NN-Based Instantaneous Target Velocity Estimation Using Automotive Radar

Hassan, Mujtaba; Fioranelli, Francesco; Yarovoy, Alexander; Ravindran, Satish; Chen, Luihi

DOI
10.1109/JSEN.2025.3576437

Licence
Dutch Copyright Act (Article 25fa)

Publication date
2025

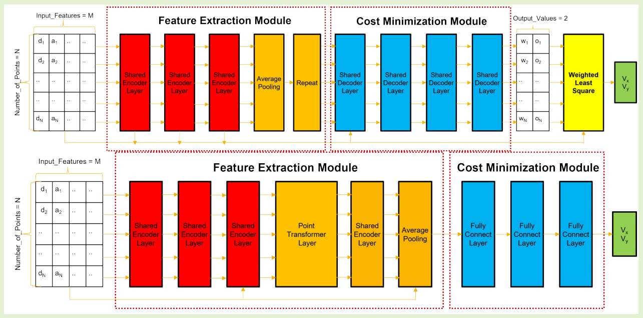
Document Version
Final published version

Published in
IEEE Sensors Journal

NN-Based Instantaneous Target Velocity Estimation Using Automotive Radar

Mujtaba Hassan, *Graduate Student Member, IEEE*, Francesco Fioranelli, *Senior Member, IEEE*, Alexander Yarovoy, *Fellow, IEEE*, Satish Ravindran, and Luihi Chen

***Abstract*—The problem of estimating instantaneous distributed target velocity using noisy measurements by multiple asynchronous automotive radar sensors is investigated. Two novel neural networks (NNs)-based approaches are proposed to address the problem. Both NNs use the point cloud with radar detections as an input. In the first approach, a hybrid NN is designed to take a set of points inside a cluster as its input, extract spatial–dynamic features to be used as weights for each input point, and apply them to obtain a weighted least square (WLS) solution for instantaneous velocity estimation. To this end, dedicated loss functions are proposed to allow the model to predict weights that can follow a velocity profile curve satisfying target constraints. Moreover, a small offset in the radial velocity value of each point is applied to adjust errors in the sensor measurements. In the second approach, a deep NN is proposed that takes a set of points inside a cluster as its input and directly outputs velocity estimates. Both approaches have been verified experimentally using the large open-source automotive RadarScenes dataset. The results show a significant improvement in terms of mean absolute error in velocity estimation over the state-of-the-art alternative techniques. Moreover, the estimated velocity is used as an additional measurement value inside a target tracker. Results show that this can increase the performance of the tracker, especially during challenging scenarios such as abrupt changes in the velocity of the target.**



***Index Terms*— Automotive radar, instantaneous velocity estimation, neural networks (NNs).**

I. Introduction

ADVANCED driver assistance systems (ADAS) have attracted a lot of attention in the automotive domain [1], and many ADAS features are key in determining the safety score of vehicles as part of the Euro NCAP requirements [2]. These features can be broadly categorized into two categories: perception stack and decision-making [3]. The former provides higher degrees of scene understanding, whereas capabilities of the latter allow the vehicle to make relevant decisions autonomously. Multiobject tracking (MOT) is a vital component of this perception stack to enable the vehicle to recognize objects present in the scene and estimate their characteristics.

For this task, different sensors can be used, each having some advantages over the others. Cameras are widely used since they are relatively cheap and can give a very detailed and intuitive representation of the scene. However, they do not provide a direct estimation of the distance and velocity of the objects, and they can fail in cases such as low light, snow, and fog. LiDAR can provide accurate information about the distance of the objects and its characteristics. However, it is an extremely costly sensor which makes it difficult to deploy for mass production, as well as not estimating directly the velocity of objects. An alternative to LiDAR is radar, currently used in ADAS systems, thanks to its relatively low costs, capabilities in low visibility conditions, and direct estimation of distance and radial velocity of objects via the Doppler effect.

For many ADAS functions, the knowledge of the velocity of objects in the scene is indeed crucial [4], and methods that can provide an instantaneous (i.e., with a single data

Received 29 April 2025; accepted 1 June 2025. Date of publication 13 June 2025; date of current version 1 August 2025. This result is part of the Important Project of Common European Interest Microelectronics/Communication Technologies (IPCEI ME/CT) and is funded in part by the European Union Next Generation EU, in part by the German Federal Ministry for Economic Affairs and Energy, in part by the Bavarian Ministry of Economic Affairs, Regional Development and Energy, and in part by the Free State of Saxony with the help of tax revenue based on the budget approved by the Saxon State parliament and the Free and Hanseatic City of Hamburg. The associate editor coordinating the review of this article and approving it for publication was Dr. Dipanwita Thakur. *(Corresponding author: Mujtaba Hassan.)*

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

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

Digital Object Identifier 10.1109/JSEN.2025.3576437

frame) estimation have been investigated in the literature on automotive radar. The major challenge here is that due to its high range resolution, a typical target (car, cyclist, pedestrian, etc.) is seen by an automotive radar as a point cloud. This cloud is very dynamic and changes considerably by changing the target aspect angle. Therefore, traditional methods for target tracking by low-resolution radars are not directly applicable to automotive radars.

A method to estimate the instantaneous velocity of targets, based on the mathematical relationship between their radial velocity and azimuth, was proposed in [5]. In [6], this approach was augmented with an RANSAC-based [7]removal of outlier points, and is widely used by other methods. Although the approach helps in reducing the overall error by removing some noisy points, this is a heuristic process which makes it difficult to remove outlier points from inlier points in a generalized and reliable manner. This approach was further extended in [8] and [9] to include nonlinear velocity estimation using dual radar. Separation of the linear and nonlinear velocities by using a single radar was proposed in [10]. A recursive least square solution for the same task was used in [11]. Orientation information to assist in velocity estimation was used in [12]. Direct computation of the full motion states by using multiple sensors was suggested in [13]. However, all these methods are based on heuristics that tend not to use all the features available in the radar point cloud and can fail in challenging cases with a high number of outliers. Moreover, the same weight, importance is given to each point, which increases the error because the different noise level in each point may cause the radial velocity–azimuth profile to shift.

Alternative methods found in the literature perform the same function using neural networks (NNs). Svenningsson et al. [14] trained a supervised NN to obtain object detections with velocity values, assuming the velocity to be in the direction of orientation. However, the orientation estimation from a single frame is inaccurate and can lead to high errors in velocity estimation. Niederlöhner et al. [15] used a self-supervised learning method to train an NN for object detection with velocity labels. Being more robust to outliers, this approach used multiple frames for this task, which breaks the requirement of fast, instantaneous target velocity estimation.

While many challenges have been addressed in previous studies, several gaps still remain. Traditional methods for instantaneous velocity estimation, often derived from [6], typically involve removing outlier points from detected targets using the RANSAC [7] algorithm. This heuristic approach can fail in certain scenarios, particularly when the number of outliers exceeds the number of inliers. Additionally, these methods primarily use radial velocity and azimuth features, as incorporating other features like shape, range, and RCS is difficult with heuristic techniques. Also, each point is given the same weight, which increases velocity estimation errors due to varying noise levels, causing shifts in the radial velocity–azimuth profile curve. Thus, a method that can target these shortcomings may improve the performance.

Moreover, the use of NNs for instantaneous velocity estimation is still underdeveloped. Most existing works are inspired by LiDAR and do not fully exploit the unique features of automotive radars and their data, focusing mainly on object detection with velocity estimation as a secondary outcome [14]. A dedicated NN for this purpose may therefore be useful.

Furthermore, it should be noted that some methods suggest improving velocity estimation through the usage of multiple frames [15] or by enhancing tracking [9], but this introduces latency, which may be undesirable for real-time automotive applications. Therefore those approaches are not considered in this work.

To address these gaps, the objective of this study is to develop an instantaneous velocity estimation method using automotive radar that can perform well for real driving scenarios. In this article, two NN-based solutions for instantaneous target velocity estimation are proposed. In the first approach, a hybrid NN-based WLS solution is designed to operate on an input point cloud, inspired from the ego-motion estimation task in [16]. Essentially, an NN is used to predict the weights of each input point, instead of applying RANSAC [7] to separate these points as done by [6]. NN is utilized because it can learn more complex features between points to obtain the weights. These weights are then used for a WLS solution. Unlike the ego-motion estimation task in [16], where a large number of points are available from static objects in the scene, for the problem considered in this article there are only a few detection points available from the detected targets. This makes the results of the velocity estimation highly susceptible to noise. In order to overcome this problem, implicit characteristics of automotive target motion are learned based on the usage of suitable loss functions that trains the proposed network. This guides the network to obtain a radial velocity–azimuth profile curve which can provide realistic velocity output expected in an automotive context. Finally, a small offset can be added to each point to reduce the error in cases when it is not possible to estimate the velocity based on only noisy measurements. Unlike in [16], a loss is specifically given the value of the magnitude of these offsets to reduce overfitting on training data and minimize offset values for points with low noise.

In the second approach, a deep NN (DNN) is specifically designed to directly estimate the target velocity using target point cloud. The motivation for this second approach is that there is a fundamental limit to the velocity estimation methods using radial velocity–azimuth profiles, which starts to deteriorate with a low number of detected points per target. This is not an uncommon situation in automotive radar. The idea of this second approach is to improve the velocity estimation on these cases by learning the output in an end-to-end fashion, directly based on training data.

The proposed approaches are extensively evaluated using the open-source experimental RadarScenes dataset [17], and show a superior performance than alternative methods. Specifically, a reduction in the mean average error (MAE) of 59% can be achieved using the proposed DNN method as compared to the network presented in [16], which was found to be the best alternative method. Moreover, it is shown that the addition of velocity as a measurement within a tracking algorithm improves its performance, especially for cases with changing dynamics of tracked objects. Specifically, a reduction in average RMSE of 41% on the tracker's velocity estimation can be achieved by adding the velocity estimated from the proposed DNN method as a measurement in the tracker. Summarizing, the main contributions of this work include the following.

1) Design of two novel NN-based methods to obtain instantaneous target velocity estimates from a set of target cloud points, which outperform alternative methods for this task.
2) Integration of the instantaneous velocity estimation as an additional measurement within tracking algorithms that improves the velocity estimation from the tracker itself.
3) Verification of the performance on real driving scenarios using the large automotive radar RadarScenes dataset, which shows the applicability of the proposed methods on various scenarios including changing dynamics of tracked objects.

The remainder of this article is organized as follows. Section II provides an overview of the methods found in literature. Section III describes the details of the two proposed methods. Section IV presents the experiments and a comprehensive evaluation of the results. Finally, Section V draws conclusions.

II. Related Work

This section discusses existing approaches available for instantaneous velocity estimation using automotive radar. These can be broadly categorized into two main approaches: traditional methods and NN-based methods.

A. Traditional Approaches

Instantaneous velocity estimation using automotive radar is a well-studied problem. Rohling et al. [5] are the pioneers in this regard who estimated the instantaneous velocity of targets, based on the mathematical relationship between their radial velocity and azimuth. However, noise points were not removed during the calculation resulting in a decrease in accuracy. Kellner et al. [6] targeted this shortcoming by introducing an RANSAC-based [7] removal of outlier points. This is a popular approach and used by many later works [8], [9], [13]. Although this approach helps in reducing the overall error by removing noise points, this is a heuristic process which makes it difficult to remove outlier points from inlier points in a generalized and reliable manner. This approach was further extended in [8] which used dual radars to obtain rotational velocity estimation. However, this resulted in higher errors for targets with lower rotational velocity. Kellner et al. [9] integrated the radial velocity–azimuth equation inside a tracking algorithm to improve tracking. Schlichenmaier et al. [10] proposed to estimate the linear and rotational velocities by using a single automotive radar. Ru and Xu [12] proposed to include orientation information to assist in velocity estimation. A recursive least square solution for the same task was developed in [11]. Schlichenmaier et al. [13] suggested direct computation of the full motion states by using multiple sensors. This showed improved performance on targets with lower rotational velocity than [8]. However, all these methods are based on heuristics that tend not to use all the features available in the radar point cloud and can fail in challenging cases with a high number of outliers. Moreover, the same weight, importance is given to each point, which increases the error because the different noise level in each point may cause the radial velocity–azimuth profile to shift. Table I summarizes the merits and limitations for these methods.

B. NN-Based Approaches

Alternative methods found in the literature perform the same function using neural networks (NN). A supervised NN to obtain object detections with velocity values, assuming the velocity to be in the direction of orientation, is proposed in [14]. However, the orientation estimation from a single frame is inaccurate and can lead to high errors in velocity estimation. A self-supervised learning method to train an NN for object detection with velocity labels is proposed in [15]. Being more robust to outliers, this approach used multiple frames for this task, which breaks the requirement of fast, instantaneous target velocity estimation. A WLS solution for ego-motion estimation whereby the weights were obtained using an NN is proposed in [16]. A similar approach can be used to obtain instantaneous velocity estimation of targets. However since both tasks are different, there is a need to develop this further to obtain accurate target instantaneous velocity estimation. Table II summarizes the merits and limitations for these methods.

III. Proposed Methodology

This section introduces the design of the proposed NN-based methods for instantaneous velocity estimation.

A. Problem Formulation

This article focuses on the instantaneous velocity estimation of targets using automotive radar point cloud. Since points at different regions of a target can be detected, the radial velocity components for each point are different based on azimuth. Moreover, usage of multiple radar sensors increases the azimuth span of the target and provides more detections with different values for radial velocity and azimuth. These factors are useful for velocity estimation which is found using the radial velocity–azimuth relationship [5] as given by the following equation:

$$\begin{bmatrix} v_(r,1) \\ v_(r,2) \\ \vdots \\ v_(r,N) \end{bmatrix} = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) \\ \cos(\theta_2) & \sin(\theta_2) \\ \vdots & \vdots \\ \cos(\theta_N) & \sin(\theta_N) \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} \tag{1}$$

where N denotes the number of point detections in the cloud, $v_{r,N}$ denotes the radial velocity feature of point N, θ_N denotes the azimuth feature of point N, V_x denotes the down-range target velocity, and V_y denotes the cross-range target velocity.

To write (1) in a more compact form, the vector-based formulation given by (2) is used

$$D = A \cdot V \tag{2}$$

where A denotes the velocity projection matrix, D denotes the radial velocity measurements, and V denotes the velocity

TABLE I
COMPARISON BETWEEN CONVENTIONAL METHODS IN THE LITERATURE FOR INSTANTANEOUS VELOCITY ESTIMATION USING AUTOMOTIVE RADAR

Method	***Merits***	***Limitations***
Lateral velocity estimation for automotive radar applications [5] (published: 2007)	Introduced a least square solution for linear instantaneous velocity estimation based on the velocity profile concept.	Does not remove outlier points for estimating the least square solution. Same weight is given to each point for the calculation.
Instantaneous lateral velocity estimation of a vehicle using Doppler radar [6] (published: 2013)	Proposed a RANSAC based removal of outlier points and performed least square solution on the remaining inlier points.	Heuristic approach to remove the outliers which cannot be easily generalized. Same weight is given to each point for the calculation.
Instantaneous full-motion estimation of arbitrary objects using dual Doppler radar [8] (published: 2014)	Proposed usage of multiple radars to estimate full motion parameters including turn rate.	Used RANSAC to remove the outlier points which have the same issues mentioned earlier. Gives high error for zero turn rate.
Tracking of extended objects with high resolution Doppler radar [9] (published: 2016)	Integrated the velocity profile into a tracking algorithm to improve the performance of tracking on highly dynamic maneuvers.	Used RANSAC to remove the outlier points which have the same issues mentioned above. Not an instantaneous method.
Improvement on velocity estimation of an extended object [12] (published: 2017)	Provide a robust velocity estimation by fusing three velocity estimation methods (based on heading angle, point tracking, velocity profile).	The proposed method for heading angle estimation is not always accurate because of the nature of automotive radar detections. Not an instantaneous method.
Instantaneous actual motion estimation with a single high-resolution radar sensor [10] (published: 2018)	Proposed a method to estimate instantaneous velocity using single radar by distinguishing between linear / nonlinear motion.	Used RANSAC to remove the outlier points which have the same issues mentioned above. Does not consider target orientation for velocity estimation.
An RLS-based instantaneous velocity estimator for extended radar tracking [11] (published: 2020)	Proposed a Recursive Least Square (RLS) based approach to reduce the impact of noise for instantaneous velocity estimation.	Requires a good initial estimate of the velocity for the method to converge to an accurate solution. Does not consider target orientation for velocity estimation.
Clustering and subsequent contour and motion estimation of automotive objects using a network of cooperative radar sensors [13] (published: 2023)	Proposed a method to estimate full instantaneous motion state that shows better performance than [8] on cases with zero turn rate using multiple sensors.	Used RANSAC to remove the outlier points which have the same issues mentioned above. Does not consider target orientation for velocity estimation.

TABLE II
COMPARISON BETWEEN NN-BASED METHODS IN THE LITERATURE FOR INSTANTANEOUS VELOCITY ESTIMATION USING AUTOMOTIVE RADAR

Method	***Merits***	***Limitations***
Radar-pointgnn: Graph based object recognition for unstructured radar point-cloud data [14] (published: 2021)	Proposed a data driven graph based method to perform object detection with velocity estimation.	The velocity direction is dependent only on orientation but the orientation estimation can be inaccurate.
Self-supervised velocity estimation for automotive radar object detection networks [15] (published: 2022).	Proposed a self-supervised data driven method for velocity estimation which does not require ground truth velocity.	The method generates velocity labels based on two consecutive frames which may be inaccurate during highly changing target maneuvers. Uses multiple data frames.
DeepEgo: Deep instantaneous ego-motion estimation using automotive radar [16] (published: 2023)	Proposed a weighted least square solution for instantaneous ego motion estimation that uses data driven approach to learn weights.	Targeted for ego motion estimation. Does not consider target motion characteristics. Offset usage can cause overfitting.

components. Equation (2) can be used to estimate V given at least two detections. Typically, standard regression approaches such as an ordinary least square (OLS) method can be used to solve this problem [5], as given by the following equation:

$$V^{\text{est}} = \left(A^T A\right)^{-1} A^T D. \tag{3}$$

However, real-world data may contain noise in measurements and outlier points due to incorrect clustering of points. Moreover, moving parts of targets such as wheels show a different radial velocity distribution than the true velocity of the target. Being susceptible to these factors, OLS will provide an inaccurate solution. To address this issue, in [6] it was proposed to use RANSAC [7] to remove the outlier points. Also, since both the radial velocity and azimuth measurements can contain noise, [6] used orthogonal distance regression (ODR) instead of least square to solve the aforementioned equations. Although this helps in reducing errors by removing some outlier points, this remains a heuristic approach that can fail in some particular cases. Moreover, same weight is given to each considered point, which increases the error in velocity estimation because the different noise levels in each point cause the radial velocity–azimuth profile curve to shift. As an illustration, a simple simulation is performed where a target is represented by seven detected points. A random noise of different magnitude was added to each of these detected points. This addition was performed to simulate measurements containing both outliers and inliers with different noise levels. Fig. 1 shows the radial velocity–azimuth profile curve and the results obtained by applying an RANSAC-based solution. Three issues can be observed: first, the approach fails to determine some outlier points; second, similar weight is given to each of the inlier points with different noise levels; third, some inlier points are considered as outliers. Because of these issues, the RANSAC-based method resulted in an inaccurate velocity estimation. To obtain a more accurate solution, different points should be given different weights based on their noise levels.

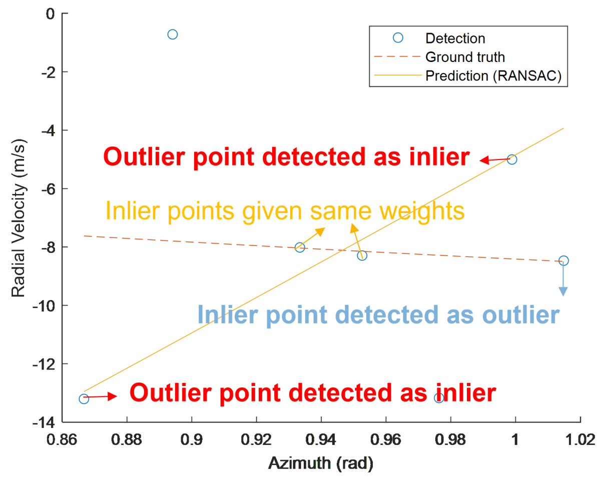


Fig. 1. Example of RANSAC-based velocity estimation for a simulated scene with some salient cases: two outlier points are detected as inlier as shown by red color; all the inlier points are given the same weights; one inlier point is detected as an outlier as shown by blue color. These factors may result in an inaccurate velocity estimation.

To address this issue, an NN-based solution is proposed in the first approach to learn these weights. The idea is that outlier points which do not originate from the target exhibit features that are different from those of points belonging to the target; hence, an NN can learn to distinguish these points from each other and assign weights accordingly.

B. Approach 1: NN-Based WLS Solution for Velocity Estimation

The first approach to estimate instantaneous target velocity is based on a hybrid NN architecture inspired from DeepEgo [16] and PointNet [18] in the literature. Fig. 2 shows the architecture of the proposed method. The input to the network is points detected from the same target, which can be obtained by a tracker using a clustering algorithm.

Essentially, this architecture comprises two parts. First is the feature extraction module. Here, each of the input point is passed through a shared multilayer perceptron (MLP) encoder block containing three shared perceptron layers to extract local features from each of these points. Multiple layers are used to extract features with different levels of detail. These local features are then averaged using an average pooling layer to obtain a combined set of global features. These global features are then replicated for each point and passed to the second part. This part is the cost minimization module. It contains MLP decoder block to serve as a cost function that is minimized to obtain the weights. Essentially, it takes the detected points, local features, and global feature as the input, concatenates these features, and then uses MLP decoder block to approximate a cost function that is minimized to obtain the desired weights. These weights can finally be used for a WLS solution. The final layer is a single perceptron layer that outputs two values for each point: weight and offset. The weight denotes the contribution of each point in determining the WLS solution. The offset provides an approximation of the noise in each point and is added to the radial velocity values to try and correct this noise. The NN is trained for this task through the usage of suitable loss functions to guide the training, detailed as follows.

1) Loss Functions:

The proposed network is trained to perform its task through the usage of five loss functions. The first two loss functions are inspired from [16], whereas the last three functions are novel and proposed in this work.

a) Motion loss: This is the loss between the true and predicted velocity which gives an indication of the closeness between these values, and is therefore the main loss function. Unlike [16], Huber loss [19] instead of mean square error (MSE) loss is used to further mitigate the impact of outliers during training. This is beneficial because for instantaneous velocity estimation, each target contains few points resulting in high impact of noise compared to the task of ego-velocity estimation. Huber loss is generally more robust to these cases than the MSE loss. The motion loss function is expressed by

$$\mathrm{Loss}_{\mathrm{Mot}} = \begin{cases} 1/2\left(V^{\mathrm{gt}} - V^{\mathrm{est}}\right)^2, & \text{if } |V^{\mathrm{gt}} - V^{\mathrm{est}}| < a \\ a\left(|V^{\mathrm{gt}} - V^{\mathrm{est}}| - 1/2a\right), & \text{if } |V^{\mathrm{gt}} - V^{\mathrm{est}}| \geq a \end{cases} \tag{4}$$

where a denotes a given threshold, V^{gt} denotes the ground truth velocity, and V^{est} denotes the estimated velocity.

b) Doppler loss: Although motion loss minimizes the difference between true and predicted velocities, it does not consider the weights of each point and works mainly as a minimization loss for a least square solution. This results in overfitting to a few points and many inlier points are simply ignored. To address this issue, [16] introduced a Doppler loss which uses the difference between expected and measured radial velocities to determine the correct weights for each points. This loss can be obtained by the following set of equations:

$$D^{\mathrm{exp}} = A \cdot V^{\mathrm{gt}} \tag{5}$$

$$D^{\mathrm{err}} = D^{\mathrm{exp}} - D^{\mathrm{meas}} \tag{6}$$

$$W^{\mathrm{gt}} = \exp\left(-\frac{(D^{\mathrm{err}} - 0)^2}{2\sigma^2}\right) \tag{7}$$

$$\mathrm{Loss}_{\mathrm{Dopp}} = \begin{cases} 1/2\left(W^{\mathrm{gt}} - W^{\mathrm{est}}\right)^2, & \text{if } |W^{\mathrm{gt}} - W^{\mathrm{est}}| < a \\ a\left(|W^{\mathrm{gt}} - W^{\mathrm{est}}| - 1/2a\right), & \text{if } |W^{\mathrm{gt}} - W^{\mathrm{est}}| \geq a \end{cases} \tag{8}$$

where σ^2 denotes the variance as a hyperparameter, D^{meas} denotes the measured radial velocity, W^{gt} denotes the expected weights from ground truth, and W^{est} denotes the estimated weights from the model.

c) Gradient loss: Although motion loss and Doppler loss minimize the difference between true and predicted velocities to obtain a WLS solution, the results are susceptible to noise especially in some challenging situations. This is because the span of azimuth values covered by a single target is usually very small with respect to the radar field of view. As a result, the noise can have a huge impact on the shape of the radial velocity–azimuth profile curve for a detected target. This is strengthened by the fact that, in the regions where there is a high gradient between radial velocity and azimuth, a small change in the radial velocity can have a huge impact on the

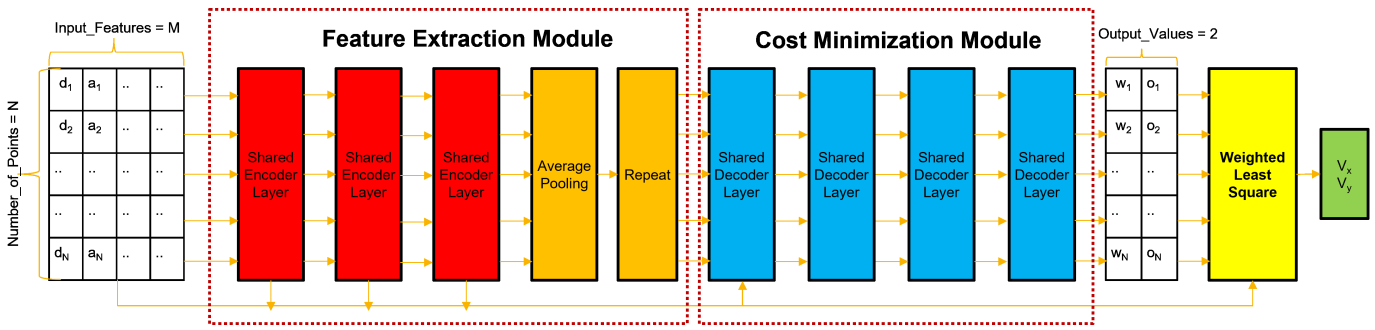


Fig. 2. Architecture for proposed Approach 1: It consists of a feature extraction module to extract pointwise features which are then passed to a cost minimization module to obtain pointwise weights and offsets. These pointwise weights and offsets are used in a WLS solution to obtain velocity.

shape of radial velocity–azimuth profile curve. As a result, it is necessary that the network predicts weights which can generate gradients of radial velocity–azimuth profile curve that closely follows the ground truth. To achieve this objective, a gradient loss is proposed here to limit the difference in gradient between the ground truth radial velocity–azimuth profile and the obtained predicted one. The loss is computed by the following equations:

$$\mathrm{Grad}^{\mathrm{ex}} = -{V_x}^{\mathrm{gt}} * \sin\theta + \ {V_y}^{\mathrm{gt}} * \cos\theta \tag{9}$$

$$\mathrm{Grad}^{\mathrm{est}} = -{V_x}^{\mathrm{est}} * \sin\theta + \ {V_y}^{\mathrm{est}} * \cos\theta \tag{10}$$

$$\mathrm{Grad}^{\mathrm{err}} = \mathrm{Grad}^{\mathrm{exp}} - \mathrm{Grad}^{\mathrm{est}} \tag{11}$$

$$W^{\mathrm{gt}} = \exp\left(-\frac{(\mathrm{Grad}^{\mathrm{err}} - 0)^2}{2\sigma^2}\right) \tag{12}$$

$$\mathrm{Loss}_{\mathrm{Grad}} = \begin{cases} 1/2\left(W^{\mathrm{gt}} - W^{\mathrm{est}}\right)^2, & \text{if } |W^{\mathrm{gt}} - W^{\mathrm{est}}| < a \\ a\left(|W^{\mathrm{gt}} - W^{\mathrm{est}}| - 1/2a\right), & \text{if } |W^{\mathrm{gt}} - W^{\mathrm{est}}| \geq a \end{cases} \tag{13}$$

where ${V_x}^{\mathrm{gt}}$ denotes the down-range ground truth velocity, ${V_y}^{\mathrm{gt}}$ denotes the cross-range ground truth velocity, ${V_x}^{\mathrm{est}}$ denotes the down-range estimated velocity, and ${V_y}^{\mathrm{est}}$ denotes the cross-range estimated velocity.

d) Orientation loss: Motion, Doppler, and gradient losses are more generic loss functions that do not consider the target properties. For instance, in an automotive context a vehicle may only be traveling in the direction of its orientation. This can provide additional constraints for the network to generate appropriate weights. To incorporate this information into the network, an orientation loss is proposed to guide the network to output weights generating a velocity estimate which is closer to the direction of target orientation. The loss function is given by the following equations:

$$\mathrm{Ori}^{\mathrm{gt}} = \tan^{-1}\left(\frac{{V_y}^{\mathrm{gt}}}{{V_x}^{\mathrm{gt}}}\right) \tag{14}$$

$$\mathrm{Ori}^{\mathrm{est}} = \tan^{-1}\left(\frac{{V_y}^{\mathrm{est}}}{{V_x}^{\mathrm{est}}}\right) \tag{15}$$

$$\mathrm{Loss}_{\mathrm{Ori}} = \begin{cases} 1/2\left(\mathrm{Ori}^{\mathrm{gt}} - \mathrm{Ori}^{\mathrm{est}}\right)^2, & \text{if } |\mathrm{Ori}^{\mathrm{gt}} - \mathrm{Ori}^{\mathrm{est}}| < a \\ a\left(|\mathrm{Ori}^{\mathrm{gt}} - \mathrm{Ori}^{\mathrm{est}}| - 1/2a\right), & \text{if } |\mathrm{Ori}^{\mathrm{gt}} - \mathrm{Ori}^{\mathrm{est}}| \geq a \end{cases} \tag{16}$$

where $\mathrm{Ori}^{\mathrm{gt}}$ denotes the orientation direction of the target's ground truth velocity, and $\mathrm{Ori}^{\mathrm{est}}$ denotes the direction of the target's estimated velocity.

e) Offset loss: It is proposed to add an offset to the radial velocity values of each point in order to minimize the noise and obtain accurate velocity estimation. However, this is prone to overfitting and if no limit is placed on the added offset values, the network can be overtrained to output large offset values based on the training data. On the other hand, the main intention of using the offset is only to add small values in radial velocity of points with higher noise to compensate for it. To achieve this, a Lasso regression is applied where a loss value is computed as $L1$ norm of the amplitude of all the offset values. This allows the network to overall learn to predict small offsets and only give higher offset values to the points with higher noise in radial velocity. The proposed loss function is given by the following equation:

$$\mathrm{Loss}_{\mathrm{Off}} = \sum_{n=1}^{n=N} |{\mathrm{Off}_n}^{\mathrm{est}}| \tag{17}$$

where N denotes the number of points in the point cloud, and ${\mathrm{Off}_n}^{\mathrm{est}}$ denotes the offset estimated from the model for each point.

Combining all components, the proposed loss function for the whole network can be written as follows:

$$\mathrm{Loss}_{\mathrm{All}} = \mathrm{Loss}_{\mathrm{Mot}} + a \cdot \mathrm{Loss}_{\mathrm{Dopp}} + b \cdot \mathrm{Loss}_{\mathrm{Grad}} + c \cdot \mathrm{Loss}_{\mathrm{Ori}} + d \cdot \mathrm{Loss}_{\mathrm{Off}} \tag{18}$$

where a, b, c, and d are hyperparameters that should be adjusted empirically. It should be noted that the losses defined in (4), (8), (13), and (16) are also computed for all N points in the point clouds, as explicitly written for the loss component in (15).

C. Approach 2: DNN-Based Instantaneous Velocity Estimation

Although the WLS-based approach can help in reducing the overall error in velocity estimation, there are some fundamental limits to methods based on the radial velocity–azimuth profile, which essentially perform worse when the underlying rigid body assumption is inaccurate and the number of detected points per target is small. This is indeed the case for some automotive radar scenarios, resulting in inaccurate velocity

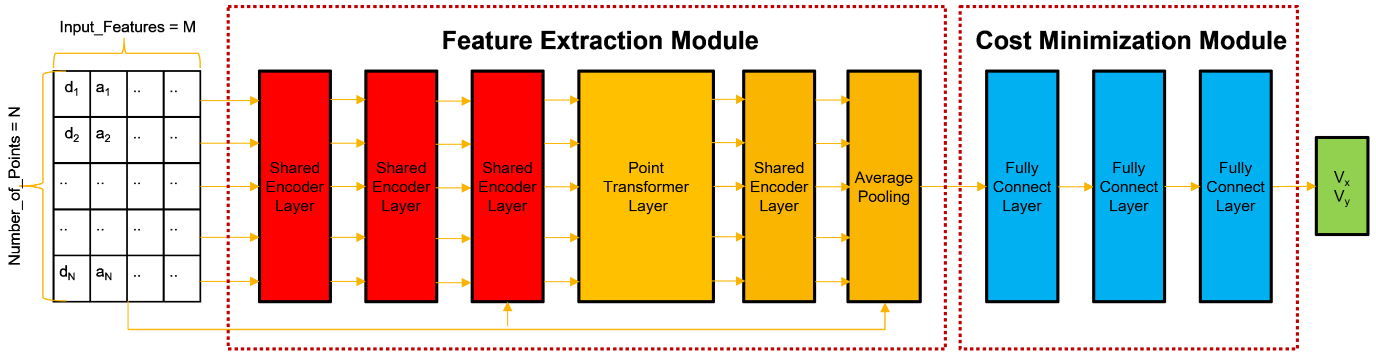


Fig. 3. Architecture for proposed Approach 2: It consists of a feature extraction module to extract pointwise local features which are then passed to a point transformer block to obtain complex local and global features. These are in turn passed through a cost minimization module to output velocity components.

estimate for these cases. Therefore, in the second proposed approach, a DNN that directly outputs the target velocity based on training data is designed, along with suitable loss functions.

Fig. 3 shows the architecture of the proposed method. The model is inspired by *PointTransformer* [20] and uses the self-attention mechanism [21] to learn the complex relationships between different points. Specifically, the points from detected targets are given as the input to the model. These points are then processed by a local feature extraction module containing shared MLP encoder layers to obtain local features for each points. These are then passed to a point transformer block. This is the core function which extracts complex features between different points using a point transformer layer to obtain discriminative features. Essentially, the input to this block is a set of N points with x features. These are passed to a set of linear layers to obtain the desired features for the point transformer layer. The point transformer layer generates complex features between different points using a self-attention mechanism. These are then passed through the cost minimization module containing three MLP decoder layers to obtain the velocity estimates from these features. These layers serve as an approximation for a function that can obtain velocity estimates from points, and is minimized using the proposed loss function. It should be noted that compared to the PointTransformer architecture in [20], which uses multiple point transformer blocks, this architecture is much simpler and only a single block is used without any transition up/down layers. This is done to extract relationships between all points while having very few points per detected target (i.e., maximum 16 points), which can be efficiently processed using a single point transformer block. Also, a smaller number of output channels (i.e., 16) are used for each layer since there are few points per target.

1) Loss Function: As opposed to Approach 1, where a WLS-based solution of the radial velocity–azimuth profile is sought after, for this approach it is undesirable to restrict the model to strictly follow such a curve, as this can fail on challenging cases such as when there are only few target points. In order to keep the training of the network as general as possible, a loss function that minimizes only the Huber loss between the ground truth velocity V^{gt} and predicted velocity V^{est} is used. The loss function is given by the following equation:

$$\text{Loss} = \begin{cases} 1/2\left(V^{\text{gt}} - V^{\text{est}}\right)^2, & \text{if } |V^{\text{gt}} - V^{\text{est}}| < a \\ a\left(|V^{\text{gt}} - V^{\text{est}}| - 1/2a\right), & \text{if } |V^{\text{gt}} - V^{\text{est}}| \geq a. \end{cases} \tag{19}$$

D. Addition of Velocity Estimation as an Additional Measurement for Tracking Algorithms

1) Motivation: Instantaneous velocity estimation of tracked targets can be very useful for a tracker, especially during changing track dynamics. For example, during a change in the movement of the target, the tracked states significantly deviate from the true states. The availability of the velocity estimation can be thus helpful in determining that there is a change in movement, thereby reducing the error in tracked states. As an illustration of this effect, a 1-D movement of a tracked target is simulated, with results presented in Fig. 4. Initially, the tracked target is static and then accelerates for 5 s with an acceleration rate of 4 m/s^2. Then, it moves at a constant speed of 20 m/s for 5 s. Afterwards, it comes to a stop again by decelerating at a rate of 4 m/s^2 for a duration of 5 s. Four different trackers are compared. The first one is a constant velocity (CV) tracker that uses only position measurement for tracking. The second is a CV tracker that uses both position and velocity as measurement. The third is a constant acceleration (CA) tracker that uses only position measurement. The fourth is a CA tracker that uses both position and velocity as measurement. Gaussian noise with a standard deviation of 0.5 is added to both position and velocity measurements.

Fig. 4 compares the error in position and velocity estimation using each of the four trackers. It can be seen that the trackers using both position and velocity measurements yield lower error than those using only position measurement. Also, the trackers using a CV model have higher error in general, showing that an incorrect tracking model results in higher errors.

2) Implementation Details: To analyze the performance of the proposed velocity estimation on tracking algorithms, the estimated velocity is integrated as an additional measurement inside a tracker. For comparison, three trackers are used: the first uses only position as the measurements, the second uses

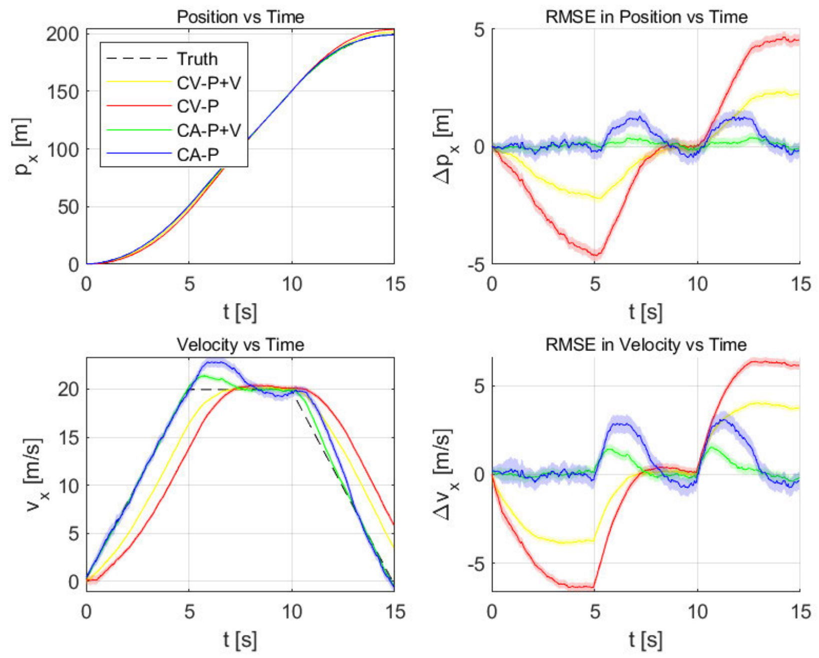


Fig. 4. Motivation for using instantaneous velocity measurements within a tracker: position estimation (top-left), RMSE in position (top-right), velocity estimation (bottom-left), and RMSE in velocity (bottom-right). The results compare CV tracker using position measurements (red), CV tracker using position and velocity measurements (yellow), CA tracker using position measurements (blue), and CA tracker using position and velocity measurements (green). Trackers using both position and velocity measurements yield better results, especially during changes in velocity.

position and velocity as the measurements but the velocity is estimated by the approach in [6], and the third uses position and velocity as measurement where the velocity is obtained using the proposed Approach 2. MATLAB's implementation [22] of an integrating multiple model (IMM) filtering technique [23] is specifically used for the state estimation where it is assumed that the tracked target can follow three motion models including CV, CA, and constant turn. Here, CA was used as the initial motion model. This was used because targets usually follow a linear motion and CV motion can be incorporated if the acceleration is set to zero. The states include position (x, y), velocity (V_x, V_y), and acceleration (A_x, A_y) in 2-D.

Since the velocity estimation module needs at least two detections per target and there can be scenarios where the number of points is less than two, there is a difference between the number of position and velocity measurements. As both the position and velocity are needed as measurements for the tracker, the predicted velocity from the motion model is given as a measurement when the velocity measurement is not directly available. Also, when there is a difference of more than 5 m/s between the predicted velocity from tracker's motion model and estimated velocity measurement, it can be assumed that the velocity estimation is incorrect and so the predicted velocity is taken as the measurement. This reduces the errors due to highly inaccurate velocity estimations, e.g. in cases when the estimation based on [6] fails, and the number of detected points is too few.

IV. Results and Discussion

This section describes the analysis of the experimental data and the results achieved using the proposed methods for instantaneous velocity estimation.

A. Experimental Dataset

In order to evaluate the performance of the proposed methods on real driving scenarios, the 2-D open-source RadarScenes dataset [17] is used. This is a very large dataset comprising a total of 158 different scenes that provide a large variability in ego and target motion as well as scene conditions. These scenes are divided into two parts: 130 training sequences and 28 testing sequences, each having diverse multiple scenarios where the ego-vehicle is static or moving. There is also a large variety in the different classes of targets that are present in the scenes. However, our methods are evaluated only on car targets given their importance and variability in shape, size, and maneuvers they can perform.

Fig. 5 provides an illustration of the setup for collecting this dataset. Essentially, this provides recordings from a set of four automotive radar sensors which are present at different locations and with different orientations. Specifically, two radars (radar 2 and radar 3) are tilted by 25°, whereas the other 2 radars (radar 1 and radar 4) are tilted by 85°. Each of these radars provides a field of view of 120°, maximum detection range of 100 m, range resolution of 0.15 m, radial velocity resolution of 0.1 km/h, and an angular resolution that ranges from 0.5° to 2° from the center to the side of the field of view.

There are multiple point detections per target in the RadarScenes dataset, thanks to the resolution of the radars and cases of overlap when the same target is seen by multiple radars. This is useful for velocity estimation because it results in the availability of more detected points and reduces the impact of noise in the estimation process. Also, the usage of multiple sensors which are present at different locations and orientations makes available a larger radial velocity–azimuth profile curve span, which is beneficial.

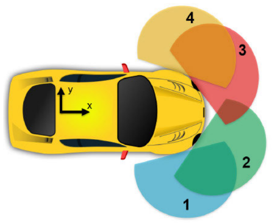


Fig. 5. Visualization of the RadarScenes dataset setup, taken from [16]. There are four radar sensors present at different orientations to cover a wide field of view. In this work, the coordinate system is used as shown in this figure.

Notably, RadarScenes provide points with track IDs. These can be used to directly obtain the set of points detected for each target by combining the points with same track ID; these are then used as inputs to the proposed networks for velocity estimation. Although the RadarScenes dataset provides detected points with track IDs, the ground truth velocity of the targets is not directly available. To the best of the authors' knowledge, this is a major issue common to all real-world driving scenario datasets, as it is difficult to obtain ground truth velocity of targets in real driving scenarios. Although some works obtain ground truth velocity in a controlled setting with very few targets instrumented with GPS [13], this approach is not scalable or feasible when a lot of different targets are needed to provide a variety in target motions. In order to still obtain a form of ground truth to compare the results of the proposed methods, it was decided to estimate the target ground truth velocity using a filtering approach.

In this respect, the position of each target within a frame is first obtained. This is found by clustering all the points with the same track ID and using the center of a bounding box generated using these points. This gives the position (i.e., not velocity) of different targets in each frame with their corresponding track IDs. Next, this position information is used to track the targets over frames and the tracked target's velocity is estimated based on the tracker's estimate. This is done using an IMM tracker [23] that considers three possible motion models including CV, CA, and constant turn with CA model as the initial motion model assumption. Here, the motion is tracked using an extended Kalman filter [24] to incorporate nonlinear process noise. Moreover, a smoothing filter is applied on top of the EKF output to refine the estimates because information about position at all frames is available. This can give more accurate results than using a conventional filter which only uses information from the previous frames. A comparison of the performance of the proposed tracker with a conventional tracker for a simulated scenario is shown in Fig. 6. Here, the target starts to accelerate from a static state. This change in target motion results in an inaccurate state estimation for both trackers. However, the proposed tracker follows the ground truth more closely as compared to the conventional tracker. This is because it has information about the future frames. Afterwards, the target moves with a CV. The proposed tracker adjusts to this change quickly as compared to the conventional tracker, resulting in a lower error. Finally, the target decelerates and comes to a halt. The proposed tracker is able to follow the ground truth velocity more closely as compared to the conventional tracker.

Although the proposed approach shows decent performance, there are limitations in this method. First, since the target center position is estimated based on the bounding box generated from the set of detected radar points, this estimated position is not completely precise. This will impact the ground truth position which may in turn affect the estimated ground truth velocity. Second, a deviation in estimated state from the true state is observed when there is an abrupt change in motion (i.e., a shift of motion from CV to acceleration). Although the usage of smoothing filter reduces this impact, optimal filter parameters for different targets cannot be estimated that may result in error. Finally, the performance of the tracking output is highly impacted during a turn of the target. Since targets mostly move linearly, a higher confidence is given to CA as compared to constant turn motion. Since it is extremely difficult to verify the target's ground-truth turn rate, this work does not estimate turn rate.

These limitations in the ground truth may introduce training bias especially for the proposed DNN method which does not directly use a specific mathematical model. Furthermore, the proposed NN + WLS method may also be impacted because it follows a radial velocity–azimuth profile model, but this will not be satisfied during these cases, resulting in higher training error. Finallyy, this may impact evaluation since the ground truth itself is not completely accurate.

B. Evaluation Metrics

Suitable evaluation metrics are required to objectively evaluate the performance of different methods for instantaneous target velocity estimation. In this work, the following metrics are considered.

1) RMSE: This measures the difference between the predicted and ground truth velocity by computing the root MSE. This is a popular metric, used frequently to provide a good estimation of the accuracy. However, it is sensitive to outliers since it uses a square term. In real automotive scenarios, outliers are often present and can greatly affect the value of this metric. Specifically in this work, where the ground truth is not directly available but estimated, and where there are scenarios with very few detected points per target, this metric may show very high values because of outliers. Hence, more metrics are considered as follows.

2) MAE: This metric compares the predicted and ground truth velocity by computing the MAE, and is less sensitive to outliers as compared to RMSE. For this reason, this metric is mostly considered here to assess velocity estimation performances.

3) Saturated-RMSE: This is a variant of the RMSE whereby the maximum error per target is set to a fixed threshold value to reduce the impact of noise and outliers; the threshold is in this case set to 10.

4) High Error Count: This metric provides an account of the number of times when a method under test fails by giving an

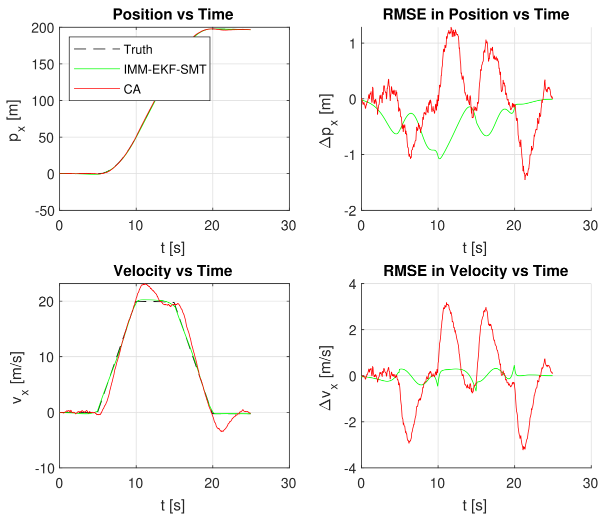


Fig. 6. Comparison of state estimation between the proposed tracker (green) and a conventional tracker (red) compared to ground truth (dashed line): position estimation (top-left), RMSE in position (top-right), velocity estimation (bottom-left), and RMSE in velocity (bottom-right). The proposed tracker gives better velocity estimation, especially during change in motion.

error exceeding a given threshold, here set to 10. This metric is not sensitive to outliers.

C. Analysis of Results Compared With Alternative Methods

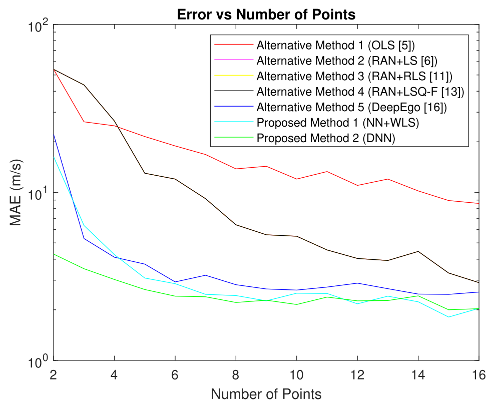
This section provides a comparison of the proposed methods for instantaneous velocity estimation with alternative ones from the literature. The effect of different parameters and scenarios is specifically considered.

1) Effect of Number of Detected Points Per Target: An analysis of the impact of the number of detected points per target on the velocity estimation methods is performed. Fig. 7 shows a plot of the MAE versus number of detected points per target, for both scenarios with static ego and those with linearly moving ego. It can be seen that the proposed DNN method, which does not depend explicitly on the radial velocity–azimuth profile, is robust even with a lower number of detected points per target. In general, also the other proposed approach of this article, the NN-based WLS, achieves good performance for a number of points higher than 4, compared with the alternative methods from the literature. It is also observed that the RANSAC method [6] performs worse than the OLS method [5] when the number of points decreases to a value less than 5. This is because in the case of fewer points per targets, RANSAC selects only a smaller subset of points as inliers resulting in even fewer points to be available for the solution. DeepEgo [16] gives a higher error than the proposed NN + WLS method except for cases with a very small number of points. This is due to the usage of nonrobust MSE loss which causes the training to focus on those outlier cases. The three different alternative methods (RAN + LS [6], RAN + RLS [11], and RAN + LSQ-F [13]), which use RANSAC-based removal of points but utilize different methods to solve least square solution, performed similar to each other, showing that the major impact on performance is dependent on the RANSAC-based removal of points, not on the method for finding the least squares solution.

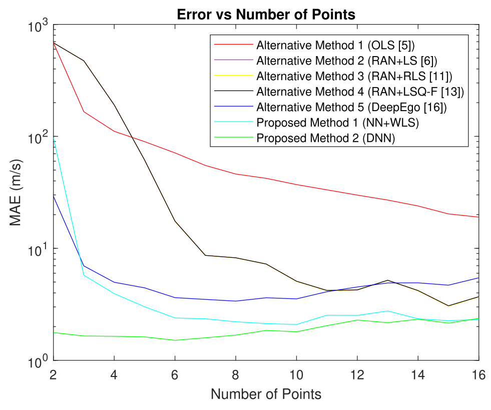
Based on the initial results shown in Fig. 7, it was decided to further evaluate the performance with a different number of minimum points per target. First, the minimum number of points per target was set to four, considered a border-line case where approaches relying on the radial velocity–azimuth curves start to fail. The second case uses a minimum number of points per target set to eight, in order to correctly evaluate the performance of the proposed NN + WLS method (Approach 1 in Section III), which achieves a robust performance for this number of points. This evaluation was done separately for scenarios with static and linearly moving ego. It is important to note that frames where the ego-vehicle makes a rotation are not included in this study, because this would make the radial velocity–azimuth profile deviate from the equations used in Section III, resulting in all the methods to fail, except the proposed DNN method (Approach 2 in Section III).

2) Results on Static Ego: The initial performance study considers static ego to avoid issues from inaccurate modeling of ego-motion. First, the evaluation is performed on relatively simpler scenarios with cars having at least eight detections. This resulted in a total of 11 371 train and 2788 test targets from the train and test scenes, respectively. In the second set of experiments, the evaluation is extended to scenarios with cars having at least four detections to consider more challenging cases. This resulted in a total of 23 436 train and 5473 test targets from the train and test scenes, respectively.

Tables III and IV show a comparison of the MAE obtained in the velocity estimation using the different considered methods, averaged across all targets. Here, V_x denotes the error in down-range and V_y in cross-range relative to the ego-vehicle direction. V is then obtained as the square root of the sum of V_x and V_y squared. It can be observed that both proposed methods provide a lower error than the alternative methods, with the DNN-based method providing the lowest error. This is because in real-world driving scenarios, targets can make turns and deviate from the linear motion assumption used



(a)



(b)

Fig. 7. Effect of number of detected points per target on MAE for different instantaneous velocity methods: red line denotes errors obtained using an OLS method (OLS [5]); magenta line refers to a method using RANSAC to remove outliers followed by a least square solution on the remaining points (RAN + LS [6]); yellow line refers to a method using recursive least square solution whereby the initial velocity estimate is obtained using RAN + LS [6] (RAN + RLS [11]); black line refers to a method using RANSAC to remove outliers followed by a least square solution to obtain linear and rotational velocity (RAN + LSQ-F [13]); blue line represents a hybrid NN method that generates weights for each point and uses those inside a WLS solution based on DeepEgo [16]; cyan line represents the proposed method 1, which uses a hybrid NN to estimate weights for a WLS solution (NN + WLS); green line represents the proposed method 2, which uses a DNN with a point transformer block to directly output velocity values (DNN). Both proposed methods provide lower error than alternative ones, with the DNN method being more robust to different number of points. It must be noted that the alternative methods (RAN + LS [6], RAN + RLS [11], and RAN + LSQ-F [13]) give similar performance resulting in an overlap of the curves. (a) Cases with static ego. (b) Cases with linearly moving ego-vehicle.

TABLE III
MAE IN INSTANTANEOUS VELOCITY ESTIMATION FOR DIFFERENT METHODS—STATIC EGO; CAR TARGETS WITH MINIMUM EIGHT DETECTED POINTS

MAE (m/s)	***Vx***	***Vy***	***V***
OLS [5]	4.24	9.36	10.3
RAN+LS [6]	1.17	2.97	3.20
RAN+RLS [11]	1.19	2.92	3.15
RAN+LSQ-F [13]	1.20	2.95	3.19
DeepEgo [16]	1.10	1.79	2.10
Proposed NN+WLS	1.04	1.65	1.96
Proposed DNN	1.32	1.72	2.18

TABLE IV
MAE IN INSTANTANEOUS VELOCITY ESTIMATION FOR DIFFERENT METHODS—STATIC EGO; CAR TARGETS WITH MINIMUM FOUR DETECTED POINTS

MAE (m/s)	***Vx***	***Vy***	***V***
OLS [5]	7.21	14.0	15.8
RAN+LS [6]	5.11	9.10	10.4
RAN+RLS [11]	5.09	9.07	10.4
RAN+LSQ-F [13]	5.12	9.12	10.5
DeepEgo [16]	1.54	2.64	3.06
Proposed NN+WLS	1.37	2.35	2.72
Proposed DNN	1.37	2.00	2.43

in approaches relying on the radial velocity–azimuth profile. On the contrary, the DNN-based method does not assume a particular target motion modeling and does not suffer from this issue.

It can also be observed that the proposed NN-based WLS method provides the lowest error out of the radial velocity–azimuth profile-based methods. This is because appropriate weights can be given to outliers and inliers with different noise levels. Additionally, implicit characteristics of the target such as its orientation and gradient of radial velocity–azimuth profile can be predicted using orientation loss and gradient loss respectively, which helps in obtaining better velocity estimates. Finally, the noise in radial velocity measurements can be corrected using offset values that reduces the impact of noise on the velocity estimation as mentioned in Section III. On the other hand, OLS [5] is not able to remove the outliers resulting in the highest error. RAN + LS [6], RAN + RLS [11], and RAN + LSQ-F [13] can remove the outliers in most cases, but cannot provide different weights based on different noise levels, and also sometimes fail to remove outliers in challenging cases. DeepEgo [16] is able to provide a lower error than RAN + LS because it can give different weights to different points based on feature learning. However, this approach does not consider the characteristics of the target, and training can be unstable because of the usage of MSE loss (instead of the Huber loss proposed here). Also, no regularization has been applied to offset values, resulting in a higher error than the proposed NN-based WLS method.

It is also observed that for the case when only cars with a minimum of eight detected points are considered, the DNN method provides higher error than the NN + WLS method. This is because the DNN needs more data for efficient training. For this case, the training data is insufficient to completely learn the patterns in data. It is expected that with the availability of more training data, the DNN will be able to show better performance than the NN + WLS method. On the other hand, the proposed NN + WLS method—which is a hybrid method where the NN part only generates weights rather than complete velocity estimates—can provide better results for cases where only few training data is available.

3) Results With Linearly Moving Ego: In this section, scenarios with moving ego-vehicle are considered as a further step from those presented with static ego-vehicle. Here, only frames with linearly moving ego-vehicle are analyzed. This is because

TABLE V
MAE IN INSTANTANEOUS VELOCITY ESTIMATION FOR DIFFERENT METHODS—LINEARLY MOVING EGO; CAR TARGETS WITH MINIMUM EIGHT DETECTED POINTS

MAE (m/s)	***Vx***	***Vy***	***V***
OLS [5]	9.52	27.5	29.1
RAN+LS [6]	3.02	4.10	5.09
RAN+RLS [11]	3.01	4.09	5.08
RAN+LSQ-F [13]	3.02	4.13	5.12
DeepEgo [16]	2.51	1.90	3.13
Proposed NN+WLS	0.97	1.10	1.47
Proposed DNN	1.55	1.05	1.88

TABLE VI
MAE IN INSTANTANEOUS VELOCITY ESTIMATION FOR DIFFERENT METHODS—LINEARLY MOVING EGO; CAR TARGETS WITH MINIMUM FOUR DETECTED POINTS

MAE (m/s)	***Vx***	***Vy***	***V***
OLS [5]	12.9	56.5	57.9
RAN+LS [6]	9.13	43.7	44.7
RAN+RLS [11]	9.12	43.7	44.7
RAN+LSQ-F [13]	9.18	43.8	44.8
DeepEgo [16]	3.21	2.95	4.37
Proposed NN+WLS	1.67	2.12	2.70
Proposed DNN	1.42	1.09	1.79

during the turning of the ego-vehicle, the underlying model of radial velocity–azimuth curves of many methods does not work. There are studies [9] that propose more complex models during turns of the ego-vehicle, but they require noise-free measurements. These can be made available in simulations or controlled test scenarios, but are not available in real-world driving scenarios such as in RadarScenes dataset used here. As for the static ego-vehicle case, two separate studies are performed, first considering cars with a minimum of eight detected points, and then with a minimum of four detected points. This resulted in a total of 73 344 and 13 328 targets from the train and test scenes for the first experiment, and a total of 156 543 and 27 019 targets for the train and test scenes for the second experiment, respectively.

Tables V and VI present a comparison of the MAE obtained in the velocity estimation using the different considered methods, averaged across all targets. Overall, the error is higher when the ego-vehicle is moving as compared to the cases when ego-vehicle is static. In any case, it can be observed that both proposed methods provide a lower error than the alternative methods, with the DNN-based method providing the lowest error. It can also be observed that the proposed NN-WLS method provides the lowest error out of those based on the radial velocity–azimuth profile. Overall, the improvement achieved using the proposed methods over the alternative methods is much greater for linearly moving ego case than for static ego. Specifically, the DNN and NN + WLS methods provide a reduction in MAE of 2.58 m/s (or 59%) and 1.67 m/s (or 38%) as compared to DeepEgo [16] respectively, whereas OLS [5], RAN + LS [6], RAN + RLS [11], and RAN + LSQ-F [13] methods result in a very high MAE. As for static ego-vehicle, it is observed that the NN + WLS approach provides lower error than the DNN one when cars with a minimum of 8 detections are considered. This is because it is easier to train the NN + WLS using a smaller number of samples as compared to the more "black-box" DNN which requires more samples to correctly learn the target features. Finally, the improvement achieved by the proposed methods compared to the alternative ones is more significant for linearly moving ego-vehicle cases than for static ones.

D. Detailed Analysis of Results

Additional tests are performed to evaluate the proposed methods in this section. For conciseness, only cases with static ego-vehicle and target cars with a minimum of 4 detected points are considered.

1) Error Distribution in Space: An analysis of the spatial distribution of the error in velocity estimation was performed. Fig. 8 shows a heat map of the MAE obtained using the proposed NN + WLS method at different spatial locations, where the MAE for all targets that lie within a particular bin in the spatial grid is averaged. The color bar denotes the error from lower (yellow) to higher (red). Regions with no targets are represented in white. Fig. 8(a) shows that the MAE in target down-range velocity V_x is lower at regions in front of the ego car (azimuth close to 0°). This is because for these regions the target down-range velocity V_x is close to the radial velocity component which is directly available as a point cloud feature. However, as the azimuth increases, the MAE in V_x increases because of the shift toward the tangential direction, resulting in a less accurate estimation of V_x. On the other hand, Fig. 8(b) shows that the MAE in target cross-range velocity V_y is higher at regions with azimuth close to 0°. This is because in these locations, the target velocity V_y component is tangential to the ego-vehicle's front direction, making its estimation less accurate.

Fig. 9 shows a similar heatmap when using the proposed DNN method for the instantaneous velocity estimation. The trend in V_x component across the space is similar to NN + WLS method with lower MAE at regions in front of the ego car (azimuth close to 0°). However, there seems to be no dominant trend in V_y component across the space in this case.

2) Error in Case of Radar Failure: The networks in the two proposed approaches, NN + WLS and DNN, have been trained with data from all the four radars of the RadarScenes dataset. As in real scenarios there can be failures or issues with data from one or more sensors, an analysis has been made here to evaluate the velocity estimation performance when only a subset of radar sensors is used during inference.

Table VII shows the relevant results. It should be noted that in this case all targets are retained, even those with a number of detected points lower than four once data from some specific radar is removed. A drop in performance for the NN + WLS approach is noted as the number of radars decreases. This is because of two reasons. First, there is a decrease in the number of points per target when the radars are removed, resulting in many cases where there are less than four points. This resulted in a significant error for methods based on the radial velocity–azimuth profile such as NN + WLS. Second, with the decrease in the number of radars, the azimuth span of the target is reduced, which results in the availability of a smaller

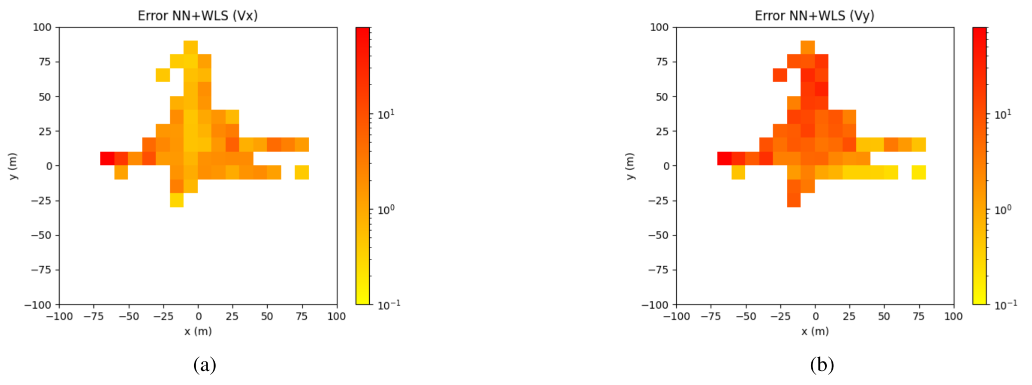


Fig. 8. Distribution of MAE in X–Y spatial dimensions using the proposed NN + WLS approach. (a) MAE in V_x component. (b) MAE in V_y component.

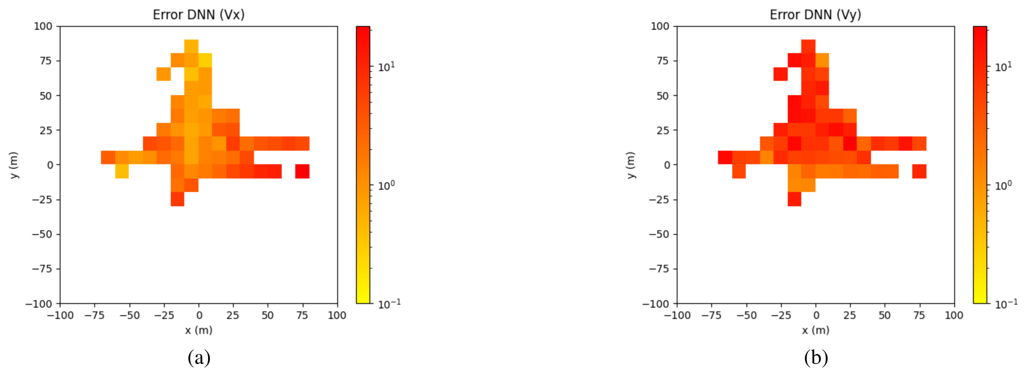


Fig. 9. Distribution of MAE in X–Y spatial dimensions using the proposed DNN approach. (a) MAE in V_x component. (b) MAE in V_y component.

TABLE VII
MAE IN INSTANTANEOUS VELOCITY ESTIMATION BY REMOVING RADARS—KEEPING TARGETS WITH LESS THAN FOUR POINTS

MAE (m/s)	WLS	DNN
All Sensors	2.72	2.43
Sensor 1	16.3	3.91
Sensor 2	19.4	2.71
Sensor 3	52.0	2.96
Sensor 4	15.9	3.26
Sensors 1+2	48.9	2.93
Sensor 2+3	10.7	2.66
Sensor 3+4	17.1	2.47
Sensor 1+2+3	4.61	2.63
Sensor 2+3+4	7.93	2.45

TABLE VIII
MAE IN INSTANTANEOUS VELOCITY ESTIMATION BY REMOVING RADARS—REMOVING TARGETS WITH LESS THAN FOUR POINTS

MAE (m/s)	WLS	DNN
All Sensors	2.72	2.43
Sensor 1	2.53	2.33
Sensor 2	3.21	2.45
Sensor 3	2.97	2.41
Sensor 4	3.80	2.24
Sensors 1+2	2.78	2.59
Sensor 2+3	2.69	2.42
Sensor 3+4	3.22	2.31
Sensor 1+2+3	2.64	2.52
Sensor 2+3+4	2.77	2.33

portion of the radial velocity–azimuth profile. This causes the noise to have a higher effect and worsens the performances. The DNN-based method on the other hand is able to maintain performance with a small increase in error, since it is more robust to targets having a lower number of detected points.

For a more complete comparison, an analysis is also shown when targets with fewer than four detected points are removed, with results shown in Table VIII. Comparing these with Table VII, it can be observed that both proposed methods are relatively robust to sensor failures, provided that the targets with very few detected points are not considered. Essentially, the drop in performance observed for Table VII is due to the cases of targets with very few detected points. Overall, the MAE for the NN + WLS method is higher than for the DNN method because of the reasons mentioned in the previous paragraph.

3) Additional Study on Proposed Method 1 (NN + WLS): An additional analysis of the impact on MAE using different components of the proposed NN + WLS method was performed. Table IX provides a breakdown of the MAE where successively one component of those described in Section III-B is removed from the training procedure. It can be seen that the removal of each proposed component results in increasing MAE, with the addition to offset values having the highest impact in increasing the resulting error.

a) Effect of loss functions: Instantaneous velocity estimation for a real target is a difficult problem because the span of the radial velocity–azimuth profile curve is small, resulting in

TABLE IX
MAE IN INSTANTANEOUS VELOCITY ESTIMATION BY REMOVING DIFFERENT COMPONENTS OF THE PROPOSED NN + WLS METHOD

MAE (m/s)	**Vx**	**Vy**	**V**
NN+WLS	1.37	2.35	2.72
No Offset	2.40	4.83	5.39
No Offset / Gradient Loss	2.77	5.40	6.07
No Offset / Gradient / Orientation Loss	4.32	8.95	9.94
No Offset / Gradient / Orientation / Doppler Loss	5.69	9.64	11.2

small noise to have a large impact on the estimation. In order to mitigate this, gradient and orientation losses are proposed during training to guide the network to implicitly correct the gradient of the radial velocity–azimuth profile, and implicitly estimate the velocity direction to be toward the orientation of the target. This allows the model to predict weights that can generate a velocity prediction which satisfies these physically meaningful constraints more closely, thus reducing error.

Fig. 10 shows an example from the RadarScenes dataset (sequence 19) when a car is moving in a lateral direction ($V_x \sim 0$ m/s, $V_y \sim 8$ m/s) and is located at a high azimuth angle such that the gradient of the radial velocity–azimuth profile curve has a high impact on V_x. Fig. 10(a) shows the spatial location of the detected points. From the detections, the orientation of the car can be estimated which is aligned in the direction of y as the extent of the detections is aligned in this direction. As the car motion is also physically aligned with its spatial orientation, this fact can help guide the networks to estimate the velocity correctly (i.e., a small V_x value and larger V_y value). Moreover, Fig. 8(b) shows the radial velocity–azimuth profile curve generated by models which are trained with/without orientation/gradient loss as defined in Section III-B. It can be seen that the gradient of the profile generated using a model trained with these losses is aligned with that generated using the ground truth velocity, resulting in a lower error in instantaneous velocity estimation. Conversely, the profile generated using a model trained without these losses is misaligned with that generated using ground truth velocity. This happens because there is a point which has a "noisy" radial velocity value; this is assumed to be an inlier and caused a shift in the gradient of the curve. This results in a large error.

b) Effect of adding offset values to radial velocity: As opposed to the task of ego-motion estimation, where there are many detected points, for instantaneous target velocity estimation only few points are usually detected by a target. This results in 'noisy' points having a significant effect on the velocity estimation. Even though a WLS solution can reduce this effect by giving low weights to such points, this may not be enough for accurate velocity estimation as noise in such points is not removed. To further address this issue, an additional offset to the radial velocity values was proposed as described in Section III-B. However, in order to avoid overfitting on the training data by having sporadic changes in the radial velocity values, an offset loss term is also added to give high values when the sum of magnitude of offsets is higher. Notably, this is different from the approach of DeepEgo [16], which does not add any loss term based on the offset values, leading to unstable results in different scenarios considered in this work.

TABLE X
ERROR IN V_x FOR DIFFERENT METHODS USING VARIOUS METRICS

	RMSE	**Sat-RMSE**	**High Error Count**
OLS [5]	35.7	4.28	647
RAN+LS [6]	87.4	2.83	236
RAN+RLS [11]	87.4	2.81	236
RAN+LSQ-F [13]	87.5	2.84	235
DeepEgo [16]	3.76	2.31	62
Proposed NN+WLS	8.28	1.78	44
Proposed DNN	2.20	2.11	28

TABLE XI
ERROR IN V_y FOR DIFFERENT METHODS USING VARIOUS METRICS

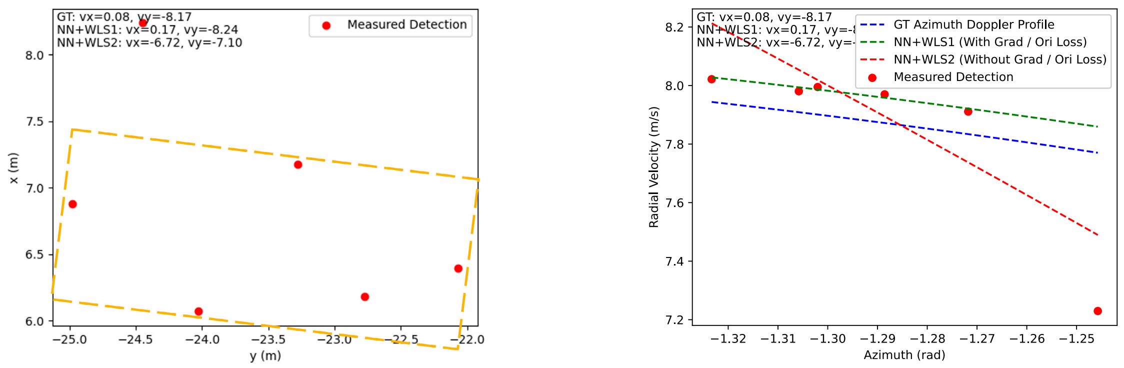
	RMSE	**Sat-RMSE**	**High Error Count**
OLS [5]	40.7	6.13	1451
RAN+LS [6]	35.4	5.17	886
RAN+RLS [11]	35.4	5.14	876
RAN+LSQ-F [13]	35.5	5.17	888
DeepEgo [16]	4.74	3.43	194
Proposed NN+WLS	5.1	3.12	132
Proposed DNN	2.95	2.82	53

Fig. 11 shows an illustration of how adding offset can help in reducing the velocity estimation error. Fig. 11(a) shows that when no offset values are used, the model is not able to provide an accurate solution because of the points with low radial velocity values in the bottom-right area. These will distort the estimated radial velocity–azimuth profile with respect to the ground truth. Fig. 11(b) shows that when offset values are used together with the correct loss functions for training, the network is able to generate a WLS solution that is closer to the ground truth. Specifically, the points with low radial velocity values in the bottom-right area are shifted upward toward the expected curve direction.

4) Evaluation Using Other Metrics: MAE metric is good to evaluate the performance of different methods in the presence of outliers. However, for further evaluation, the results of the other metrics described in Section IV-B are also presented.

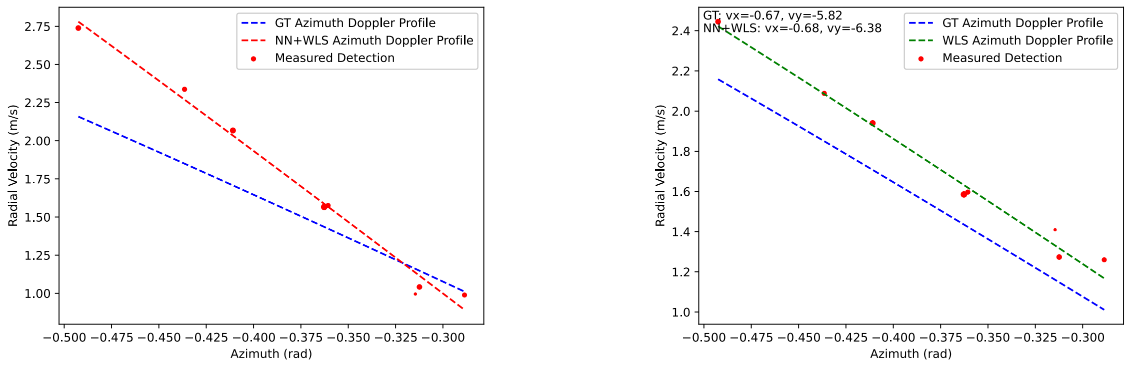
Table X presents the results for the estimation of V_x and Table XI for V_y for each of the different methods. Both proposed approaches provide lower errors than other methods, with the DNN method providing the lowest error for all metrics. As an exception, the NN + WLS method gives higher RMSE than DeepEgo [16]. This is because RMSE is prone to outliers, resulting in high value when some cases with very large errors are present. However, in general NN + WLS has a better performance. This can be confirmed by the fact that when a threshold on the maximum error is set using Saturated-RMSE, NN + WLS performs better than DeepEgo. Also, the high error count is smaller for NN + WLS than DeepEgo, showing that the proposed method performs well. Also, the RANSAC-based methods (RAN + LS [6], RAN + RLS [11], and RAN + LSQ-F [13]) show similar performance, with the RAN + RLS [11] method resulting in slightly better performance than the other two.

Fig. 12 illustrates the cumulative distribution of the square error for the different methods. It can be observed that the proposed NN + WLS method generally provides lower error



(a) (b)

Fig. 10. Improvement in instantaneous velocity estimation with versus without gradient/orientation loss in the proposed approach presented in Section III-B. (a) Spatial location of detected points for a target vehicle, whose orientation can be implicitly estimated. (b) Radial velocity–azimuth profile for models trained with versus without gradient/orientation loss, compared with ground truth.



(a) (b)

Fig. 11. Improvement in instantaneous velocity estimation using additional offset values for the radial velocity as presented in Section III-B. (a) Radial velocity–azimuth profile for model without offset. (b) Radial velocity–azimuth profile for model with offset.

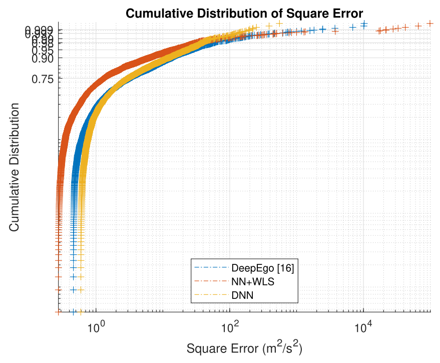
for most of the cases. However, there are few cases when the square error is very high, resulting in its mean (i.e., MSE) and consequently RMSE to become very high. Notably, DeepEgo has an overall higher error for most targets and more cases when the square error passes the threshold of high error count (i.e., absolute error: 10 m/s, square error: 100 m^2/s^2). On the other hand, the proposed DNN method is least affected by outliers since it is trained not to output outlier velocity values with very high magnitude. This is also evident from Tables X and XI where RMSE and Sat-RMSE are similar for the proposed DNN method.

E. Integration of Estimated Velocity as an Additional Measurement Inside a Tracker

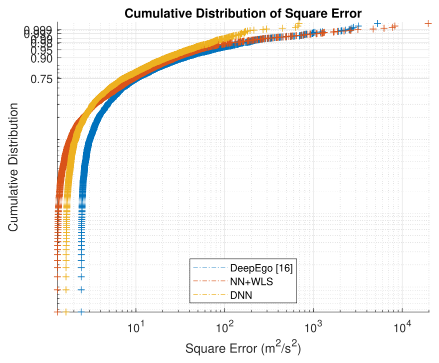
Instantaneous target velocity can be used as an additional measurement inside a tracker to help in state estimation, especially during changing track dynamics. For a performance assessment, three trackers using different measurements were considered. The first tracker uses only position as the measurement. The second tracker uses both position and velocity as the measurement, where the velocity is given by the RANSAC + LS-based method in [6]. The third tracker uses both position and velocity as the measurement, where the velocity measurement is obtained using the proposed DNN-based velocity estimator.

All trackers used an IMM approach [23] with three possibilities for tracking including CA, CV, and constant turn (CT) models, where the initial model was taken as CA. The motion and measurement model parameters were kept constant for each tracker. It should be noted that there are cases when it is not possible to estimate the instantaneous velocity correctly, such as cases when the tracked target has fewer than two detected points, or is turning. In these cases, the tracked velocity obtained using state prediction at the previous frame was kept as the current velocity measurement. However, to account for the uncertainty in measurement, the velocity components of the state covariance matrix were given higher value by scaling with a factor of 10. Moreover, if the difference between the estimated velocity using the proposed approach and the tracked velocity using a state prediction at the current timestamp has a magnitude difference greater than 5 m/s, it was assumed that the instantaneous velocity estimation was inaccurate. Thus, the velocity predicted by the tracker at the current state was used as the current measurement.

Table XII presents the average RMSE obtained using different methods on all testing scenarios with static ego-vehicle in the RadarScenes dataset. This gives a total of 137 tracks. Average RMSE over all tracks is reported because of the RMSE wide usage in many tracking metrics (e.g., in OSPA [25]). The results show that the tracker using the proposed DNN-based



(a)



(b)

Fig. 12. Cumulative distribution of square error for the different instantaneous velocity estimation methods. Blue lines denote square error obtained using DeepEgo [16], red lines denote square error obtained using the proposed NN + WLS method, and orange lines denote square error obtained using the proposed DNN method. It is observed that NN + WLS provides the lowest square error for most cases, but also includes outlier cases with high square error. (a) Cumulative distribution of square error in V_x component. (b) Cumulative distribution of square error in V_y component.

TABLE XII
ERROR IN TRACKER'S STATE ESTIMATE: STATIC EGO—ALL TRACKS (137 TRACKS)

Average RMSE	**Pos**	**Vel**
Tracker with Pos Measurement	1.01	3.45
Tracker with Pos / (RAN+LS) Vel Measurement	7.63	22.0
Tracker with Pos / (DNN) Vel Measurement	1.00	2.03

TABLE XIII
ERROR IN TRACKER'S STATE ESTIMATE: STATIC EGO—TRACKS WITH CHANGING DYNAMICS (20 TRACKS)

Average RMSE	**Pos**	**Vel**
Tracker with Pos Measurement	1.74	7.50
Tracker with Pos / (RAN+LS) Vel Measurement	4.51	12.2
Tracker with Pos / (DNN) Vel Measurement	0.90	1.72

TABLE XIV
ERROR IN TRACKER'S STATE ESTIMATE: LINEARLY MOVING EGO—TRACKS WITH CHANGING DYNAMICS (100 TRACKS)

Average RMSE	**Pos**	**Vel**
Tracker with Pos Measurement	2.39	8.21
Tracker with Pos / (RAN+LS) Vel Measurement	7.22	116
Tracker with Pos / (DNN) Vel Measurement	1.38	2.96

instantaneous velocity estimation as measurement provides the best state estimation. It can also be observed that the tracker using only position measurement is able to achieve a good average RMSE. This is because most of the times, tracked targets move at a CV. In these cases, the prediction from the tracker's motion model gives an accurate estimation of the state. However, in the cases when there is an abrupt change in motion, the velocity prediction from this model becomes inaccurate, degrading the tracking performance. It is also observed that the performance of the tracker using RANSAC + LS-based velocity measurement is worse than the tracker using only position measurements. This is because for many cases the RANSAC + LS method provides an inaccurate velocity estimation, which is worse than the velocity estimated using tracker's motion model.

In order to investigate the performance on cases with changing track dynamics, a dedicated analysis was done on a subset of the relevant tracks in the RadarScenes dataset. Specifically, 20 tracks with high changes in dynamics were considered. Table XIII compares the performance of different methods on these cases. An improvement in tracker's state estimation is reported using the proposed DNN-based velocity measurement as compared to other methods. Also, the tracker using position measurement gives higher error for these cases compared to the cases in Table XII. This is because the tracker using only position measurement is not able to correctly estimate the states during changes in track dynamics. Finally, the error for the tracker using RANSAC + LS-based instantaneous velocity measurement is still higher than other methods, as there are many frames with incorrect velocity estimation.

In the next evaluation, scenarios with linearly moving ego-vehicle are considered, for a total of 100 tracks exhibiting changes in their dynamics. Table XIV shows the relevant performance using different methods. Overall, the error is higher as compared to scenes with static ego-vehicle. This is because of the errors in ground-truth data generation and velocity estimation due to the inaccuracy in tracker modeling and noise in the measurements. It is observed that the proposed DNN model provides the best performance. However, the improvement in average RMSE against the tracker using only position measurement is lower than for static ego-vehicle cases. The tracker using RANSAC + LS measurements provides a very high RMSE error on average. This is because of the addition of challenging cases, resulting at times in inaccurate velocity estimation.

In addition to the average RMSE, an analysis on some typical individual tracking scenarios was also performed. Fig. 13 shows an example of performance on a tracked target that moves linearly in front of the ego-vehicle and changes its speed during the measurement. The ego-vehicle eventually overtakes this tracked target terminating the track. The tracker using position measurement gives higher error at the start of the scene because the initial velocity is not known and it takes some time for the tracker to correctly estimate velocity. Afterwards, the tracked target accelerates, and the tracker using only position measurement is not able to estimate velocity correctly.

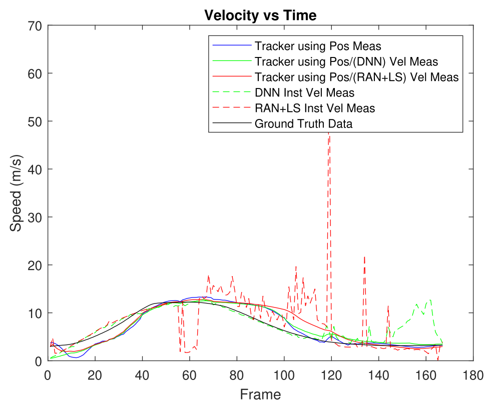


Fig. 13. Example of velocity estimation for a tracked target with changing dynamics: tracker using only position measurement (blue); tracker using position and velocity measurement obtained from the proposed DNN method (green); tracker using position and velocity measurement obtained from the RANSAC + LS method (red); instantaneous velocity estimation using the proposed DNN method (dotted green); instantaneous velocity estimation using RANSAC + LS method (dotted red); ground truth velocity (black). The tracker with the proposed DNN provides the best velocity estimation for most frames.

However, trackers using both position and velocity measurements are able to follow the ground truth velocity more closely. Afterwards, the tracked target starts to decelerate. All the trackers assumed a positive acceleration of the tracked target based on the past observations, and are not able to estimate this change initially, taking some time to correct this mismatch. However, the tracker using the proposed DNN method is able to recover faster than other methods resulting in a better state estimation. This is because accurate velocity measurement is available from the proposed DNN method. The tracker using RANSAC + LS-based velocity measurement results in higher error since many velocity estimations from the RANSAC + LS method are inaccurate during this time. It is also observed that the tracker using the proposed DNN method provides an inaccurate state estimation for the last few frames. This corresponds to the time when the ego-vehicle overtakes the tracked target. This is one of the failure cases of the proposed DNN method because this is a rare event not observed during training of the model.

From these results, it can be concluded that using velocity as a measurement inside a tracker helps in improving the velocity estimation for cases with changing track dynamics. However, this is sensitive to the accuracy of the velocity measurement. The velocity estimation given by the tracker can also degrade if the measurement is inaccurate.

F. Analysis on Real-Time Deployment

Table XV provides estimates of the real-time deployment needs of the two proposed methods. It can be observed that both proposed methods require very low memory and computational load, making them suitable to be implemented on embedded hardware. The parameter count for both methods is few Kbytes, so it is easy to fit these in SRAMs which are typically in MBytes. Moreover, given a frame rate of 10 Hz and 64 targets per frame, the compute is few GFlops. This makes it suitable to run on DSPs or NN accelerators which are typically in the range of hundreds of GOPs to few TOPs. This is further validated by the low runtime for both methods on a CPU core, showing their potential for real-time deployment.

TABLE XV
PRACTICAL REQUIREMENTS FOR THE TWO PROPOSED METHODS. THE INVESTIGATION PROVIDES THE NUMBERS FOR A SINGLE INFERENCE. INTEL XEON W-2245 CORE IS USED FOR RUNTIME EVALUATION

Method	Params (K)	Flops (M)	Runtime (ms)
Proposed NN+WLS	73.8	2.04	1.18
Proposed DNN	3.17	0.49	0.88

Both methods contain mostly standard layers which are supported by the NN accelerators. However, there are some details that need to be kept in consideration. First, the NN + WLS method involves matrix inversion operation which is not easily implementable in hardware. However, the size of the matrix is small here (2×2), so this will not be a bottleneck. Second, the DNN method contains a point attention layer that involves multiplication between each point. This typically requires high computational load and memory, but will not be an issue here because of 16 points per target. Hence, both proposed methods have good potential to be effectively implemented on embedded hardware for ADAS.

V. CONCLUSION

To address the challenge of distributed target tracking with radar(s), two NN-based solutions for instantaneous target velocity estimation are proposed. First, a novel hybrid NN-based WLS approach is proposed. The network takes a target point cloud as its input and extracts spatial–dynamic features to be used as weights for each input point. These weighted points are then used to obtain a weighted least square solution for instantaneous target velocity estimation. Second, a DNN is proposed to estimate instantaneously target velocity by learning directly from the radar data.

The proposed approaches have been validated on the RadarScenes experimental dataset, showing a significant improvement in target velocity estimation over the alternative state-of-the-art methods presented in the literature. Specifically, a reduction in the MAE of 59% can be achieved using the proposed DNN method as compared to the network presented in [16], which was found to be the best alternative method. Moreover, using the instantaneous velocity estimate as an additional measurement inside tracking algorithms improved performances, especially for cases with changing track dynamics.

Since it is challenging to obtain accurate ground-truth rotational velocity of the ego car and of the detected targets, this work is focused on linear instantaneous velocity estimation and assumed linear ego-vehicle motion. As part of future work, rotational velocity of the targets can be added into the models, as well as processing across multiple frames to estimate target acceleration. In this work, a rigid body assumption is taken which considers the velocity of each point to be the same. This assumption is typically correct for several target classes such as cars, bus, and trucks. However, it might not be true for classes such as pedestrians and cyclists, where the movement of body parts changes the velocity of different parts within the target. Accordingly, the accuracy of the proposed methods might be impacted on these classes. This will especially affect the performance of the proposed NN + WLS method which considers rigid body assumption in the radial velocity–azimuth equation to estimate velocity. However, the impact on performance for the proposed DNN method will be smaller since it does not assume a specific mathematical model and rather learns this function using data. As a future work, a thorough investigation on the performance of the proposed methods for different classes can be performed.

References

[1] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE Access*, vol. 8, pp. 58443–58469, 2020.

[2] E. NCAP. (2024). *Safety Assist*. [Online]. Available: https://www.euroncap.com/en/car-safety/the-ratings-explained/safety-assist

[3] S. Behere and M. Törngren, "A functional architecture for autonomous driving," in *Proc. 1st Int. Workshop Automot. Softw. Archit. (WASA)*, May 2015, pp. 3–10.

[4] R. Thakur, "Scanning LiDAR in advanced driver assistance systems and beyond: Building a road map for next-generation LiDAR technology," *IEEE Consum. Electron. Mag.*, vol. 5, no. 3, pp. 48–54, Jul. 2016.

[5] H. Rohling, F. Folster, and H. Ritter, "Lateral velocity estimation for automotive radar applications," in *Proc. IET Int. Conf. Radar Syst.*, 2007, pp. 1–4.

[6] D. Kellner, M. Barjenbruch, K. Dietmayer, J. Klappstein, and J. Dickmann, "Instantaneous lateral velocity estimation of a vehicle using Doppler radar," in *Proc. 16th Int. Conf. Inf. Fusion*, Jul. 2013, pp. 877–884.

[7] 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, doi: 10.1145/358669.358692.

[8] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous full-motion estimation of arbitrary objects using dual Doppler radar," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2014, pp. 324–329.

[9] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Tracking of extended objects with high-resolution Doppler radar," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 5, pp. 1341–1353, May 2016.

[10] 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.

[11] N. B. Gosala and X. Meng, "An RLS-based instantaneous velocity estimator for extended radar tracking," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 2273–2280.

[12] J. Ru and C. Xu, "Improvement on velocity estimation of an extended object," in *Proc. 20th Int. Conf. Inf. Fusion (Fusion)*, Jul. 2017, pp. 1–7.

[13] J. Schlichenmaier, M. Steiner, T. Grebner, and C. Waldschmidt, "Clustering and subsequent contour and motion estimation of automotive objects using a network of cooperative radar sensors," *IEEE Trans. Veh. Technol.*, vol. 72, no. 1, pp. 428–443, Jan. 2023.

[14] P. Svenningsson, F. Fioranelli, and A. Yarovoy, "Radar-PointGNN: Graph based object recognition for unstructured radar point-cloud data," in *Proc. IEEE Radar Conf. (RadarConf21)*, May 2021, pp. 1–6.

[15] D. Niederlohner et al., "Self-supervised velocity estimation for automotive radar object detection networks," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2022, pp. 352–359.

[16] 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.

[17] 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)*, Nov. 2021, pp. 1–8. [Online]. Available: https://api.semanticscholar.org/CorpusID:233033633

[18] R. Q. Charles, H. Su, M. Kaichun, 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. 77–85.

[19] P. J. Huber, "Robust estimation of a location parameter," *Ann. Math. Stat.*, vol. 35, no. 1, pp. 73–101, 1964, doi: 10.1214/aoms/1177703732.

[20] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 16259–16268.

[21] A. Vaswani et al., "Attention is all you need," 2017, *arXiv:1706.03762*.

[22] MATLAB. (2024). *TrackingIMM Interacting Multiple Model (IMM) Filter for Object Tracking*. [Online]. Available: https://www.mathworks.com/help/fusion/ref/trackingimm.html

[23] A. F. Genovese, "The interacting multiple model algorithm for accurate state estimation of maneuvering targets," *Johns Hopkins Apl Tech. Dig.*, vol. 22, no. 4, pp. 614–623, Jan. 2001. [Online]. Available: https://api.semanticscholar.org/CorpusID:16168268

[24] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," *J. Fluids Eng.*, vol. 83, no. 1, pp. 95–108, Mar. 1961. [Online]. Available: https://api.semanticscholar.org/CorpusID:8141345

[25] 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.

Mujtaba Hassan (Graduate Student Member, IEEE) received the bachelor's degree in electrical engineering from the National University of Science and Technology, Islamabad, Pakistan, in 2016, and the master's degree in communications engineering from the Technical University of Munich, Munich, Germany, in 2019. He is pursuing the Ph.D. degree with the Microwave Sensing, Signals and Systems Group, Delft University of Technology (TU Delft), Delft, The Netherlands.

He completed his master's thesis with NXP Semiconductors, Hamburg, Germany, where he designed an optimized convolutional neural network for dense optical flow estimation. Afterwards, he joined NXP as a Systems and Application Engineer in 2019. During this period, he had investigated machine learning algorithms involved during different stages of ADAS as well as their requirements for implementation on hardware accelerators. Currently, he is focusing on neural networks together with their requirements relevant to radar perception in ADAS.

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

He is currently an Associate Professor at Delft University of Technology (TU Delft), Delft, The Netherlands, and 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 has authored over 190 peer-reviewed publications, and edited the books on *Micro-Doppler Radar and Its Applications* and *Radar Countermeasures for Unmanned Aerial Vehicles* (IET-Scitech, 2020). 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.

Dr. Fioranelli received four Best Paper Awards and IEEE AESS Fred Nathanson Memorial Radar Award 2024.

Alexander Yarovoy (Fellow, IEEE) received the Diploma (Hons.) degree in radiophysics and electronics, the Ph.D. degree in physical and mathematical sciences, and the Doctor of Sciences degree in radiophysics from Kharkov State University, Kharkiv, Ukraine, in 1984, 1987, and 1994, respectively.

In 1987, he joined the Department of Radiophysics at the Kharkov State University as a Researcher and became a Full Professor in 1997. From September 1994 to 1996, he was with Ilmenau University of Technology, Ilmenau, Germany, as a Visiting Researcher. Since 1999, he has been with the Delft University of Technology (TU Delft), Delft, The Netherlands, where he has been the Chair of Microwave Sensing, Systems and Signals since 2009. He has authored and co-authored more than 500 scientific or technical papers, seven patents, and 14 book chapters. His main research interests are in high-resolution radar, microwave imaging, and applied electromagnetics (in particular, UWB antennas).

Dr. Yarovoy was a recipient of the European Microwave Week Radar Award for the paper that 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 2010, he received the Best Paper Award of the Applied Computational Electromagnetic Society (ACES), together with D. Caratelli. He served as the General TPC Chair for the 2020 European Microwave Week (EuMW'20), the Chair and 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). He serves as an Associate Editor for IEEE TRANSACTION ON RADAR SYSTEMS. From 2011 to 2018, he served as an Associate Editor for the *International Journal of Microwave and Wireless Technologies*. In the period 2008 to 2017, he served as the Director of the European Microwave Association (EuMA).

Satish Ravindran has around 12 years of experience developing AI solutions for different industries such as Autonomous Driving, Intelligent Traffic Sensing (ITS), and IoT. He has worked on a wide spectrum of applications in AI, including NLP, computer vision, and radar processing. He joined NXP in 2018 and is currently the AI Technical Lead for Radar Innovations working in the NXP R&D Division, Austin, TX, USA. 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.

Luihi Chen received the B.Sc. degree in applied mathematics from the 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 the research of automotive ADAS products, including surround vision, driver monitoring, and HD mapping. He authored and co-authored 13 patents, with five systems delivered for production. He joined NXP's R&D Division, Austin, TX, USA, in 2022 as the Radar Perception Project Lead. His research interests include AI solutions for perceptions with different sensors, and neutral network optimization for edge devices.