Simulating Automotive Radar with Lidar and Camera Inputs

Peili Song[1], Dezhen Song[2], Yifan Yang[1], Enfan Lan[1], and Jingtai Liu[1*]

***Abstract*— Low-cost millimeter automotive radar has received more and more attention due to its ability to handle adverse weather and lighting conditions in autonomous driving. However, the lack of quality datasets hinders research and development. We report a new method that is able to simulate 4D millimeter wave radar signals including pitch, yaw, range, and Doppler velocity along with radar signal strength (RSS) using camera image, light detection and ranging (lidar) point cloud, and ego-velocity. The method is based on two new neural networks: 1) DIS-Net, which estimates the spatial distribution and number of radar signals, and 2) RSS-Net, which predicts the RSS of the signal based on appearance and geometric information. We have implemented and tested our method using open datasets from 3 different models of commercial automotive radar. The experimental results show that our method can successfully generate high-fidelity radar signals. Moreover, we have trained a popular object detection neural network with data augmented by our synthesized radar. The network outperforms the counterpart trained only on raw radar data, a promising result to facilitate future radar-based research and development.**

I. INTRODUCTION

Low-cost automotive radars work at millimeter wave (mmWave) length and are widely used in autonomous driving due to its effectiveness under adverse weather and lighting conditions. During foggy, stormy, and smoky days, it is the last resort that can ensure the safety of vehicles on the road. However, quality datasets are limited due to existing research that mainly focuses on the use of regular camera and light detection and ranging (lidar) modalities, which hinders the development of radar-based navigation algorithms. To address these issues, we propose using existing camera and lidar datasets, which are widely available, to simulate radar signal strength (RSS) and 4D radar signals, including pitch, yaw, depth, and Doppler velocity without the need of the prior 3D scene model or knowledge about material type.

However, synthesizing radar points using images and lidar point colors is non-trivial due to signal consistency, large depth variations, large point dispersion, and cross-modality sensitivity difference, as shown in Fig. 1. To address these issues, we employ learning-based approaches. First, we estimate radar signal distribution and number of reflections using a custom designed network that takes object appearance and radar ego-velocity as input. The signal RSS values are generated for each point by fusing the multimodal information using a custom designed neural network which utilizes the information about local appearance and geometric shape to predict RSS without the need of precise 3D scene/material models or complex radiation pattern computation. Both networks can be easily trained using datasets with real radar signals. We have implemented the proposed algorithm and tested it using open datasets from 3 different commercial radar models. The ablation study results have shown that our design is successful. Signal distribution and quality are statistically indistinguishable from raw radar signals. Moreover, we have trained an object detection neural network with data augmented by our synthesized radar. The network outperforms the counterpart that only trained on raw radar data, a promising result for future radar-based research.

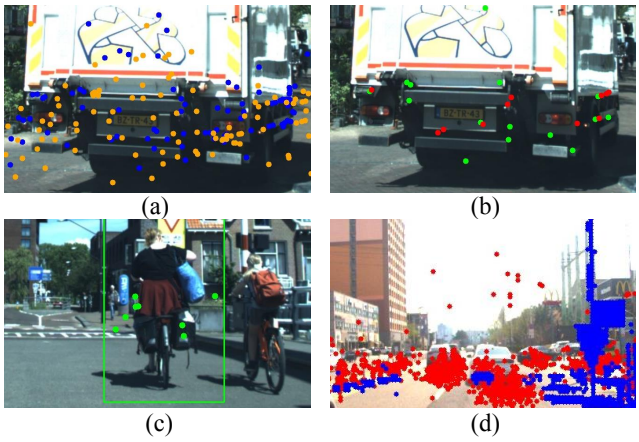


Fig. 1: Properties of mmWave radar point cloud and the associated challenges of data synthesis: (a) Signal inconsistency: two radar point clouds (brown and blue) from the same perspective do not have the same point positions. (b) Large depth variations: Green points are where radar points agrees lidar points in depth, red points otherwise. (c) Large point dispersion: radar signals reflected by a certain object may "float" around it. (d) Cross-modality sensitivity difference: radar (red) and lidar (blue) point cloud possess different reflection sensitivity.

II. RELATED WORKS

Simulating radar datagram from camera images and lidar point clouds relates to the working principle of millimeter-wave radar, its applications in autonomous driving, and radar simulation techniques.

*This work is supported by the National Natural Science Foundation of China under Grant 62173189.

[1]Peili Song, Yifan Yang, Enfan Lan and Jingtai Liu are with the Institute of Robotics and Automatic Information System, Nankai University, Tianjin 300350, China; Tianjin Key Laboratory of Intelligent Robotics, Tianjin 300350, China; and also with TBI center, Nankai University, Tianjin 300350, China (e-mail: {peilisong,yangyifan,lef}@mail.nankai.edu.cn; liujt@nankai.edu.cn).

[2]Dezhen Song is with the Department of Robotics, Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI), Abu Dhabi, United Arab Emirates (e-mail:dezhen.song@mbzuai.ac.ae).

*Corresponding author

Manuscript 2763 submitted to 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 1, 2025.

MmWave radar has radio signal wavelengths between 1.0mm and 10.0mm. It transmits and receives electromagnetic waves to obtain measures. The RSS of the signal depends on the material, size, and structure of the reflection object. Typically, it is measured by radar cross section (RCS), which is equivalent to the cross-sectional area of a hypothetical sphere that is generated to measure the reflectivity of the target object [1]. However, a low-cost automotive radar may/may not use RCS, but reports its own measured signal strength to characterize RSS in a nonstandard way. Therefore, in the paper, we will use the RSS as a broader measure instead of the RCS. 4D Mmwave radar plays a key role in autonomous driving systems due to its all-weather performance, high-precision range and velocity measurement, penetration capability, and resistance to interference [2]. 4D mmwave radar data contains range, yaw, pitch, and Doppler velocity. Currently, the number of 4D radar datasets is very limited [3], [4], [5], [6], which limits the development of radar-based navigation.

Due to the lack of available radar datasets, methods of simulating radar data have been explored. CARLA [7], a well-known autonomous driving simulation platform, can only simulate radar data with very low fidelity. Vira [8] is based on Unity game engine using its built-in rendering pipeline. 4D radar data in [9], a generic dataset, is generated by WaveFarer. To reduce the computational overhead of electromagnetic field simulations, a radar simulation framework based on a graphic simulation program, Blender, [10] has been proposed. There are also a few data-driven methods using generative models to generate radio-frequency (RF) data [11], [12] for certain human actions in static environments. These existing methods are either very limited in fidelity or have to rely on the computing radiation pattern that requires detailed knowledge such as 3D scene models and materials of particular targets, and frequency and antenna design of the simulated radar, all of which are often not available in practice. Our approach is the first of the kind to completely simulate 4D mmWave radar signals with radar signal strength information.

III. Problem Formulation

Input, Output, and Coordinate Systems: At any given frame, we want to simulate a typical automotive radar using ego-velocity, a lidar point cloud, and a camera image. The coordinate systems are shown in Fig. 2.

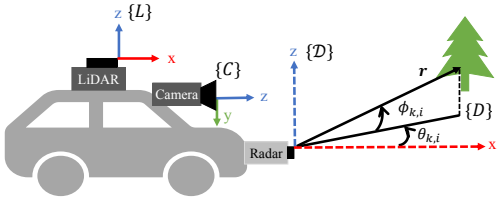


Fig. 2: Coordinate systems illustration.

According to convention, a radar coordinate system $\{D\}$ is a spherical coordinate system. The corresponding 3D Cartesian coordinate system $\{\mathcal{D}\}$ is established with its origin coinciding with $\{D\}$, X-axis pointing forward, and Z-axis pointing vertically upwards. The lidar coordinate system $\{L\}$ and camera coordinate system $\{C\}$ are established as is shown in 2. All 3D Cartesian systems are right-handed.

The output radar datagram is defined as $\mathbf{D} := \{\mathbf{s}_{i=1:n}\}$ where i is the data index and n is the maximum number of radar data points. Each reflected signal is in the spherical coordinates of $\{\mathbf{D}\}$ and contains:

$$\mathbf{s}_i = [r_i, \theta_i, \phi_i, v_i, a_{\text{RSS},i}]^\mathsf{T}, \tag{1}$$

where r_i is the range/depth, v_i is the Doppler velocity, θ_i and ϕ_i are the azimuth and elevation (i.e. pitch and yaw) angles of the i-th signal, respectively, and $a_{\text{RSS},i}$ is the RSS value. Radars with such 5-element datagrams are often referred to as a 4D radar because it has first four elements in (1) describing spatial and motion information.

Assumption: Appearances of targets contain sufficient information of material and shape, which can determine the reflectivity of radar signals.

Based on the assumption, with camera images providing appearance information such as shape and color, and lidar point clouds providing range/depth information, there are sufficient inputs for radar signal simulation.

Nomenclature: Let us define the important notations before introducing our problem.

$\mathbf{I}$	the camera image ,
$\mathbb{I}, \{I\}$	the 2D image space $\mathbb{I} \subset \mathbb{R}^2$ and the associated coordinate $\{I\}$ with pixel coordinates $[u, v]^\mathsf{T}$ and origin located at top-left corner of the image, u-axis pointing leftward, and v-axis pointing downward,
$\mathbf{l}_j$	The j-th lidar point denoted as $\mathbf{l}_j = [x_j, y_j, z_j]^\mathsf{T} \in \mathbb{R}^3$,
$\mathbf{L}$	lidar point cloud: $\mathbf{L} := \{\mathbf{l}_{j=1:n_l}\}$, n_l is the maximum number of lidar points,
${}^{\mathcal{D}}_{L}T$	4×4 homogeneous coordinate transformation matrix from $\{L\}$ to $\{\mathcal{D}\}$,
${}^{\mathcal{D}}_{C}T$	4×4 homogeneous coordinate transformation matrix from $\{C\}$ to $\{\mathcal{D}\}$.

Our problem is defined as follows.

Definition 1: Given camera image $\mathbf{I}$, lidar point cloud $\mathbf{L}$, transformation matrices ${}^{\mathcal{D}}_{L}T$, ${}^{\mathcal{D}}_{C}T$, and radar's ego velocity ${}^{\mathcal{D}}\mathbf{v}_D$, generate corresponding radar datagram $\mathbf{D}$.

It is worth noting that we are not attempting to generate raw radar signal, but to simulate a commercial low cost mmWave radar output which is usually much sparse after signal filtering, as shown in Fig. 1.

IV. Algorithms

Fig. 3 illustrates our algorithm pipeline which is consistent of three modules shaded in light blue. Box 4.1 predicts radar signal distribution in the image frame and number of radar signals. Based on that, we generate 4D signals in Box 4.2. To complete the signal, RSS is generated in the last module. Let us explain them in sequence.

Manuscript 2763 submitted to 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 1, 2025.

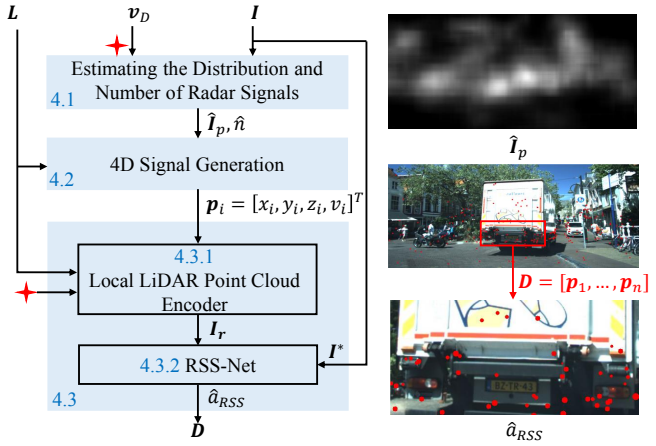


Fig. 3: Algorithm pipeline for generating simulated radar signals (inference stage). The algorithm takes lidar point cloud, camera image, and radar ego-velocty as input. The grayscale image represents the predicted radar signal distribution with high insensitivity representing high probability. The size of the red dot in the lower right image indicates the predicted RSS.

A. Estimating the Distribution and Number of Radar Signals

To simulate radar datagram, the first key issue is to determine where radar signals would be reflected and how many reflections would occur. As is shown in Fig. 1 (a) and (c), a radar signal often disperse around the target due to mmWave radio signal refraction, reflection and scattering. This is very different from optical sensors at shorter wavelength. To address the issue, we design a Dis-Net to estimate the probability density function (PDF) and the total number of signals.

1) Dis-Net Architecture: Fig. 4 illustrates the architecture of Dis-Net which has a two-branch structure: a distribution branch to estimate the probability distribution and a number branch to estimate the number of radar signals.

Let us first define the image input to be

$$\mathbf{I} := \{(r,g,b)_{(u,v)} \big| u \in [1, u_{\max}] \cap \mathbb{N}, v \in [1, v_{\max}] \cap \mathbb{N}\}, \quad (2)$$

where $(r,g,b) \in [0,255] \times [0,255] \times [0,255]$ are the red, green, and blue intensity values, respectively, u and v are pixel coordinates, and $u_{\max}$ and $v_{\max}$ are maximum horizontal and vertical pixel indices, respectively.

To estimate radar signal distribution probability density in the image frame (represented as a grayscale image) $\mathbf{I}_p$, ResNet-18 [13] is used as the backbone to encode $\mathbf{I}$ to be a high-dimensional feature map $\mathbf{F}$ containing appearance information such as color, texture, and semantic information. $\mathbf{F}$ is processed by a transpose convolutional network to restore the original image's size and reduce the number of channels to 1. An upsample module is used to ensure the output $\hat{\mathbf{I}}_p$ has the same size as $\mathbf{I}_p$.

The number branch predicts the number of reflections which requires the ego-velocity of the radar in addition to the camera image. Most objects has non-zero Doppler velocity as the radar moves, and the signals with significant Doppler velocity are likely to be retained after filtering [14] in radar signal processing algorithm. Therefore, only the magnitude of $\|^{\mathcal{D}}\mathbf{v}_D\|_2$ is utilized. Feature map $\mathbf{F}$ and $\|^{\mathcal{D}}\mathbf{v}_D\|_2$ are each encoded by a fully connected (FC) layer with the same output size. They are concatenated and processed by the last FC layer to estimate $\hat{n}$, the number of signals. At the end of both branches, the Sigmoid function is employed to map the output values to the range $(0, 1)$.

2) Dis-Net Training Scheme: To train Dis-Net, we employ real radar signals as ground truth. Because Dis-Net outputs the spatial signal distribution, it is necessary to generate the ground-truth radar signal distribution based on real radar readings. The first step is to project the radar signal from its polar coordinate to the image coordinate $\{I\}$. This is performed by first transferring signal $\mathbf{s}_i$ in spherical coordinate to the collocated Cartesian coordinate,

$$^{\mathcal{D}}\mathbf{p}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} r_i \cdot \cos(\phi_i) \cdot \cos(\theta_i) \\ r_i \cdot \cos(\phi_i) \cdot \sin(\theta_i) \\ r_i \cdot \sin(\phi_i) \end{bmatrix}, \quad (3)$$

where $^{\mathcal{D}}\mathbf{p}_i$ is the corresponding coordinate in $\{\mathcal{D}\}$, which is then converted to coordinate $\{I\}$ using pinhole model [15]

$$\begin{bmatrix} ^I\mathbf{p}_i \\ 1 \end{bmatrix} = \lambda K \begin{bmatrix} ^C_{\mathcal{D}}R & ^C_{\mathcal{D}}\mathbf{t} \end{bmatrix} \begin{bmatrix} ^{\mathcal{D}}\mathbf{p}_i \\ 1 \end{bmatrix}), \quad (4)$$

where K is the intrinsic matrix of the camera, $(^C_{\mathcal{D}}R,\ ^C_{\mathcal{D}}\mathbf{t})$ are the rotation matrix and the translation vector for the coordinate transform from $\{\mathcal{D}\}$ to $\{C\}$, respectively, and λ is the scaling factor. The projection leads to radar datagram $^I\mathbf{D} = \{^I\mathbf{p}_1, ..., ^I\mathbf{p}_n\}$, where $^I\mathbf{p}_i = [u_i, v_i]^\mathsf{T} \in \mathbb{I}$. The observation point set allows us to establish a posterior probability distribution for radar signals using a Gaussian mixture model. For a potential radar signal to be located at $^I\mathbf{x}$, each observation $^I\mathbf{p}_i$ determines the a 2D Gaussian distribution for $^I\mathbf{x}$ with the following probability density function (PDF), which is also known as 2D Bell function,

$$\phi(^I\mathbf{x}; ^I\mathbf{p}_i, \mathbf{\Sigma}) = \quad (5)$$
$$\frac{1}{2\pi}\sqrt{\det(\mathbf{\Sigma})}\exp\Big(-\frac{1}{2}(^I\mathbf{x} - ^I\mathbf{p}_i)^\mathsf{T}\mathbf{\Sigma}^{-1}(^I\mathbf{x} - ^I\mathbf{p}_i)\Big),$$

where variance matrix $\mathbf{\Sigma}$ describes the spatial uncertainty of the signal due to the noisy nature of the radar. The value of $\mathbf{\Sigma}$ can be obtained through statistics of the 2D projection distribution of points from the annotated objects.

When there are many observations, the posterior radar signal distribution is modeled as a Gaussian mixture with the following aggregated PDF,

$$p(^I\mathbf{x}) = \frac{1}{Z}\sum_{i=1}^{n}\phi(^I\mathbf{x}; ^I\mathbf{p}_i, \mathbf{\Sigma}), \quad (6)$$

where and Z is the normalization factor to ensure

$$\Sigma_{^I\mathbf{x}\in\mathbb{I}}\ \ p(^I\mathbf{x}) = 1. \quad (7)$$

The posterior signal distribution can be illustrated in image space $\mathbb{I}$ as $\mathbf{I}_p$ with each pixel (u,v) intensity as $\mathbf{I}_p(u,v) = p([u,v]^\mathsf{T}) \times 255$. Fig. 5 illustrates a sample $\mathbf{I}_p$.

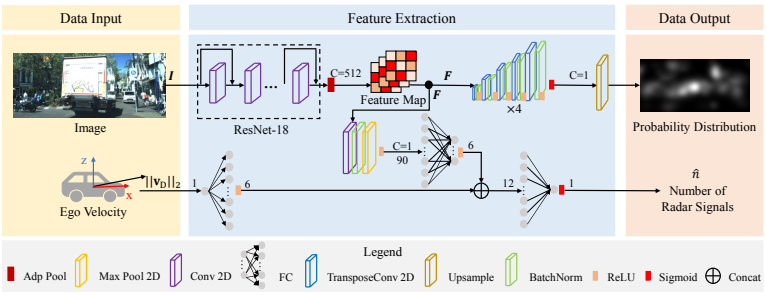


Fig. 4: Overview of Dis-Net. The first branch generates a grayscale image as the probability density function of radar signals' distribution, and the second branch fits the number of signals. Avg Pool refers to average pooling, Adp Pool refers to adaptive pooling, Conv 2D refers to 2D convolutional layer, FC refers to fully connected layer (dense layer), TransposeConv 2D refers to 2D transpose convolutional layer, and Concat refers to concatenation.

Fig. 5: Left: radar signal (red dots) projection on image, and Right: the corresponding signal distribution function illustrated in the grayscale image (translucent).

During training, Dis-Net predicts the radar signal distribution $\hat{\mathbf{I}}_p$ where symbol $\hat{}$ is used to indicate the prediction of the network for the corresponding variable, which is a convention in the rest of the paper. On the other hand, $\mathbf{I}_p$ is used as the ground truth. To compare the similarity between two distributions, we employ Kullback–Leibler (KL) divergence as the loss function of the distribution branch:

$$L_{\text{KL}} = \sum_{{}^{I}\mathbf{x}\in\mathbb{I}} \mathbf{I}_p({}^{I}\mathbf{x}) \cdot \log\left(\frac{\hat{\mathbf{I}}_p({}^{I}\mathbf{x})}{\mathbf{I}_p({}^{I}\mathbf{x})}\right). \tag{8}$$

The second branch is to estimate the number of signals where the ground truth n is the length of real radar datagram $\mathbf{D}$. We propose to use the normalize relative difference as the loss L_n,

$$L_n = \frac{1}{m}\left(\sum_{i=1}^{m} \frac{\hat{n}_i - n_i}{n_i}\right)^2, \tag{9}$$

where m is the number of frames, and i is the frame index. To allow the network simultaneously predict the distribution and the number of signals, we combine total loss of Dis-Net as

$$L = L_{\text{KL}} + \alpha L_n, \tag{10}$$

where α is a relative weighting coefficient between the two branches.

B. 4D Signal Generation

In the inference phase, Dis-Net outputs the radar signal distribution $\hat{\mathbf{I}}_p$ and the signal number $\hat{n}$. Based on them, we can generate 4D radar signals through four steps: 1) signal generation, 2) pitch and yaw generation, 3) range estimation, and 4) the Doppler velocity generation.

Signal Generation: Dis-Net's output $\hat{\mathbf{I}}_p$ and $\hat{n}$ allow us to draw $\hat{n}$ simulated radar signals. Recall that $\hat{\mathbf{I}}_p$ is the signal distribution function in grayscale image format, we can reconstruct the distribution function as follows,

$$p(u, v) = \frac{\hat{\mathbf{I}}_p(u, v)}{\Sigma_{(x,y)\in\mathbb{I}} \hat{\mathbf{I}}_p(x, y)}. \tag{11}$$

Based on the PDF, the task is to generate $\hat{n}$ signals in the image space $\mathbb{I}$ that can be obtained from the inverse transform sampling technique [16] in u- and v- coordinates using its two-step random sampling process. Repeating it for $\hat{n}$ times, the process results in radar signals $\{[u_i, v_i]^{\mathsf{T}}_{i=1:\hat{n}}\} \subset \mathbb{I}$.

Pitch and Yaw Coordinates Generation: For each generated radar signal (u_i, v_i), the corresponding yaw and pitch angles in radar coordinate $\{D\}$ can be obtained as follows,

$$\begin{aligned} \theta_i &= \text{atan2}(y, x), \\ \phi_i &= \arcsin(z, \|[x, y, z]^{\mathsf{T}}\|_2), \end{aligned} \tag{12}$$

where $[x, y, z]^{\mathsf{T}} = {}^{\mathcal{D}}_{C}RK^{-1}[u_i, v_i, 1]^{\mathsf{T}}$ is the back projection of pinhole model in (4) and ${}^{\mathcal{D}}_{C}R = {}^{C}_{\mathcal{D}}R^{-1}$.

Range Estimation: Range information can be estimated using the nearby lidar point range. For any point ${}^{L}\mathbf{p}_j \in \mathbb{R}^3$, it can be transformed to $\mathcal{D}$ as follows

$$[{}^{\mathcal{D}}\mathbf{p}^{\mathsf{T}}, 1]^{\mathsf{T}} = {}^{\mathcal{D}}_{L}T[{}^{L}\mathbf{p}^{\mathsf{T}}, 1]^{\mathsf{T}}, \tag{13}$$

and it can be further transformed to spherical coordinate $\{D\}$ using (12). We denote the transformed lidar point cloud by ${}^{D}\mathbf{L} = \{{}^{D}\mathbf{l}_{j=1:n_l}\}$, where ${}^{D}\mathbf{l}_j = [r_j, \theta_j, \phi_j]^{\mathsf{T}}$ and n_l is the number of lidar points. Thus we can retrieve the nearby local point cloud through

$${}^{D}\mathbf{L}^* = \{{}^{D}\mathbf{l}_j | {}^{D}\mathbf{l}_j \in {}^{D}\mathbf{L}, |\theta_i - \theta_j| \le \delta_1, |\phi_i - \phi_j| \le \delta_2\}, \tag{14}$$

where thresholds δ_1 and δ_2 are the simulated radar's horizontal and vertical angular resolution. Since lidar's angular resolution is higher than that of the radar, ${}^{D}\mathbf{L}^*$ must have $n_Q > 0$ nearby points. The range of radar signal r_i is the

average of that of the nearby points ${}^{D}\mathbf{L}^{*}$,

$$r_i = \frac{\sum_{j=1}^{n_Q} r_j}{n_Q}. \tag{15}$$

Doppler Velocity Generation: Given the velocity of radar and target objects, Doppler velocity is the projection of the velocity vector to the unitary directional vector of the signal through vector dot product $< \cdot >$,

$$v_i = \frac{< ({}^{\mathcal{D}}\mathbf{v}_O - {}^{\mathcal{D}}\mathbf{v}_D) \cdot ({}^{\mathcal{D}}_{C}RK^{-1}[u_i, v_i, 1]^{\mathsf{T}}) >}{\|{}^{\mathcal{D}}_{C}RK^{-1}[u_i, v_i, 1]^{\mathsf{T}}\|_2}, \tag{16}$$

where ${}^{\mathcal{D}}\mathbf{v}_O$ is the object's velocity which can be obtained by using the method in [17] to distinguish signals reflected by background objects or dynamic objects.

C. RSS Estimation

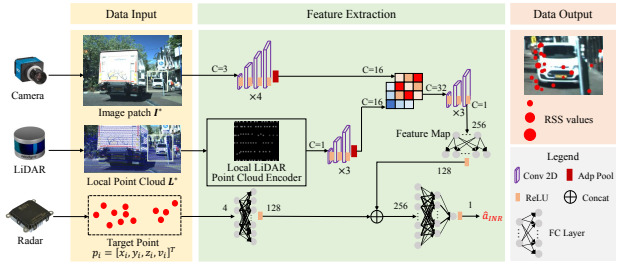


Fig. 6: Overview of RSS-Net. The output is estimated RSS value, $\hat{a}_{\text{RSS}}$, and signals with larger RSS values are illustrated as larger red circles.

The next step is to generate the RSS values. We design an RSS-Net, which is a multimodal neural network that integrates camera image appearance information, the local geometric feature of the lidar point cloud, and the generated 4D radar signals to predict RSS values a_{RSS}. Fig. 6 illustrates the overall neural network design with the three types of input modalities where the 4D radar signals are used as anchor positions to identify the corresponding image patches and nearby lidar points. To facilitate the analysis, we use the 4D signals ${}^{\mathcal{D}}\mathbf{p}_i = [x_i, y_i, z_i, v_i]^{\mathsf{T}}$ in the Cartesian frame $\{\mathcal{D}\}$.

Given a radar signal ${}^{\mathcal{D}}\mathbf{p}_i$, the local image patch is first obtained to capture appearance information of nearby objects. ${}^{\mathcal{D}}\mathbf{p}_i$ can be projected in image coordinate $[u_i, v_i]^{\mathsf{T}}$ using (4). Its neighboring pixel patch is

$$\mathbf{I}^* := \{(u, v) | u \in (u_i - r_c, u_i + r_c], v \in (v_i - r_c, v_i + r_c]\}, \tag{17}$$

where r_c determines the neighboring range.

*1) **Local Point Cloud Encoder**:* Lidar point clouds contains shape information and should be included as RSS-Net input. Again, since only the neighboring lidar points directly affect radar reflectivity, we only focus on those points. First, we convert lidar data to radar coordinate and denote the data as ${}^{\mathcal{D}}\mathbf{L}$ using (13). For each target radar signal ${}^{\mathcal{D}}\mathbf{p}_i$, local lidar point cloud is

$${}^{\mathcal{D}}\mathbf{L}^* := \{\mathbf{p} | \mathbf{p} \in {}^{\mathcal{D}}\mathbf{L}, \|\mathbf{p} - {}^{\mathcal{D}}\mathbf{p}_i\|_2 \leq r_l\}, \tag{18}$$

where r_l is neighboring distance parameter.

To ensure the shape encoding of target objects is not affected by the position of ${}^{\mathcal{D}}\mathbf{p}_i$, the coordinate of any local point is translated and then projected to pixel coordinates (u_j, v_j) as follows,

$$u_j = \left\lfloor \frac{1}{2}(1 - \frac{y_j - y_i}{r_l})w \right\rfloor, v_j = \left\lfloor (1 - \frac{z_j - z_i + r_l}{2r_l})h \right\rfloor, \tag{19}$$

where ${}^{\mathcal{D}}\mathbf{l}_j = [x_j, y_j, z_j]^{\mathsf{T}}$, ${}^{\mathcal{D}}\mathbf{p}_i = [x_i, y_i, z_i]^{\mathsf{T}}$, w and h are the width and height of the grayscale image, respectively, and $u_j \in [0, w)$, $v_j \in [0, h)$. The depth is encoded into grayscale $\mathbf{I}_r$ for all pixels as

$$\mathbf{I}_r := \{r_{(u_j, v_j)}, \forall (u_j, v_j)\}, \tag{20}$$

where

$$r = \begin{cases} 127 + \lceil (\frac{\|{}^{\mathcal{D}}\mathbf{l}_j - {}^{\mathcal{D}}\mathbf{p}_i\|_2}{2r_l}) \cdot 255 \rceil, & \|{}^{\mathcal{D}}\mathbf{l}_j\|_2 \geq \|{}^{\mathcal{D}}\mathbf{p}_i\|_2 \\ 127 - \lfloor (\frac{\|{}^{\mathcal{D}}\mathbf{l}_j - {}^{\mathcal{D}}\mathbf{p}_i\|_2}{2r_l}) \cdot 255 \rfloor, & \|{}^{\mathcal{D}}\mathbf{l}_j\|_2 < \|{}^{\mathcal{D}}\mathbf{p}_i\|_2 \end{cases}, \tag{21}$$

is the normalized depth in grayscale. If multiple points are mapped to the same pixel, the corresponding grayscale value is averaged. $\mathbf{I}_r$ is also called the range image [18], [19] which is used as the input of the RSS-Net.

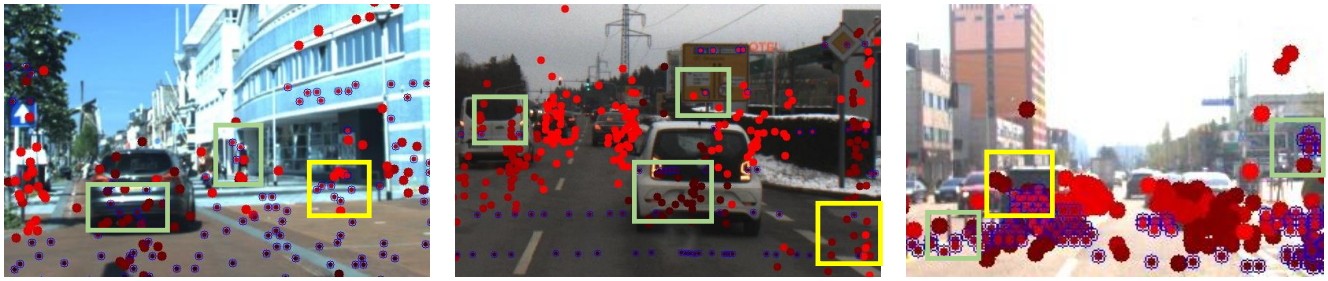
2) Network Architecture and Training Scheme: Fig. 6 illustrates the overall network structure of the RSS-Net. The image patches $\mathbf{I}^*$ are $2r_c \times 2r_c \times 3$ in input dimension. For each corresponding local lidar point cloud, it is a $w \times h \times 1$ range image $\mathbf{I}_r$. The positions of both are determined by the corresponding radar point ${}^{\mathcal{D}}\mathbf{p}_i = [x_i, y_i, z_i, v_i]^{\mathsf{T}}$. It is worth noting that the Doppler velocity of the target point is also used as input because moving targets also tend to trigger stronger radar signals due to its filtering algorithm.

Due to the limited input size for RSS prediction, we design a lightweight neural network instead of utilizing the ResNet backbone. As shown in Fig. 6, image patches and local lidar point clouds are processed separately by two convolutional networks whose outputs are the same in dimension. The outputs are then concatenated to form a 32-channel feature map. It is further processed by a convolutional network to reduce the channel number to 1, so it is flattened and input to a FC layer. The output tensor contains both appearance and geometry information. It is concatenated with the feature of the embedded target point, and the concatenated tensor is finally processed by a multilayer perceptron (MLP) to generate the estimated RSS $\hat{a}_{\text{RSS}}$.

In the training stage, we employ signals from the real radar datagram that include ground truth signal positions, Doppler velocity, and RSS value a_{RSS}. Similar to (9), the following loss function is the normalized INS difference and used to measure the performance of the network and to optimize parameters:

$$L_a = \frac{1}{m} \sum_{i=1}^{m} \left(\frac{a_{\text{RSS},i} - \hat{a}_{\text{RSS},i}}{a_{\max} - a_{\min}} \right)^2, \tag{22}$$

where m is the total number of signals, and $a_{\max}$, $a_{\min}$ are maximum and minimum RSS values, respectively.



(a) ZF FRGen21, VoD (b) Astyx 6455, Astyx (c) Oculii Eagle, MSC

Fig. 7: Mmwave radar (big dots) and lidar (small dots with blue outer ring) point cloud projections in representative scenes on three datasets. The color of dots illustrates depth of each point, gradually changing from black (0 m) to red (50 m). Green and yellow bounding boxes represent good and bad depth consistency for mmwave radar and lidar, respectively. Calibration errors, synchronization errors, and differences in sensor properties cause inconsistencies.

V. EXPERIMENTS

We have implemented the proposed Dis-Net and RSS-Net using PyTorch. The training and evaluation procedures are conducted with an NVIDIA RTX 3090 graphics card. Adam optimizer [20] is used for both networks, and the learning rate is 1×10^{-4} for both networks. For RSS-Net, the shape of lidar range image is set to 128×32, which is determined by the ratio of horizontal and vertical angular resolutions of lidar.

A. Dataset Illustration

Experiments have been conducted on three open datasets with three different radar models: 1) View-of-Delft (VoD) dataset [4], 2) Astyx 2019 dataset [3], and 3) MSC-Rad4R dataset [6]. The first two datasets provide rich annotations for object detection and tracking, and the third provides pose information for Simultaneous Localization and Mapping (SLAM). The detailed information is shown in Tab. I. VoD dataset contains 21 sequences, 13 of which were collected from residential areas, 7 from urban streets, and one mixed type. Astyx dataset includes only one sequence recorded on open roads. As for MSC-Rad4R (MSC) dataset, sequence URBAN_D0 is utilized because it provides the most accurate calibration.

The representative scenes and properties of the equipped radars/lidars are as shown in Fig. 7. The radar signals in VoD dataset are less noisy, but the signal distribution is not concentrated on the target object. The radar signals in the Astyx dataset are more concentrated on moving vehicles and metal products, but they are extremely noisy, and the lidar point cloud is sparse. The radar signals in the MSC dataset accurately appeared near objects such as vehicles and road pillars, but the lidar did not successfully return readings for all vehicles. Due to the fact that depth inconsistency increases as depth itself increases, we only generate radar datagram within 50 meters at the intersection of the sensors' FoV.

To separate training and testing datasets for Dis-Net, we randomly sample 30% frames from sequences of different terrains/scene types for testing and leave the rest for training. The random sampling process ensures the distribution of different terrains/scene types in the training set and test set remains the same. The training set and test set of RSS-Net are obtained by randomly sampling radar signals from each frame of Dis-Net's training set and testing set. The numbers of sampled signals from VoD, Astyx, and MSC datasets are 50, 100, and 100 per frame, respectively, covering signals reflected from varies of ranges, angles, and objects. The parameter settings for RSS-Net is $r_c = 50$ and $r_l = 1$, which are tuned from 9 pairs of parameters drawn from sets $r_c \in \{25, 50, 125\}$ and $r_l \in \{0.5, 1, 2\}$.

B. Ablation Study

We conduct an ablation study to evaluate the impact of various input modalities on the performance of both Dis-Net and RSS-Net.

Tab. II shows the results of the Dis-Net ablation study, showing that utilizing both image and ego velocity leads to better performances except on VoD dataset. It is worth noting that VoD dataset is collected by a radar with very sparse signals in low-speed scenes which means the Doppler velocity feature is not significant, thus the use of ego velocity does not obtain better results.

Tab. III shows the results of the ablation study for RSS-Net. It is clear that utilizing all three input modalities, i.e. camera (Cam), lidar (Li), and target radar point (RP), produce the best results, especially on datasets with cluttered and noisy radar signals. Moreover, for low-fidelity radar data, such as VoD and Astyx dataset, the target point and the local lidar point cloud are among the more important modalities in predicting RSS, which is not surprising, because radar signal attenuates over distance and the point cloud also carries shape information. 2) Model trained on MSC dataset is not sensitive to the input modalities, because it provides high fidelity radar signals with less noise, making the network converge significantly better than others. There is a big difference before and after adding RP. This is due to the low resolution of the images, and the sensitivity difference between lidar and radar (as shown in Fig. 1 (d)).

Manuscript 2763 submitted to 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 1, 2025.

TABLE I: Dataset information. #Lines refer to the number of lidar scan lines. Avg #Signal/F refers to Average number of signals per frame. #Train (F) refers to number of the training frames. δ_1 and δ_2 are the horizontal and vertical angular resolutions of radar, respectively. Note that pitch and yaw angular resolutions of radar δ_1 and δ_2 for Astyx are estimated from the data due to lack of documentation.

Dataset	Radar type	δ_1	δ_2	Avg #Signal/F	Min/Max #Signal/F	#Lines	Image Resolution	#Train (F)	#Test (F)
VoD	ZF FRGen21	1.5°	1.5°	278.26	49/661	64	1936×1216	6065	2617
Astyx	Astyx 6455	1.5°	1.5°	969.08	183/3629	16	2048×618	382	164
MSC	Oculii Eagle	1.0°	1.0°	1639.43	978/2462	128	720×540	744	320

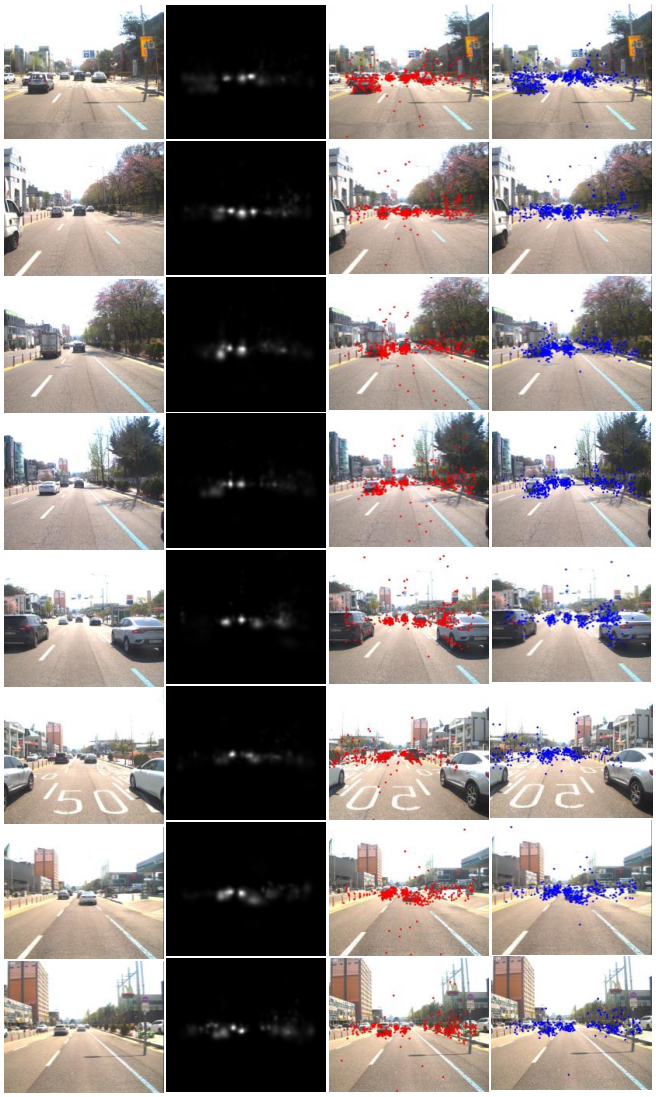


Fig. 8: Performance visualization of Dis-Net. The four pictures in each row are raw images, predicted radar point probability distributions, and real (red) and synthesized (blue) radar point cloud projections. It is worth noting that closeup vehicles in the raw images may not register on radar and lidar modalities due to perspective difference.

C. Dis-Net Result Visualization

The result of KL-divergence loss L_{KL} and number loss L_n in Tab. II actually show that the predicted signal distribution and number are consistent with the ground truth. To better demonstrate this conclusion, visualization is performed on the MSC dataset. Fig. 8 shows typical results of the simulated radar datagram when compared to its ground-truth counterpart. Only 20% of points within FoV are shown in the third and fourth columns to avoid cluttering. All sets of images show that the synthesized radar signal distribution is highly consistent with that of the real radar. More visualization results are shown in the attached video.

TABLE II: Ablation study of Dis-Net. Bold fonts indicate the best results.

		VoD		Astyx		MSC	
Image	Velo	L_{KL}	L_n	L_{KL}	L_n	L_{KL}	L_n
✓	×	**0.1538**	0.0266	0.2359	0.4745	0.0539	0.3561
✓	✓	0.1564	0.0266	**0.2271**	**0.4499**	**0.0524**	**0.3545**

TABLE III: Ablation study of RSS-Net. Bold fonts indicate the best results for each dataset.

			VoD	Astyx	MSC
Cam	Li	RP	L_a	L_a	L_a
✓	×	×	8.03×10^{-3}	1.53×10^{-2}	1.50×10^{-2}
×	✓	×	5.89×10^{-3}	1.51×10^{-2}	2.77×10^{-2}
×	×	✓	5.31×10^{-3}	1.53×10^{-2}	2.47×10^{-7}
×	✓	✓	4.14×10^{-3}	1.47×10^{-2}	7.27×10^{-8}
✓	×	✓	4.73×10^{-3}	1.51×10^{-2}	6.91×10^{-8}
✓	✓	✓	$\mathbf{3.95 \times 10^{-3}}$	$\mathbf{1.45 \times 10^{-2}}$	$\mathbf{5.16 \times 10^{-8}}$

D. Test Synthesized Radar Signals with Object Detection Task

Next we show the synthesized radar datagram can be used to train an existing radar-based network for object detection, i.e. distinguishing cyclists and cars on the street. We follow the setup of the VoD dataset [4] to implement training and testing process using PointPillars (**PP**) [22], which is available at OpenPCDet [23].

Tab. IV shows the results of **PP** training on only real data and on both real and synthesized data. **PP** is first trained and tested on subset S_1 and S_2 from real radar data. The result is compared with another training scheme, training on $S_1 \cup S_3$ and testing on S_2, where S_3 is from synthesized radar data. S_1 and S_2, with length of 3149 and 1350, are sampled from the training set of Dis-Net. S_3, with length of 723, comes from the inference results of Dis-Net on its testing set.

The results demonstrate that $\textbf{PP}_{S_1 \cup S_3}$ outperforms $\textbf{PP}_{S_1}$ significantly in cyclist detection, while maintaining a slightly better performance in detecting cars. This is due to the

Manuscript 2763 submitted to 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 1, 2025.

TABLE IV: Performance comparison of Models PP_{S_1} and $\text{PP}_{S_1 \cup S_3}$ (using x, y, z, v features) on Car (IoU=0.5) and Cyclist (IoU=0.25), following the same settings as [4]. Green up arrow ↑ represents improvement, red down arrow ↓ represents the opposite. The evaluation metrics are consistent with the KITTI open dataset [21].

Model	**Car (IoU=0.5)**				**Cyclist (IoU=0.25)**			
	bbox AP	**bev AP**	**3D AP**	**AOS AP**	**bbox AP**	**bev AP**	**3D AP**	**AOS AP**
$\textbf{PP}_{S_1}$	52.18	54.54	24.48	52.14	18.69	16.45	11.73	18.46
$\textbf{PP}_{S_1 \cup S_3}$	52.20 ↑	54.46 ↓	24.53 ↑	52.16 ↑	26.81 ↑	21.30 ↑	14.87 ↑	26.77 ↑
$\textbf{PP}_{S_1 \cup S_3}$(with noise)	53.23↑	54.21↓	24.74↑	53.17↑	19.34↑	16.58↑	7.73↓	19.18↑

depth inconsistency between synthesized/lidar data and real data has a greater impact on detecting objects with larger bounding boxes. In addition, to further simulate the noise caused by scattering or multipath effects, 5% of uniform noise is added to the synthesized data. Results show that training on dataset augmented by synthesized data improves for most metrics.

VI. CONCLUSIONS

To assist development for radar-based vehicle navigation, we reported a new algorithm that can simulate automotive radar signals using an image, a lidar point cloud, and ego-velocity. The method does not require a precise 3D scene model or material type information for radiation pattern simulation. It directly predicted radar signal distribution and number using Dis-Net, a neural network built on a ResNet-18 backbone. We also designed a multimodal RSS-Net to predict RSS, a signal strength measure. We have implemented and tested the proposed method with datasets from 3 different commercial radars. Our experimental results have shown that the design was successful and the simulated radar signals have shown very high fidelity when comparing to the ground truth.

In the future, we will further improve the network design to better utilize the multimodal information to improve fidelity and test the development in Sim2Real settings.

REFERENCES

[1] K. Harlow, H. Jang, T. D. Barfoot, A. Kim, and C. Heckman, "A new wave in robotics: Survey on recent mmwave radar applications in robotics," *IEEE Transactions on Robotics*, 2024.

[2] L. Fan, J. Wang, Y. Chang, Y. Li, Y. Wang, and D. Cao, "4d mmwave radar for autonomous driving perception: a comprehensive survey," *IEEE Transactions on Intelligent Vehicles*, 2024.

[3] M. Meyer and G. Kuschk, "Automotive radar dataset for deep learning based 3d object detection," in *2019 16th european radar conference (EuRAD)*. IEEE, 2019, pp. 129–132.

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

[5] L. Zheng, Z. Ma, X. Zhu, B. Tan, S. Li, K. Long, W. Sun, S. Chen, L. Zhang, M. Wan *et al.*, "Tj4dradset: A 4d radar dataset for autonomous driving," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2022, pp. 493–498.

[6] M. Choi, S. Yang, S. Han, Y. Lee, M. Lee, K. H. Choi, and K.-S. Kim, "Msc-rad4r: Ros-based automotive dataset with 4d radar," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7194–7201, 2023.

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

[8] C. Schöffmann, B. Ubezio, C. Böhm, S. Mühlbacher-Karrer, and H. Zangl, "Virtual radar: Real-time millimeter-wave radar sensor simulation for perception-driven robotics," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4704–4711, 2021.

[9] X. Cheng, Z. Huang, L. Bai, H. Zhang, M. Sun, B. Liu, S. Li, J. Zhang, and M. Lee, "M 3 sc: A generic dataset for mixed multi-modal (mmm) sensing and communication integration," *China Communications*, vol. 20, no. 11, pp. 13–29, 2023.

[10] M. Ouza, M. Ulrich, and B. Yang, "A simple radar simulation tool for 3d objects based on blender," in *2017 18th International Radar Symposium (IRS)*. IEEE, 2017, pp. 1–10.

[11] X. Chen and X. Zhang, "Rf genesis: Zero-shot generalization of mmwave sensing through simulation-based data synthesis and generative diffusion models," in *Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems*, 2023, pp. 28–42.

[12] K. Deng, D. Zhao, Q. Han, Z. Zhang, S. Wang, A. Zhou, and H. Ma, "Midas: Generating mmwave radar data from videos for training pervasive and privacy-preserving human sensing tasks," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 7, no. 1, pp. 1–26, 2023.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[14] M. Song, J. Lim, and D.-J. Shin, "The velocity and range detection using the 2d-fft scheme for automotive radars," in *2014 4th IEEE International Conference on Network Infrastructure and Digital Content*. IEEE, 2014, pp. 507–510.

[15] P. Sturm, "Pinhole camera model," in *Computer Vision: A Reference Guide*. Springer, 2021, pp. 983–986.

[16] W. P.-B. Rendering, "Physically-based rendering," *Procedia IUTAM*, vol. 13, no. 127-137, p. 3, 2015.

[17] A. Kingery and D. Song, "Improving ego-velocity estimation of low-cost doppler radars for vehicles," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9445–9452, 2022.

[18] S. Stiene, K. Lingemann, A. Nuchter, and J. Hertzberg, "Contour-based object detection in range images," in *Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06)*. IEEE, 2006, pp. 168–175.

[19] B. Steder, G. Grisetti, M. Van Loock, and W. Burgard, "Robust on-line model-based object detection from range images," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 4739–4744.

[20] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[21] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The international journal of robotics research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[22] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 12 697–12 705.

[23] O. D. Team, "Openpcdet: An open-source toolbox for 3d object detection from point clouds," https://github.com/open-mmlab/OpenPCDet, 2020.