Hierarchical Architecture and Feature Mixing for Ego-Motion Estimation using Automotive Radar

Zhu, Simin; Fioranelli, Francesco; Yarovoy, Alexander; Ravindran, Satish; Chen, Lihui

Publication date
2024

Document Version
Final published version

Published in
ICMIM 2024

Citation (APA)
Zhu, S., Fioranelli, F., Yarovoy, A., Ravindran, S., & Chen, L. (2024). Hierarchical Architecture and Feature Mixing for Ego-Motion Estimation using Automotive Radar. In *ICMIM 2024: International Conference on Microwaves for Intelligent Mobility - 7th IEEE MTT Conference* (pp. 99-102). (ICMIM 2024: International Conference on Microwaves for Intelligent Mobility - 7th IEEE MTT Conference). VDE Verlag GMBH.

Hierarchical Architecture and Feature Mixing for Ego-Motion Estimation using Automotive Radar

Simin Zhu[a], Francesco Fioranelli[a], Alexander Yarovoy[a], Satish Ravindran[b], Lihui Chen[b]
[a] Microwave Sensing Signals & Systems (MS3) Group, Delft University of Technology, Delft, The Netherlands
[b] NXP Semiconductors, San Jose, California, United States of America

Abstract

This paper focuses on the challenge of estimating the 2D instantaneous ego-motion of vehicles equipped with an automotive radar. To further improve our previous study based on the weighted least squares (wLSQ) method and purpose-designed neural networks (NNs), this work proposes a new network architecture that supports local and global feature extraction as well as point-wise dynamic feature channel mixing. Compared with our previous work, the proposed method provides better estimation accuracy, lighter network size, and faster runtime performance.

1 Introduction

Ego-motion estimation for vehicles stands out as a crucial stage for contemporary autonomous vehicles[1]. This provides the current speed of the vehicle, which directly informs motion control strategies. Furthermore, it significantly impacts the effectiveness of various subsequent applications, e.g., mapping, object tracking, and path planning. Traditionally, vehicle ego-motion can be monitored via odometry sensors such as wheel encoders, inertial measurement units (IMU), and global positioning systems (GPS). However, they are not always reliable [2] and more data redundancy is needed from other sensing modalities. Many different sensors can be used for ego-motion estimation, such as stereo camera, LiDAR, and sonar. Compared with them, automotive radar has incomparable advantages: it can operate in extreme weather conditions, is less sensitive to lighting conditions unlike cameras, and can illuminate targets not in the direct line of sight.

Owing to these benefits, numerous approaches have been proposed to leverage automotive radars for ego-motion estimation. In general, these previous approaches can be divided into two groups. The first category is known as the scan-matching methods [3], originally designed to solve the ego-motion estimation problem with LiDAR data [4], some of which have then been adapted for processing radar data [5]. One of the advantages of these methods is that they can use a single radar to estimate the complete two-dimensional (2D) motion of the vehicle, including lateral, longitudinal, and rotational velocities [6]. However, since scan-matching is based on data association, these methods often require sensors with high angular resolution and stable object detection capabilities.

The second type of ego-motion estimation methods are called instantaneous approaches [7]. They exploit the relationship between the vehicle ego-motion, Doppler frequency, and angle-of-arrival (AoA) [8]. Thus, they estimate ego-motion with a single radar frame, without the need for data association. However, these methods cannot exploit other object features captured by automotive radar and have poor runtime performance due to the internal iterative process. To address these gaps, our recent work [9] proposes to use neural networks (NNs)-based weighted least squares (wLSQ) for instantaneous ego-motion estimation. This uses NNs to process multidimensional radar point clouds to directly estimate the vehicle motion without iterations. Nevertheless, limited by its network architecture, this method is constrained to extracting only global spatial features and offers limited support for information sharing among points. However, these functionalities are crucial, as the task handled by the NN-based wLSQ is close to a segmentation problem. Therefore, to go a step further, this work proposes a new network architecture for hierarchical feature extraction and dynamic channel mixing. Specifically, it allows progressive feature extraction at multiple scales and supports mixing point feature channels dynamically. Compared with the previous work [9], the proposed method improves both estimation accuracy and runtime performance.

The rest of the paper is structured as follows. **Section 2** delves into a detailed explanation of the problem statement and the proposed solution. **Section 3** presents the evaluation results for the proposed method. Ultimately, **Section 4** concludes the paper.

2 Methodology

2.1 Problem Description

This paper focuses on solving the problem of estimating the vehicle ego-motion in a 2D plane based only on measurements from an automotive radar. Compared to our previous study [9], this work investigates the same problem and uses a similar methodology (i.e., the NN-based wLSQ approach), but with a special focus on the architecture design of the NN.

For the n_{th} radar on a vehicle, its measurement after detection is a matrix of size $J \times M$, where J is the number of detection points and M is the number of features. Two important object features are the radial velocity and AoA; for the j_{th} detection, they are denoted as d_j^n and α_j^n.

ISBN: 978-3-8007-6363-4

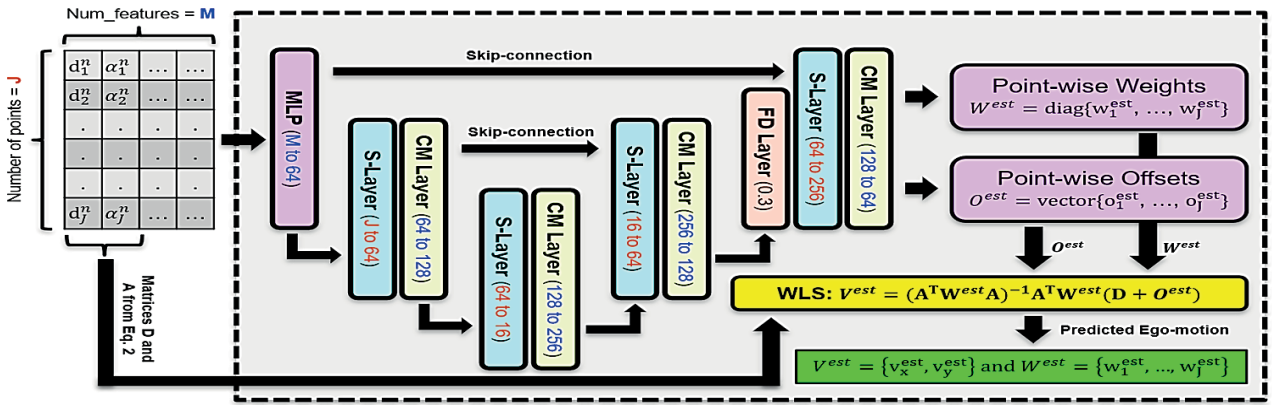

Figure 1 The architecture of the proposed method. The input radar point cloud is a $J \times M$ matrix. The network consists of 4 S-Layer, 4 CM-Layer, 1 FD-Layer, and 1 MLP layer. Based on the extracted features, it predicts point-wise weights (W^{est}) and offsets (O^{est}) for computing wLSQ. The network outputs the ego-motion of the radar platform (V^{est}) which can be easily converted into the self-motion of the vehicle.

Therefore, the ego-motion $\mathrm{V}^{\mathrm{est}}$ of the n_{th} radar can be expressed as:

$$\begin{bmatrix} \mathrm{d}_1^{\mathrm{n}} \\ \mathrm{d}_2^{\mathrm{n}} \\ \dots \\ \mathrm{d}_{\mathrm{J}}^{\mathrm{n}} \end{bmatrix} = - \begin{bmatrix} \cos(\alpha_1^{\mathrm{n}}) & \sin(\alpha_1^{\mathrm{n}}) \\ \cos(\alpha_2^{\mathrm{n}}) & \sin(\alpha_2^{\mathrm{n}}) \\ \dots & \dots \\ \cos(\alpha_{\mathrm{J}}^{\mathrm{n}}) & \sin(\alpha_{\mathrm{J}}^{\mathrm{n}}) \end{bmatrix} \begin{bmatrix} \mathrm{v}_{\mathrm{x}}^{\mathrm{n}} \\ \mathrm{v}_{\mathrm{y}}^{\mathrm{n}} \end{bmatrix} \tag{1}$$

Or in the matrix form:

$$\mathrm{D} = \mathrm{A} \cdot \mathrm{V}^{\mathrm{est}} \tag{2}$$

Although one can use regression methods such as ordinary least squares (OLS) to estimate $\mathrm{V}^{\mathrm{est}}$, the performance will be severely influenced by outliers such as moving objects, false alarms, and multi-path reflections. To solve this issue, this work uses point weights $\mathrm{W}^{\mathrm{est}}$ estimated by NNs together with the wLSQ method, as:

$$\mathrm{V}^{\mathrm{est}} = (\mathrm{A}^{\mathrm{T}}\mathrm{W}^{\mathrm{est}}\mathrm{A})^{-1}\mathrm{A}^{\mathrm{T}}\mathrm{W}^{\mathrm{est}}\mathrm{D} \tag{3}$$

$\mathrm{V}^{\mathrm{est}} = \{\mathrm{v}_{\mathrm{x}}^{\mathrm{est}}, \mathrm{v}_{\mathrm{y}}^{\mathrm{est}}\}$ is the ego-motion of the radar, and the vehicle self-motion can be computed easily by using the relative position between the radar and vehicle.

Finally, it is important to acknowledge that this work assumes zero lateral velocity ($\mathrm{v}_{\mathrm{y}}^{\mathrm{car}} = 0$ m/s). However, this is a common assumption in the literature [7], [10], [11] and can be lifted when multiple radars are used [12].

2.2 Proposed Solution

Overview. **Figure** 1 shows the architecture of the proposed method. The key idea behind the proposed approach is to first use a neural network to extract point features from the input radar point cloud. Then, these extracted features are used to estimate the weights for each point. However, unlike in our previous work [9], this new approach proposes a different hierarchical architecture and a feature mixing scheme, which allows the network to progressively aggregate critical features at local and global scales. The proposed model consists of three basic components, namely sampling layer (S-Layer), channel mixing layer (CM Layer), and feature dropout layer (FD-Layer).

S-Layer. The main goal of the S-Layer is to upsample and downsample the input point cloud to build a hierarchical path for feature extraction. As shown in **Figure** 1, the proposed method first downsamples the number of points in the input point cloud, and then upsamples it back to the input size. As studied in [13], [14], the sampling mechanism provides an efficient way to achieve a large receptive field for each point in the point cloud, which is important for identifying outliers and assigning them appropriate weights. Moreover, unlike [9], which uses a simple pooling layer to aggregate the entire point cloud, the hierarchical sampling architecture can help the feature extraction process capture local structures (e.g., slow-moving objects). Finally, S-Layer uses the Farthest Point Sampling (FPS) algorithm for point sampling and skip-connections to propagate fine-grained features learned in earlier layers [15].

CM-Layer. There are two operations performed by the CM-Layer. First, CM-Layer implements point-wise feature encoding and decoding during the down-sampling and up-sampling stages, respectively. As shown in **Figure** 1, the number of point features increases during downsampling (from 64 to 256) and decreases during upsampling (from 256 to 64). Feature encoding and decoding are achieved using multi-layer perceptron (MLP) [16]. Second, the CM-Layer updates the features of each point in the next layer (i.e., anchor point) by using their adjacent points in the previous layer. By doing this, each point can aggregate features from its neighbours. Additionally, the idea of dynamic graph attention [17] is used so that anchor points can learn to aggregate features from their most relevant neighbours. In summary, the CM-Layer changes the feature dimension of point clouds (point by point) and mixes point features (channel by channel).

FD-Layer. The main objective of the FD-Layer is to prevent the network from over-relying on a few point features for point weight prediction. Unlike traditional dropout [18], FD-Layer randomly selects a set of features and mutes them throughout the point cloud. Therefore, the network is forced to spread its focus over a larger set of point features than without using the FD-Layer.

2.3 Implementation Details

As mentioned earlier, the input radar point cloud is assumed to be a J × M matrix. First, a MLP projects the input

ISBN: 978-3-8007-6363-4

point features to a higher feature space (from M to 64). Then, the S-Layer gradually down-samples the input point cloud to different subsets, from J to 16 using the FPS algorithm. For up-sampling, instead of resampling, the same subsets generated during downsampling are used, from 16 back to J. A CM-Layer is always located after the S-Layer and it uses FPS and K-nearest neighbors (KNN) to find 3 nearest neighbor points and 2 far points. Lastly, FD-Layer randomly sets point features to 0 with a frequency of 0.3. Finally, it is important to note that the point-wise weight and offset predictions, the wLSQ implementation, and loss functions are the same as in the previous work [9], and readers are referred to this for more details.

3 Results

3.1 Dataset and Evaluation

For performance evaluation, the RadarScenes dataset [19] is used. The evaluation data are five radar recordings captured by a forward-looking radar, containing different road types such as collector roads, local streets, and arterial roads. The same evaluation data are used as in [9]; it is worth mentioning that no evaluation data was used during model training. For more details on the dataset and evaluation setup, please refer to [19] and [9].

3.2 Performance Comparison

Estimation Accuracy. As shown in **Table** 1, the proposed method outperforms the original DeepEgo [9] in terms of translational and rotational velocity estimation. Furthermore, since the proposed method uses a smaller maximum feature length (256) than DeepEgo (512), the performance of DeepEgo with the same feature length is also measured. It can be seen that as the feature length is shortened, the performance of DeepEgo decreases. However, thanks to the hierarchical architecture, the proposed method can have a lower network complexity and better estimation accuracy. Additionally, an FD-Layer is added to DeepEgo, but the measured performance improvement is not significant.

Methods	RMSE Vx (m/s)	RMSE Rot. (deg/s)
DeepEgo [9]	0.0876	0.911
DeepEgo*	0.0896	0.959
DeepEgo + FD-Layer	0.0852	0.879
Proposed	0.0864	0.829
Improvement	+1.4%	+9.0%

Table 1 Root-mean-square-error in translational (Vx) and rotational (Rot.) velocity estimation. The proposed method is compared to the original DeepEgo [9], and two of its variants. DeepEgo* uses a maximum feature length of 256 instead of 512, to match the same network complexity of the proposed method. Another variant adds FD-Layer to DeepEgo (i.e., DeepEgo + FD-Layer). Results are averaged over 5 radar recordings from the evaluation data.

Network Complexity. One of the main challenges when considering deploying NNs in real-world scenarios is the limited memory size. Not to mention that complex networks are data-hungry, and radar data is not easily available. Therefore, even with (somewhat) reduced performance, a lighter network is often desired. As shown in **Table** 2, the proposed method has a smaller number of trainable parameters compared with DeepEgo (4.7 × lighter). Yet, the proposed method still performs better in ego-motion estimation as shown in **Table** 1. Due to the lightweight network, the running time of the proposed method is approximately 2.2 × faster than the previous work.

Methods	Trainable Parameters	Runtime (FPS)
DeepEgo [9]	~800K	~243.8
Proposed	~170K	~547.4
Improvement	4.7x lighter	2.2x faster

Table 2 Network complexity comparison. Trainable parameters are the trainable biases and weights in a neural network. The runtime performance is measured by frames per second (FPS) using the Delft High Performance Computing Center (DHPC).

3.3 Visualization of Attention Weights

Figure 2 shows the down-sampling (first row) and up-sampling (second row) processes and the learned attention weights for each anchor point. It is easy to see from the figure that, regardless of whether the anchor point is an inlier or an outlier, it will try to assign high weights to neighbouring points that are inliers. This is reasonable because not only the relative vehicle motion is encoded in these inliers (detected stationary objects), but the information of inliers can help better distinguish outliers.

4 Conclusion and Future Work

This work proposed a novel neural network with a hierarchical architecture and channel mixing mechanism for radar-based vehicle ego-motion estimation. Unlike previous works, the proposed method can capture local and spatial features gradually and form a large receptive field on the top layer. Moreover, it can automatically learn attention weighs and aggregate point features from informative neighbours. Furthermore, tested on a challenging real-world dataset, the proposed method shows higher estimation accuracy compared with previous work despite lower network complexity. For future learning directions, it is important to consider a more general case where a moving vehicle is equipped with an unsynchronized radar sensor network and how to fuse information from different perspectives to achieve better estimation accuracy.

5 Literature

[1] Ranga, Adithya, et al. "Vrunet: Multi-task learning model for intent prediction of vulnerable road users." arXiv preprint arXiv:2007.05397 (2020).

ISBN: 978-3-8007-6363-4

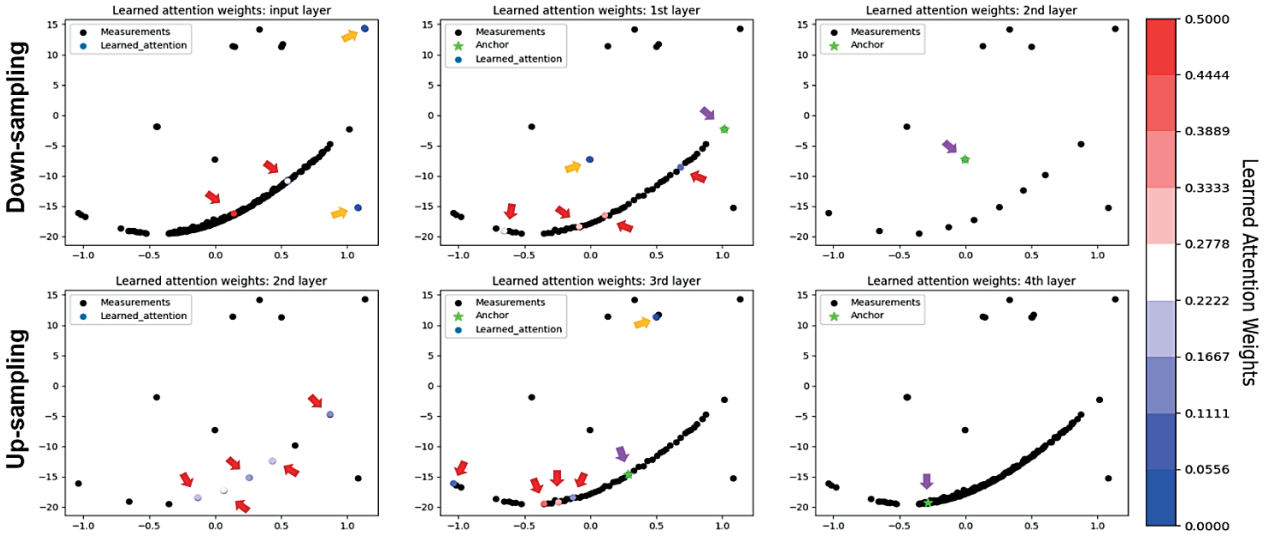

Figure 2 A sketch of the learned attention weights in each layer. The figures in the 1st row show the down-sampling path while the 2nd row shows the up-sampling path. Examples of anchor points are represented by green stars and pointed by purple arrows. Neighbouring points of the anchor point in the previous layer are pointed by red arrows (inliers) or orange arrows (outliers). The colour of neighbouring points indicates their attentional weights relative to their anchor points.

[2] Gu, Yanlei, et al. "Vehicle self-localization in urban canyon using 3D map based GPS positioning and vehicle sensors." 2014 International Conference on Connected Vehicles and Expo (ICCVE). IEEE, 2014.

[3] Adams, Martin, and Martin David Adams. Robotic navigation and mapping with radar. Artech House, 2012.

[4] Lu, Feng, and Evangelos Milios. "Robot pose estimation in unknown environments by matching 2d range scans." Journal of Intelligent and Robotic systems 18 (1997): 249-275.

[5] Li, Wei, et al. "Indoor Positioning System Using a Single-Chip Millimeter Wave Radar." IEEE Sensors Journal 23.5 (2023): 5232-5242.

[6] Rapp, Matthias, et al. "A fast probabilistic ego-motion estimation framework for radar." 2015 European Conference on Mobile Robots (ECMR). IEEE, 2015.

[7] Kellner, Dominik, et al. "Instantaneous ego-motion estimation using Doppler radar." 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013). IEEE, 2013.

[8] Lim, Sohee, et al. "Radar-Based Ego-Motion Estimation of Autonomous Robot for Simultaneous Localization and Mapping." IEEE Sensors Journal 21.19 (2021): 21791-21797.

[9] Zhu, Simin, Alexander Yarovoy, and Francesco Fioranelli. "DeepEgo: Deep Instantaneous Ego-motion Estimation using Automotive Radar." IEEE Transactions on Radar Systems (2023).

[10] Rapp, Matthias, et al. "Probabilistic ego-motion estimation using multiple automotive radar sensors." Robotics and Autonomous Systems 89 (2017): 136-146.

[11] Kung, Pou-Chun, Chieh-Chih Wang, and Wen-Chieh Lin. "A normal distribution transform-based radar odometry designed for scanning and automotive radars." 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021.

[12] Kellner, Dominik, et al. "Instantaneous ego-motion estimation using multiple Doppler radars." 2014 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2014.

[13] Qi, Charles Ruizhongtai, et al. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space." Advances in neural information processing systems 30 (2017).

[14] Ronneberger, Olaf, Philipp Fischer, Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." Medical Image Computing & Computer-Assisted Intervention–MICCAI, 18th International Conference, Munich, Oct 5-9 2015, Proceedings, Part III 18. Springer International,.

[15] Zhao, Hengshuang, et al. "Point transformer." Proceedings of the IEEE/CVF international conference on computer vision. 2021.

[16] Qi, Charles R., et al. "Pointnet: Deep learning on point sets for 3d classification and segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[17] Brody, Shaked, Uri Alon, and Eran Yahav. "How attentive are graph attention networks?." arXiv preprint arXiv:2105.14491 (2021).

[18] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." The journal of machine learning research 15.1 (2014): 1929-1958.

[19] Schumann, Ole, et al. "RadarScenes: A real-world radar point cloud data set for automotive applications." 2021 IEEE 24th International Conference on Information Fusion (FUSION). IEEE, 2021.

ISBN: 978-3-8007-6363-4