detections are ubiquitous in real-world driving. Fig. 10 compares the performance of DeepEgo+ with two previous studies in terms of the S-RMSE metric measured using different outlier ratios. First, it is reasonable that when the outlier ratio is less than 50%, that is, inliers are in the majority and it is difficult to make mistakes, the score of the RANSAC-based instantaneous method is similar to DeepEgo. Despite this, DeepEgo+ still outperforms both comparison methods. It is assumed that this improvement is due to the proposed temporal NNs (Module B), which offsets the effect caused by vehicle acceleration. Second, when the outlier ratio is between 50% and 90%, where estimation errors are likely to occur, the advantage of the NN-based instantaneous method becomes obvious. Still, DeepEgo+ has improved performances by nearly 50% compared to DeepEgo, demonstrating the importance of sensor fusion. Finally, when the outlier ratio exceeds 90%, the performance of all methods drops significantly because there is little useful information to be captured.

V. Conclusion

This article presents a novel solution for estimating the 2-D motion of a moving vehicle (ego motion) equipped with multiple unsynchronized automotive radar sensors. The proposed method, named DeepEgo+, consists of two NN-based components: Modules A and B. DeepEgo+ successfully addresses several limitations and challenges outlined in the relevant literature for ego-motion estimation. First, DeepEgo+ uses a decentralized signal processing architecture to provide the first solution for ego-motion estimation using unsynchronized radar networks. Specifically, the output of each radar is processed locally and independently by Module A without the need for good synchronization between sensors. Second, based on the previous work of DeepEgo [22], Module A has extensive upgrades in its original loss function and network architecture, improving its robustness and learning capabilities. Third, Module B uses temporal NNs to mimic the KF procedures and estimate the Kalman gain, thus bypassing the conventional, tedious manual tuning process. The late fusion approach applied in Module B further improves the system robustness to sensor failures and radar frames with high outlier ratios, where conventional methods fail. Fourth, DeepEgo+ addresses the challenges posed by vehicle nonzero acceleration to ego-motion estimation and proposes the first effective solution. To verify its performance, DeepEgo+ is tested using a challenging real-world radar dataset (RadarScenes) containing various driving scenarios on the road. The results demonstrate its superiority over a range of evaluation metrics compared to other methods in the literature. For future research, it would be interesting to compare the performance of early, halfway, and late fusion techniques for radar-based ego-motion estimation.

Acknowledgment

The authors would like to express their sincere gratitude to the anonymous reviewers for their valuable suggestions and constructive feedback throughout the review process. Their insightful and detailed comments greatly improved the clarity and quality of this article.

References

[1] S. Sun, A. P. Petropulu, and H. V. Poor, "MIMO radar for advanced driver-assistance systems and autonomous driving: Advantages and challenges," *IEEE Signal Process. Mag.*, vol. 37, no. 4, pp. 98–117, Jul. 2020.

[2] Z. Hong, Y. Petillot, A. Wallace, and S. Wang, "RadarSLAM: A robust simultaneous localization and mapping system for all weather conditions," *Int. J. Robot. Res.*, vol. 41, no. 5, pp. 519–542, Apr. 2022.

[3] P.-C. Kung, C.-C. Wang, and W.-C. Lin, "A normal distribution transform-based radar odometry designed for scanning and automotive radars," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 14417–14423.

[4] A. Palffy, E. Pool, S. Baratam, J. F. P. Kooij, and D. M. Gavrila, "Multi-class road user detection with 3+1D radar in the view-of-delft dataset," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 4961–4968, Apr. 2022.

[5] A. Ranga et al., "VRUNet: Multi-task learning model for intent prediction of vulnerable road users," 2020, *arXiv:2007.05397*.

[6] J. Borenstein and L. Feng, "Measurement and correction of systematic odometry errors in mobile robots," *IEEE Trans. Robot. Autom.*, vol. 12, no. 6, pp. 869–880, Dec. 1996.

[7] J. Yi, J. Zhang, D. Song, and S. Jayasuriya, "IMU-based localization and slip estimation for skid-steered mobile robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2007, pp. 2845–2850.

[8] Y. Gu, Y. Wada, L. Hsu, and S. Kamijo, "Vehicle self-localization in urban canyon using 3D map based GPS positioning and vehicle sensors," in *Proc. Int. Conf. Connected Vehicles Expo (ICCVE)*, Nov. 2014, pp. 792–798.

[9] Y. Lu, H. Ma, E. Smart, and H. Yu, "Real-time performance-focused localization techniques for autonomous vehicle: A review," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6082–6100, Jul. 2022.

[10] Q. Deng, H. Sun, F. Chen, Y. Shu, H. Wang, and Y. Ha, "An optimized FPGA-based real-time NDT for 3D-LiDAR localization in smart vehicles," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 9, pp. 3167–3171, Sep. 2021.

[11] P. Berthold, M. Michaelis, T. Luettel, D. Meissner, and H.-J. Wuensche, "Probabilistic vehicle tracking with sparse radar detection measurements," *J. Adv. Inf. Fusion*, vol. 17, no. 2, p. 116, 2022.

[12] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot., Sci. Syst. Conf.*, vol. 2, no. 9. Berkeley, CA, USA, Jul. 2014, pp. 1–9.

[13] M. Rapp, M. Barjenbruch, K. Dietmayer, M. Hahn, and J. Dickmann, "A fast probabilistic ego-motion estimation framework for radar," in *Proc. Eur. Conf. Mobile Robots (ECMR)*, Sep. 2015, pp. 1–6.

[14] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Signal Process. Mag.*, vol. 34, no. 2, pp. 22–35, Mar. 2017.

[15] V. Kubelka, E. Fritz, and M. Magnusson, "Do we need scan-matching in radar odometry?," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2024, pp. 13710–13716.

[16] V.-J. Štironja, L. Petrović, J. Peršić, I. Marković, and I. Petrović, "RAVE: A framework for radar ego-velocity estimation," in *Proc. IEEE Int. Conf. Multisensor Fusion Integr. Intell. Syst. (MFI)*, Sep. 2024, pp. 1–6.

[17] H. Kim, H. Jang, and A. Kim, "2D ego-motion with yaw estimation using only mmWave radars via two-way weighted ICP," 2024, *arXiv:2404.00830*.

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

[19] A. Galeote-Luque, V. Kubelka, M. Magnusson, J.-R. Ruiz-Sarmiento, and J. Gonzalez-Jimenez, "Doppler-only single-scan 3D vehicle odometry," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2024, pp. 13703–13709.

[20] F. Trombetta, A. Albaba, M. Bauduin, H. Sahli, and A. Bourdoux, "Radar-network-based odometry and elevation estimation," in *Proc. IEEE Radar Conf. (RadarConf)*, May 2024, pp. 1–6.

[21] S. Lovett, K. MacWilliams, S. Rajan, and C. Rossa, "Enhancing Doppler ego-motion estimation: A temporally weighted approach to RANSAC," in *Proc. IEEE Sensors Appl. Symp. (SAS)*, Jul. 2024, pp. 1–6.

[22] S. Zhu, A. Yarovoy, and F. Fioranelli, "DeepEgo: Deep instantaneous ego-motion estimation using automotive radar," *IEEE Trans. Radar Syst.*, vol. 1, pp. 166–180, 2023.

[23] P. R. M. de Araujo, A. Noureldin, and S. Givigi, "Toward land vehicle ego-velocity estimation using deep learning and automotive radars," *IEEE Trans. Radar Syst.*, vol. 2, pp. 460–470, 2024.

[24] P. K. Rai, N. Strokina, and R. Ghabcheloo, "4DEgo: Ego-velocity estimation from high-resolution radar data," *Frontiers Signal Process.*, vol. 3, Jun. 2023, Art. no. 1198205.

[25] C. X. Lu et al., "MilliEgo: Single-chip mmWave radar aided egomotion estimation via deep sensor fusion," in *Proc. 18th Conf. Embedded Netw. Sensor Syst.*, Nov. 2020, pp. 109–122.

[26] O. Schumann et al., "RadarScenes: A real-world radar point cloud data set for automotive applications," in *Proc. IEEE 24th Int. Conf. Inf. Fusion (FUSION)*, 2021, pp. 1–8.

[27] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous ego-motion estimation using Doppler radar," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 869–874.

[28] M. Pawłowski, A. Wróblewska, and S. Sysko-Romanczuk, "Effective techniques for multimodal data fusion: A comparative analysis," *Sensors*, vol. 23, no. 5, p. 2381, Feb. 2023.

[29] P. Biber and W. Strasser, "The normal distributions transform: A new approach to laser scan matching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Jun. 2003, pp. 2743–2748.

[30] M. Rapp, M. Barjenbruch, M. Hahn, J. Dickmann, and K. Dietmayer, "Probabilistic ego-motion estimation using multiple automotive radar sensors," *Robot. Auto. Syst.*, vol. 89, pp. 136–146, Mar. 2017.

[31] M. Heller, N. Petrov, and A. Yarovoy, "A novel approach to vehicle pose estimation using automotive radar," 2021, *arXiv:2107.09607*.

[32] M. Barjenbruch, D. Kellner, J. Klappstein, J. Dickmann, and K. Dietmayer, "Joint spatial- and Doppler-based ego-motion estimation for automotive radars," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2015, pp. 839–844.

[33] C. D. Monaco and S. N. Brennan, "RADARODO: Ego-motion estimation from Doppler and spatial data in RADAR images," *IEEE Trans. Intell. Vehicles*, vol. 5, no. 3, pp. 475–484, Sep. 2020.

[34] K. Haggag, S. Lange, T. Pfeifer, and P. Protzel, "A credible and robust approach to ego-motion estimation using an automotive radar," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 6020–6027, Jul. 2022.

[35] P. Meiresone, D. Van Hamme, W. Philips, and T. Verbelen, "Ego-motion estimation with a lowpower millimeterwave radar on a UAV," in *Proc. Int. Conf. Radar Syst. (RADAR)*, Oct. 2022, pp. 371–376.

[36] W. Li, R. Chen, Y. Wu, and H. Zhou, "Indoor positioning system using a single-chip millimeter wave radar," *IEEE Sensors J.*, vol. 23, no. 5, pp. 5232–5242, Mar. 2023.

[37] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous ego-motion estimation using multiple Doppler radars," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 1592–1597.

[38] J. Schlichenmaier, L. Yan, M. Stolz, and C. Waldschmidt, "Instantaneous actual motion estimation with a single high-resolution radar sensor," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Apr. 2018, pp. 1–4.

[39] K. Thormann and M. Baum, "Single-frame radar odometry incorporating bearing uncertainty," in *Proc. IEEE Symp. Sensor Data Fusion Int. Conf. Multisensor Fusion Integr. (SDF-MFI)*, Nov. 2023, pp. 1–7.

[40] M. Michaelis, P. Berthold, T. Luettel, and H.-J. Wuensche, "Generating odometry measurements from automotive radar Doppler measurements," in *Proc. IEEE Symp. Sensor Data Fusion Int. Conf. Multisensor Fusion Integr. (SDF-MFI)*, Nov. 2023, pp. 1–8.

[41] A. Galeote-Luque, V. Kubelka, M. Magnusson, J.-R. Ruiz-Sarmiento, and J. Gonzalez-Jimenez, "Doppler-only single-scan 3D vehicle odometry," 2023, *arXiv:2310.04113*.

[42] S. A. S. Mohamed, M. Haghbayan, T. Westerlund, J. Heikkonen, H. Tenhunen, and J. Plosila, "A survey on odometry for autonomous navigation systems," *IEEE Access*, vol. 7, pp. 97466–97486, 2019.

[43] D. Adolfsson, M. Magnusson, A. Alhashimi, A. J. Lilienthal, and H. Andreasson, "LiDAR-level localization with radar? The CFEAR approach to accurate, fast, and robust large-scale radar odometry in diverse environments," *IEEE Trans. Robot.*, vol. 39, no. 2, pp. 1476–1495, Apr. 2023.

[44] S. Lim, J. Jung, S. Kim, and S. Lee, "Radar-based ego-motion estimation of autonomous robot for simultaneous localization and mapping," *IEEE Sensors J.*, vol. 21, no. 19, pp. 21791–21797, Oct. 2021.

[45] T. Grebner, V. Janoudi, P. Schoeder, and C. Waldschmidt, "Self-calibration of a network of radar sensors for autonomous robots," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 59, no. 5, pp. 6771–6781, May 2023.

[46] C. Waldschmidt, J. Hasch, and W. Menzel, "Automotive radar—From first efforts to future systems," *IEEE J. Microw.*, vol. 1, no. 1, pp. 135–148, Jan. 2021.

[47] F. Hau, F. Baumgärtner, and M. Vossiek, "The degradation of automotive radar sensor signals caused by vehicle vibrations and other nonlinear movements," *Sensors*, vol. 20, no. 21, p. 6195, Oct. 2020.

[48] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, "Sensor and sensor fusion technology in autonomous vehicles: A review," *Sensors*, vol. 21, no. 6, p. 2140, Mar. 2021.

[49] H. Chen, Y. Liu, and Y. Cheng, "DRIO: Robust radar-inertial odometry in dynamic environments," *IEEE Robot. Autom. Lett.*, vol. 8, no. 9, pp. 5918–5925, Sep. 2023.

[50] S. Yang, M. Choi, S. Han, K.-H. Choi, and K.-S. Kim, "4D radar-camera sensor fusion for robust vehicle pose estimation in foggy environments," *IEEE Access*, vol. 12, pp. 16178–16188, 2024.

[51] Y. Zhuang, B. Wang, J. Huai, and M. Li, "4D iRIOM: 4D imaging radar inertial odometry and mapping," *IEEE Robot. Autom. Lett.*, vol. 8, no. 6, pp. 3246–3253, Jun. 2023.

[52] Y. Z. Ng, B. Choi, R. Tan, and L. Heng, "Continuous-time radar-inertial odometry for automotive radars," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 323–330.

[53] R. Yanase, D. Hirano, M. Aldibaja, K. Yoneda, and N. Suganuma, "LiDAR-and radar-based robust vehicle localization with confidence estimation of matching results," *Sensors*, vol. 22, no. 9, p. 3545, 2022.

[54] J. Michalczyk, R. Jung, and S. Weiß, "Tightly-coupled EKF-based radar-inertial odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 12336–12343.
[55] C. Doer and G. F. Trommer, "X-RIO: Radar inertial odometry with multiple radar sensors and yaw aiding," *Gyroscopy Navigat.*, vol. 12, no. 4, pp. 329–339, Dec. 2021.
[56] M. Hoffmann, L. Krabbe, C. Schüßler, P. Gulden, and M. Vossiek, "Instantaneous ego-motion estimation using a coherent radar network," in *Proc. 19th Eur. Radar Conf. (EuRAD)*, Sep. 2022, pp. 321–324.
[57] Z. Zeng, X. Dang, Y. Li, and X. Liang, "Multiview fusion automotive radar SLAM," *IEEE Sensors J.*, vol. 23, no. 19, pp. 23158–23168, Oct. 2023.
[58] H. Caesar et al., "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11621–11631.
[59] F. Engels, P. Heidenreich, M. Wintermantel, L. Stäcker, M. A. Kadi, and A. M. Zoubir, "Automotive radar signal processing: Research directions and practical challenges," *IEEE J. Sel. Topics Signal Process.*, vol. 15, no. 4, pp. 865–878, Jun. 2021.
[60] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. Amer. Soc. Mech. Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.
[61] G. Welch and G. Bishop, *An Introduction to the Kalman Filter*. Chapel Hill, NC, USA: Univ. North Carolina, 1995.
[62] P. J. Huber, "Robust estimation of a location parameter," in *Breakthroughs in Statistics: Methodology and Distribution*. Cham, Switzerland: Springer, 1992, pp. 492–518.
[63] O. Schumann et al., "Radarscenes: A real-world radar point cloud data set for automotive applications," in *Proc. IEEE Int. Conf. Inf. Fusion (FUSION)*, Nov. 2021, pp. 1–8.
[64] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 652–660.
[65] G. Revach, N. Shlezinger, X. Ni, A. López Escoriza, R. J. G. van Sloun, and Y. C. Eldar, "KalmanNet: Neural network aided Kalman filtering for partially known dynamics," *IEEE Trans. Signal Process.*, vol. 70, pp. 1532–1547, 2022.
[66] A. S. Rahmathullah, Á. F. García-Fernández, and L. Svensson, "Generalized optimal sub-pattern assignment metric," in *Proc. 20th Int. Conf. Inf. Fusion (Fusion)*, Jul. 2017, pp. 1–8.

Simin Zhu (Graduate Student Member, IEEE) received the B.Sc. degree in electrical engineering and automation from Central South University, Changsha, China, in 2016, and the master's degree from the Microwave Sensing, Signals and Systems (MS3) Group, Delft University of Technology (TU Delft), Delft, The Netherlands, in November 2021, specialized in radar signal processing and machine learning, where he is currently pursuing the Ph.D. degree.

He was a Hardware Engineer at Huawei Technology Company Ltd., Wuhan, China, for 1.5 years.

Satish Ravindran has around 12 years of experience in developing AI solutions for different industries such as autonomous driving, intelligent traffic sensing (ITS), and the Internet of Things (IoT). He worked on a wide spectrum of applications in AI including NLP, computer vision, and radar processing. He joined NXP Semiconductors, San Jose, CA, USA, in 2018, where he is currently an AI Technical Lead of radar innovations working at the NXP Research and Development Division. He has led the development of a comprehensive portfolio of AI applications at all stages of the radar processing chain, from signal processing to perception. He is also helping in the definition of the next generation of NXP SoCs and has multiple patents and papers published in radar signal processing and AI solutions.

Lihui Chen received the B.Sc. degree in applied mathematics from Sun Yat-sen University, Guangzhou, China, in 1999, and the Ph.D. degree in electrical and electronic engineering from Loughborough University, Loughborough, U.K., in 2006.

He has experience in machine learning for 18 years, with 12 years focusing on research of automotive ADAS products, including surround vision, driver monitoring, and HD mapping. He has authored and co-authored 13 patents, with five systems delivered for production. In 2022, he joined the NXP's Research and Development Division, San Jose, CA, USA, as the Radar Perception Project Lead. His research interests include AI solutions for perceptions with different sensors, and neutral network optimization for edge devices.

Alexander G. Yarovoy (Fellow, IEEE) received the Diploma degree (Hons.) in radiophysics and electronics and the Candidate Phys. and Math. Sci. and Doctor Phys. and Math. Sci. degrees in radiophysics from Kharkiv State University, Kharkiv, Ukraine, in 1984, 1987, and 1994, respectively.

In 1987, he joined the Department of Radiophysics, Kharkiv State University, Kharkiv, as a Researcher, where he became a Full Professor in 1997. From September 1994 to 1996, he was with the Technical University of Ilmenau, Ilmenau, Germany, as a Visiting Researcher. Since 1999, he has been with Delft University of Technology, Delft, The Netherlands. Since 2009, has been the Chair of Microwave Sensing, Systems and Signals. He has authored and co-authored more than 600 scientific or technical articles, 11 patents, and 14 book chapters. His main research interests are in high-resolution radar, microwave imaging, and applied electromagnetics (in particular, UWB antennas).

Prof. Yarovoy was a recipient of European Microwave Week Radar Award for the paper Best Advances the State-of-the-Art in Radar Technology in 2001 (together with L. P. Ligthart and P. van Genderen) and 2012 (together with T. Savelyev). In 2023 together with Dr. I. Ullmann, N. Kruse, R. Gündel, and Dr. F. Fioranelli, he got the Best Paper Award at the IEEE Sensor Conference. In 2010 together with D. Caratelli, he got the Best Paper Award of the Applied Computational Electromagnetic Society (ACES). He served as the General TPC Chair for the 2020 European Microwave Week (EuMW'20), the Chair and the TPC Chair for the 5th European Radar Conference (EuRAD'08), and the Secretary for the 1st European Radar Conference (EuRAD'04). He also served as the Co-Chair and the TPC Chair for the Xth International Conference on GPR (GPR2004). From 2008 to 2017, he served as the Director for European Microwave Association (EuMA). He serves on various editorial boards, such as IEEE TRANSACTIONS ON RADAR SYSTEMS. From 2011 to 2018, he served as an Associate Editor for *International Journal of Microwave and Wireless Technologies*. He has been a member of numerous conference steering and technical program committees.

Francesco Fioranelli (Senior Member, IEEE) received the Ph.D. degree from Durham University, Durham, U.K., in 2014.

He was an Assistant Professor with the University of Glasgow, Glasgow, U.K., from 2016 to 2019, and a Research Associate at University College London, London, U.K., from 2014 to 2016. He is currently an Associate Professor at TU Delft, Delft, The Netherlands. His research interests include the development of radar systems and automatic classification for human signatures analysis in healthcare and security, drones and UAVs' detection and classification, and automotive radar. He has authored or co-authored over 190 peer-reviewed publications and edited the books titled *Micro-Doppler Radar and Its Applications* and *Radar Countermeasures for Unmanned Aerial Vehicles* (IET-Scitech, 2020).

Dr. Fioranelli received four best paper awards and the IEEE AESS Fred Nathanson Memorial Radar Award in 2024.