Radar-guided Monocular Depth Estimation and Point Cloud Fusion for 3D Road User Detection

MSc Thesis

S. Baratam

TU Delft
Delft University of Technology

Radar-guided Monocular Depth Estimation and Point Cloud Fusion for 3D Road User Detection

MSc Thesis

by

S. Baratam

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on June 28, 2022.

Student number:	5248566	
Project duration:	September 9, 2021 – June 28, 2022	
Thesis committee:	Prof. dr. D. M. Gavrila,	TU Delft, supervisor
	Ir. A. Palffy,	TU Delft, daily supervisor
	Dr. J.F.P. Kooij,	TU Delft,
	Dr. F. Fioranelli,	TU Delft

This thesis is confidential and cannot be made public until June 28, 2022.

TU Delft

Acknowledgement

I would like to first convey my gratitude to Prof. Dariu Gavrila for supervising my thesis research. His guidance and feedback always pushed me conduct a high quality research with a pragmatic outlook. My daily supervisor, Ir. Andras Palffy, has always been there with continuous feedback, support and priceless discussions. Together, they filled my journey in the second year of masters program with learning, value and fun.

My parents Prasad Rao Baratam and Surya Kumari Baratam, along with my brothers Manikanta Bhagya Raj and Krishna Teja have extended their best support at every moment of my life including these two years. It is their love and support that gives me the strength to achieve my dreams. I also thank my friends Anish and Iva for making my life in Delft cheerful.

S. Baratam
Delft, June 2022

Radar-guided Monocular Depth Estimation and Point Cloud Fusion for 3D Road User Detection

Srimannarayana Baratam[1]

***Abstract*—Multi-class road user detection using the next-generation, 3+1D (range, azimuth, elevation, and Doppler) radars has been shown feasible, thanks to the increased density of their point clouds and the inclusion of elevation information. However, object detection networks using LiDAR (64-layer) point clouds still dominate the performance metrics. In this work, we explore the potential of fusing a 3+1D radar point cloud and a monocular image to further close this performance gap in 3D object detection. We propose a generic and modular fusion architecture to extract both spatial and semantic cues from a RGB image to complement the radar point cloud. In a two stage approach, we first generate a 3D point cloud representation of the input monocular image appended with semantic information through our proposed RAID (RAdar guided Instance-aware Depth) network, which takes monocular depth map and panoptic masks predicted from any pre-trained state-of-the-art networks, and a radar depth map as input. We then append the resulting point cloud to the 3+1D radar point cloud in a straightforward fusion scheme and train a point cloud based object detection network.**

Results on the View-of-Delft dataset [1] show that our fusion approach significantly outperforms multiple state of the art radar-camera fusion methods (proposed fusion vs. best baseline: 53.6 mAP vs. 50.8 mAP), and yields comparable performance to a network trained on LiDAR input when evaluated in the safety critical driving corridor (80.5 mAP vs. 81.6 mAP).

I. INTRODUCTION

Detection of the most common road users - pedestrians, cyclists and cars - is critical for autonomous driving. LiDAR has dominated camera and radar sensors for 3D object detection in research to date, but its commercial adoption is limited from the perspective of price and integration within stock vehicles.

The performance gap of radar or camera based networks compared to those using LiDAR input is mostly due to insufficient information: while monocular camera images are rich in semantics and texture, they lack depth information [2], [3]. On the other hand, point clouds from conventional 2+1D (range, azimuth, 1D refers to Doppler) radars provide excellent range measurements but are often claimed too sparse for object classification [4], [5] and precise 3D bounding box regression [1].

The complementary properties of camera and radar sensors are often recognized in the literature [6], [7], [8]. Although it would be a highly cost effective [9] sensor setup, the fusion of radar and camera sensors for the task of 3D multi-class object detection still falls behind in performance when compared to networks relying on high-end LiDARs [10]. This is partly due to the aforementioned information gap and partly because the lack of elevation information in radar point cloud further acts as a bottleneck in learning the association between radar points and camera pixels [7], [11].

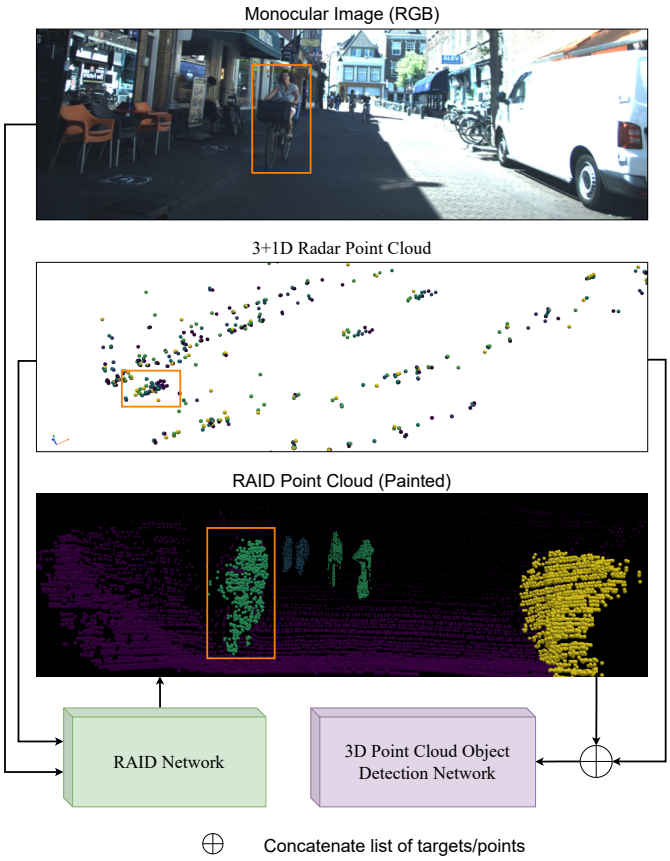


Fig. 1: High-level overview of our proposed two-stage fusion approach for 3+1D radar point clouds and monocular images. We first generate a point cloud representation of the image using radar depth cues via our novel "RAID" network in Stage-1. In the next stage, we append it to the list of radar points, and pass the resulting fused point cloud as input to a 3D point cloud object detection network. An instance of a cyclist is highlighted with orange boxes both in the inputs (camera image, radar point cloud) and in the output (painted point cloud) of RAID. In the RAID point cloud image, cyclist, car and pedestrian are painted in green, yellow and blue here for the purpose of illustration.

Next generation, 3+1D automotive radars may help to overcome these limitations. Unlike 2+1D radars, they have three spatial dimensions: range, azimuth, elevation along with Doppler as a fourth dimension, and also tend to provide a

1) Intelligent Vehicles Group, TU Delft, The Netherlands

denser point cloud. With the additional elevation information and increased density, these radars may be better suited for fusion with monocular camera images for multi-class 3D object detection.

Recently, [1] introduced a multi-modal dataset called the "View-of-Delft" (VoD) featuring a 3+1D radar, a 64-layer LiDAR and a high definition stereo-camera. Prior to this work, "Astyx" [12] was the only dataset ($\sim$ 550 frames) using a 3+1D radar sensor. Limited literature on fusion between 3+1D radar point cloud and camera can thus be attributed also to the lack of datasets featuring such radar point clouds.

Research on fusion of 3D point clouds and camera images has been largely focused around LiDAR-camera fusion, where the camera image is either used to extract semantic information [13], [14] and/or proposals [15], while relying heavily on LiDAR for 3D geometric/spatial cues. However, this approach cannot be directly extended to 3+1D automotive radars because the density of radar point clouds is lower than LiDAR point clouds. Also radar does not capture sufficient object delineation even after accumulating multiple scans. Availability of the radar points on static instances is further limited compared to dynamic instances since they cannot be resolved along the Doppler dimension.

Recent works [16], [17] on LiDAR-camera fusion argue that detection performance is degraded for distant objects with few LiDAR points even though these instances are clearly seen and detected in the camera image (e.g., via semantic masks). Thus, they propose to lift image pixels into 3D space to extract spatial cues along with semantic cues, even though their downstream detection pipelines are different. We recognize the similarity of this problem statement to our task and propose to move in a similar direction i.e., to fuse monocular image with the radar point cloud by re-projecting the pixels into 3D space using depth.

We therefore address the task of road user detection using a 3+1D radar and a monocular camera in a two-stage approach. First, in a novel, modular and task-aware method called "RAID" (**RA**dar guided **I**nstance-aware **D**epth), the representation of monocular images is changed into painted point clouds using radar depth cues. In this stage, we exploit state-of-the-art monocular depth estimation and panoptic segmentation networks pre-trained on existing large uni-modal datasets in a modular way to predict dense depth required to re-project pixels into 3D space. In the next stage, the obtained painted point cloud is concatenated with radar point cloud for the downstream multi-class 3D object detection task. More specifically, the fused point cloud is passed as input to a state-of-the-art object detector (PointPillars [18]) for 3D bounding box inference. In contrast to the "end-to-end" object detection network paradigm [19] (i.e., training a single deep neural network from scratch with point clouds and/or images as input and detections as output), where large, fully annotated multi-sensor datasets are needed, our proposed method exploits both uni-modal research and datasets to alleviate this concern.

II. RELATED WORK

A. Monocular camera

There are two main paradigms in the literature on 3D object detection with a monocular camera - one approach is to first solve for dense (i.e. for all pixels in the image) depth [20], [21], [22] either by using off-the-shelf monocular depth estimation networks like [23], [24], [25] or by proposing a new approach to deal with depth [20], [26], [27]. Alternatively, researchers also use geometric priors as in [28], [29] to rather resolve depth at the object instance level. In [2], [3], the authors demonstrate that object depth estimation is the limiting factor for monocular 3D object detection performance; they replace predictions with ground truth values for depth, orientation, offset, etc and observe that an accurate depth can bring significant performance improvement. Monocular camera only 3D object detection methods are still inferior in benchmarking performance compared to LiDAR based networks.

Estimation of depth per-pixel is often formulated as a regression problem in continuous space [30], [31], [32], [33]. In more recent approaches however, researchers choose to formulate the prediction of depth map as a classification or ordinal regression problem [24], [34], [35] instead of a classical regression problem. In other words, depth is discretized into bins (i.e., set of intervals along the depth dimension) rather than treating it a continuous variable. [34] argues that ground truth depth value cannot be exactly regressed with image input and a bin/range based prediction relaxes this constraint. [24] further argues that regression methods are slow in convergence and incur high computational cost.

Researchers also used additional instance information to improve image reconstruction loss [36], or to train separate branches to estimate the depth of foreground and background pixels separately [26] in the literature. Alternatively, some methods [37] [38], [39], [40] also jointly perform monocular depth estimation and semantic segmentation to achieve state-of-the-art results in the respective tasks. Nevertheless, none of the found methods used instance segmentation masks as input to the depth prediction network during inference.

B. Radar-camera fusion

While fusion of radar and mono-camera has been performed for 3D perception tasks such as 3D object detection [6] and depth estimation [11], [41], [42], most radar-camera fusion research so far involved a 2+1D radar that lacks elevation information and provides a sparse radar point cloud output. Therefore, radar-camera fusion networks addressing object localization [43], [44] and detection tasks [7], [45], [46] are mostly limited to 2D space - either in bird's-eye-view or image plane. To improve the density of radar signal when projected to image plane, researchers even extrapolated radar targets into vertical pillars [7] or circles [11]. While these extrapolation methods are non-complex in implementation, they are prone to propagating noise due to strong underlying assumptions, e.g, if a radar point corresponds to reflection on a pedestrian in the foreground, extrapolating it in a vertical pillar would propagate this depth cue to a background object like a pole or a wall. While works like [11] propose a more advanced way

to tackle the lack of elevation information, it still remains a bottleneck in effectively propagating depth.

Unlike depth completion using LiDAR and monocular camera where sufficient depth cues are provided from LiDAR alone to predict dense depth [47], the task is non-trivial to perform in the context of radar point cloud due to sparsity and noise. In radar-camera based depth estimation research, pre-processing of the radar point cloud is usually aimed at three challenges: handling radar points occluded by foreground objects, removing noisy/outlier points, and finally, improving point cloud density. For example, [11] first learns an association between radar points and camera pixels using LiDAR depth as supervision signal which implicitly filters the occluded radar points. On the other hand, [41], [42] use an iterative depth prediction approach. First, a "raw" dense depth is estimated to filter the radar point cloud which is then used to further "refine" the depth prediction in the subsequent stage. However, there is no discussion on "trails" that occur in radar point cloud accumulated over multiple scans for dynamic objects (refer to Appendix section A). Further, these works use 2+1D radar point cloud and also lack research on the feasibility of the predicted depth maps for downstream perception tasks like 3D object detection.

Next generation automotive radars provide elevation information and denser point clouds. [48] is the only 3+1D radar-camera fusion attempt in literature so far where the authors directly adopt the AVOD network from LiDAR-camera fusion domain for 3D object detection. Also, the authors experimented on a single class (car) detection alone and did not use Doppler information which can help improve the object detection performance as shown in [1].

C. LiDAR-camera fusion

In the context of fusion between LiDAR and camera sensors for 3D object detection, multiple paradigms have emerged producing state-of-the-art results in 3D object detection.

Researchers, in early attempts, gave a "regular/grid-like" representation to LiDAR point cloud by projecting it directly onto the horizontal/ground plane to generate "bird's-eye-view (BEV) image" [49], [50] and/or image plane to generate a "sparse depth image/perspective-view image" [51], [52] of the environment. This allowed them to apply 2D image processing backbones to extract features from LiDAR point cloud along with deep features from the input RGB image using another image backbone network in a parallel stream. In [51], [52], proposals are generated from BEV stream to perform RoI (Region of Interest) pooling of the generated features for downstream fusion. [49] follows an iterative approach where the network uses an anchor grid to crop and fuse these multi-view features to generate proposals. These proposals are again used to aggregate features for final prediction of objects. Alternatively, [50] uses "continuous fusion layers" to pass on deep features from camera stream to LiDAR stream in a uni-directional way. With the exception of [52] where dense depth completion is also performed to learn pixel-wise feature fusion between the two streams, these view-based feature aggregation methods generally extract semantic features and shape cues from input image, while relying on LiDAR for 3D information.

In an intuitive approach, some methods use 2D detections inferred from the monocular image to "extract" the corresponding LiDAR point subset by either projecting the point cloud onto the image plane [53] or extend the 2D bounding box into the 3D space forming a frustum [15], [54], [55]. In a sequential approach, the extracted point cloud is then used to regress a 3D bounding box. If an object is not detected in 2D image domain, no query is passed to the downstream point cloud network to detect an object in 3D. Further, this approach assumes availability of 3D targets/points within the extended frustum.

A more recent paradigm in LiDAR-camera fusion is to "paint" the LiDAR points with image based semantic features/masks. The association between 2D semantic features/masks and 3D LiDAR point cloud is usually done by projecting the points onto the image plane. [13], [14], [56] use an off-the-shelf image segmentation network to generate semantic masks and append the class information to the corresponding LiDAR points along the feature dimension. Alternatively, [57], [58] argue that painting the LiDAR points with semantic features is better than just appending class scores, i.e., instead of adding a class encoding to the point, they append a vector of features. Methods following this painting approach currently lead the NuScenes [10] 3D object detection benchmark. However, this approach assumes availability of sufficient points to "pick" the semantic cues from the image.

Lifting image pixels (i.e., re-projecting) into 3D space as seen in [16], [17], [59] is often done when the LiDAR points are sparse either at a scene level [59] or at instance level [16], [17] irrespective of the downstream perception task. In [16], [17], authors also append semantic information to the re-projected pixels, thereby propagating both spatial and semantic cues from the camera image to 3D space. Among all the LiDAR-camera fusion paradigms, object detection with highly sparse point cloud and monocular image is addressed only in this approach.

Except for the networks that lift monocular images into 3D space i.e., re-project pixels into 3D, all other approaches typically exploit only semantic features from the image, while relying significantly on LiDAR for geometric/spatial cues. However, in the quest to fuse 3+1D Radar with camera image, this may not be effective as a 3+1D radar point cloud is significantly sparser, and unlike LiDAR, the radar targets are not uniformly spread throughout the image plane when projected. Furthermore, for stationary instances of classes like standing/sitting pedestrians, radar points may not be always available.

Our main contributions are as follows:

1) We introduce a novel, modular, and multi-level fusion architecture for 3+1D radar and monocular camera-based 3D object detection. It creates a point-cloud representation of the camera image via radar-guided depth estimation to address a downstream object detection task for the first time in research.
2) To demonstrate the capability of our proposed fusion approach, we perform extensive experiments on the View-of-Delft dataset. Results show that our method outperforms the state-of-the-art approach for radar-camera

fusion and yield comparable performance to detections from state-of-the-art LiDAR and LiDAR-camera fusion networks in the driving corridor (Our Method vs. LiDAR vs. LiDAR-camera fusion: 80.5 mAP vs. 81.6 mAP vs. 85.3 mAP).

III. METHODOLOGY

A. Overview

Synchronized, calibrated, and annotated datasets featuring specific sensor combinations of interest are highly limited in dataset size and label types (bounding boxes, dense depth, semantic masks, etc.) compared to uni-modal datasets. End-to-end network paradigms for sensor fusion often suffer from this bottleneck. We argue that a modular approach can help alleviate this problem and therefore we present a two-stage solution to fuse a radar point cloud and a monocular image for 3D object detection as illustrated in Figure 2. In Stage-1 (subsection III-B), we generate a sparse depth map from a 3+1D radar point cloud, a monocular depth map, and panoptic masks from a single-frame RGB image. We propose a cross-sensor supervised depth refinement network to take these inputs and predict a metric depth map , then lift the image pixels into a semantic painted point cloud. In the subsequent Stage-2 (subsection III-C), this point cloud is fused again with the radar point cloud in a simple yet effective approach. Any state-of-the-art point cloud-based 3D object detection network can be trained on the fused point cloud, supervised by annotated labels.

B. Stage-1: RAID

Ground-truth signal: A high-end LiDAR sensor is capable of returning highly accurate 3D points representing the surroundings with a density sufficient enough to act as a supervising signal for dense depth estimation. In a "cross-modal" supervision scheme, we use the 3D LiDAR points projected onto the image plane as ground truth for our network. This, therefore, avoids the need for manual annotation since it is both expensive and laborious. However, due to the commonly observed setup of LiDAR atop the roof of an ego-vehicle, "conflict of depth" arises within instance masks from the camera perspective as shown in Appendix Figure 9. It results in a noisy supervision signal if not addressed. Hence, we apply two filtering techniques to mitigate this problem so that LiDAR points only visible from the camera perspective provide ground truth signals. First, a 2D kernel filter looks for depth conflicts in z-order by sliding through the depth map generated from LiDAR and ensures that only fore-most points are retained. Subsequently, for points falling on each instance mask (of interested class), DBSCAN filtering is implemented in 3D space to remove the background points. Specifically, background points that are visible to the LiDAR but not to the camera, as the object occludes them, are removed. See Appendix section B for further details.

Input radar point cloud pre-processing: A sparse image with depth, compensated Doppler and RCS values as channels is generated by projecting radar point cloud accumulated over five scans after ego-motion compensation onto the image plane. However, due to phenomenon like object trails in accumulation of scans and multi-path propagation, radar point clouds cannot be projected onto the image plane directly for providing depth cues. In order to ensure that only reliable radar depth cues are passed as an input signal and also improve the density of radar point cloud, we propose a sequential combination of pre-processing steps for the accumulated point cloud. Specifically, "target propagation" to deal with the object "trails", "spatial-temporal filtering" to remove noisy targets in a voting based method, and finally "sensor aware re-sampling" and "vertical sampling" to improve the density of radar point cloud.

Detailed explanation of these steps are provided in Appendix section A. Since the radar point cloud is accumulated over multiple scans, conflict of depth arises here too as each scan presents a view from different spatial locations (assuming that the ego-vehicle is moving in between the scans). Hence, kernel and DBSCAN filters are also applied to the radar sparse depth map.

Input image pre-processing: Unlike depth completion using LiDAR point clouds, where depth cues are more uniformly available, radar point clouds are comparatively sparse without any consistent pattern when projected onto the image plane. Further, most depth estimation and/or completion approaches treat foreground and background pixels with equal importance without any specific attention to the downstream task. For road user detection, however, pixels corresponding to the foreground instances, especially of classes of interest such as cars, pedestrians and cyclists, are of significance. To address these problems, we choose to exploit state-of-the-art monocular depth estimation [60] and panoptic segmentation networks [61] pre-trained on the KITTI-Depth [62] and Cityscapes [63] datasets, respectively. We would like to emphasize that the choice of networks here is not constrained with our approach, and other leading network(s) may also be used. First, to extract depth cues from image, we perform monocular depth estimation by using [60] to obtain a dense depth map. Using the pre-processed radar point cloud, the monocular depth map is then scaled to metric space through median scaling as done in [64]. In parallel, we generate semantic and instance masks with [61] to bring attention to foreground pixels in two ways - as input channels to the depth refinement network and in the loss function (see Stage-1 in Figure 2).

To summarize, the input to the depth refinement network consists of a monocular depth map (one channel - depth), panoptic masks in image representation (two channels - semantic and panoptic masks) and a radar depth map (three channels - depth, compensated Doppler and RCS) concatenated along the channel dimension to make it a six channel input.

Depth refinement network: The depth refinement network comprises of a lightweight 2D image processing semantic backbone [65] to learn deep features suitable for the per-pixel depth inference task, and an ordinal regression layer [24] to consume these feature maps and transform the depth regression at each pixel into a classification problem. We also discretize the depth space into K number of ordinal bins following the SID (Spacing-Increasing Discretization) scheme

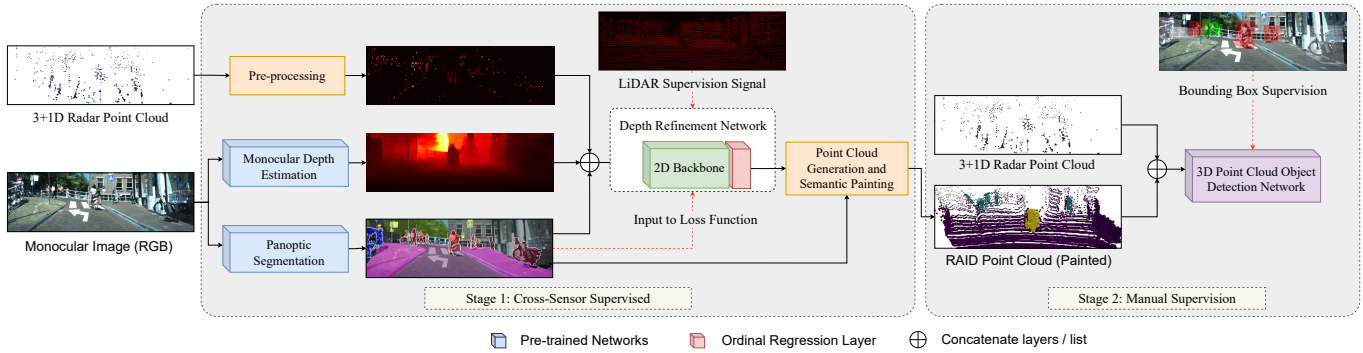


Fig. 2: Our proposed fusion architecture is modular and comprises of two stages. In Stage-1, also called RAID, we transform the representation of an input RGB image into a semantic painted point cloud in a novel approach with the help of radar depth cues. In contrast to the existing radar-camera fusion methods, we first extract raw monocular depth map and panoptic masks from the camera image using pre-trained networks, and concatenate them with a sparse radar depth map generated through our novel 3+1D radar pre-processing pipeline. A depth refinement network comprising of a lightweight 2D semantic backbone and an ordinal regression layer takes this input and learns to predict a refined depth map, supervised by the LiDAR depth map. We propose a combination of scene level and instance level loss function to bring attention to the foreground objects. Using camera intrinsic parameters and this refined depth map, pixels are then lifted to 3D space and appended with semantic class as a point feature. In Stage-2, the generated point cloud is further concatenated with the 3+1D radar point cloud after bringing in dimensional consistency. The resulting fused point cloud can then be passed to any point cloud based 3D object detection network using bounding box annotations as supervisory signal.

in [24], for both loss calculation and inference. The output from ordinal regression layer is then used to decode the depth at each pixel.

To facilitate the supervision through a sparse ground truth signal (LiDAR) and also bring attention to the instances of target classes, we define a hybrid loss function L_{RAID} comprising of a scene level loss L_S and an instance level loss L_I. For ease of comprehension and continuity, we follow the same notations from [24]. Given a six channel multi-modal input image I of size $(W \times H \times 6)$, let $\chi = \varphi(I, \Phi)$ denote the output feature maps of size $(W \times H \times C)$ from the 2D semantic backbone, where Φ represents its learnable parameters. Then, $Y = \psi(\chi, \Theta)$ of dimension $(W \times H \times 2K)$ denotes the output of ordinal regression layer, where Θ is the set of layer weights to be learned. Let $\hat{d}_{(w,h)}$ be the metric depth decoded at pixel location (w, h) from the predicted discrete ordinal label $\hat{l}_{(w,h)}$, and $l_{(w,h)}$ be the ordinal supervision label encoded via SID from the corresponding ground truth depth $d_{(w,h)}$. Then we define the loss to be optimized for our RAID network as,

$$L_{RAID} = L_S + L_I \tag{1}$$

where L_S is the loss calculated over the whole image for every pixel where sparse LiDAR supervision is available, and L_I is the loss calculated only where panoptic segmentation network detected classes of interests along with a ground truth signal. Both losses are defined below.

$$L_S = -\frac{1}{N}\sum_{i=0}^{N-1} \Psi(w_i, h_i, \chi, \Theta), \tag{2}$$

N represents the number of pixels in the depth map where a groundtruth signal (i.e., LiDAR points projected onto the image plane) is available for depth supervision and $\Psi(w_i, h_i, \chi, \Theta)$ is the pixel level ordinal loss introduced in [24].

Our Instance Loss L_I aims to "direct" the network's attention towards our main interest: road users. It is first calculated for each instance as a combination of the pixel level ordinal loss above, and a 3D reconstruction loss (see below). Then, the total instance loss for the scene (or batch) is a weighted average of this loss for each occurring instance. We emphasize that the instances and semantic classes are purely obtained through inference by a pre-trained panoptic network.

$$\begin{aligned} L_I &= \frac{\sum_{q=0}^{Q-1} w\left(c_q\right) \cdot I_q}{\sum_{q=0}^{Q-1} w\left(c_q\right)}; \\ I_q &= -\frac{1}{M}\sum_{j=0}^{M-1} \left(\Psi(w_j, h_j, \chi, \Theta) + \Delta(w_j, h_j, d_j, \hat{d}_j)\right) \end{aligned} \tag{3}$$

where Q denotes the number of instances in the frame that belong to specific classes of interest, I_q represents the instance level loss for an instance $q \in \{q_0, q_1, ...q_{Q-1}\}$ and c_q is its semantic class. A class dependent weighted average of I_q over all instances is proposed to overcome skewed distribution of classes in the dataset if any. The class weights $w(c_q)$ are hyper-parameters that can be tuned based on the class statistics from training data. M represents the number of pixels within an instance q where the ground truth signal is available. I_q is then defined as the sum of mean of the two pixel level losses i.e., $\Psi(w_j, h_j, \chi, \Theta)$ and $\Delta(w_j, h_j, d_j, \hat{d}_j)$. Δ is the reconstruction loss at a pixel location (w, h), defined as the euclidean distance between two re-projections of the pixel to 3D metric space - one obtained through predicted depth $\hat{d}_j$ and the other via ground truth depth d_j:

$$\Delta(w_j, h_j, d_j, \hat{d}_j) = ||T(w_j, h_j, d_j) - T(w_j, h_j, \hat{d}_j)||_2. \tag{4}$$

T is the mapping of the pixel from image space to 3D metric space which is a straightforward task given the camera intrinsic parameters. To avoid redundancy, the expressions for Ψ, $\hat{l}$, $\hat{d}$ are not further elaborated here. For further details, we refer the reader to Appendix, section C.

Point cloud generation and semantic painting: To generate a point cloud from the refined depth map (i.e. the output of Stage-1), we sample the pixels in two steps. First, a mask mimicking a 64-layer LiDAR 's scanning pattern projected onto the image plane is used to sample pixels from the dense depth map. Then a random sampling scheme is used to select a class-specific, pre-defined number of pixels for each instance to boost their representation. To be specific, the latter strategy is aimed at overcoming the bottleneck of decreasing resolution with range in LiDAR-like scanning. Therefore, we retain as much information as possible for farther objects for downstream detection task. The selected pixels can then be lifted to 3D metric space in any desired frame of reference using camera intrinsic and extrinsic parameters, thus, creating a point cloud representation of the environment. We further choose to append the semantic class information in a one-hot encoding to the generated point cloud, i.e., point-painting, similar to [13]. At the end of Stage-1, we have a semantic painted point cloud generated using a refined depth output from our RAID network comprising of six features per point $[x, y, z, s_1, s_2, s_3]$: three spatial coordinates and one-hot encoding of the semantic class.

C. Stage-2: Point cloud fusion and 3D object detection

Usage of 3+1D radar point cloud in Stage-1 is limited to image plane and only aimed at propagating depth cues for the refinement of depth map. As experimentally demonstrated in [1], Doppler and RCS features of radar can contribute significantly to the object detection task. Hence, as these features were not exploited in Stage-1, we choose to fuse the full-feature 3+1D radar point cloud again with the output from Stage-1 in a simple yet effective approach to provide more discriminatory features. In a recent work, [19] concatenated lists of point clouds from LiDAR, 2+1D radar and a stereo camera based point cloud representation of image for object detection. We choose to move in a similar direction in our method, i.e., we concatenate the radar point cloud with the list of point cloud generated in Stage-1 after ensuring dimensional consistency. This makes the overall fusion architecture a multi-level sensor fusion paradigm. For this second fusion step, the accumulated radar sweeps over 5 frames after ego-motion compensation is used albeit without any pre-processing to retain as much raw sensor information as possible, unlike Stage-1 where only reliable and consistent depth cues are expected from radar. Each radar target contains seven features $[x, y, z, rcs, v_r, v_{rc}, t]$, where v_r and v_{rc} relative and absolute (i.e., ego-motion compensated) radial velocity. t indicates the scan from which the target originates from. Target from the current (most recent) scan has $t = 0$ while a point from the second most recent has $t = -1$. The two lists of point clouds are first transformed to the same frame of reference. Then dimensions of these point clouds are adjusted to bring in consistency and enable concatenation of the lists. Each point in the concatenated list contains 11 channels - $[x, y, z, rcs, v_r, v_{rc}, t, s_1, s_2, s_3, m]$ where m indicates the "source sensor" of the point with a binary flag, and any unavailable feature of the respective point clouds are assigned a value of zero. For example, a radar target will have $[x, y, z, rcs, v_r, v_{rc}, t, s_1 = 0, s_2 = 0, s_3 = 0, m = 0]$ as features while the RAID point cloud is expanded as $[x, y, z, rcs = 0, v_r = 0, v_{rc} = 0, t = 0, s_1, s_2, s_3, m = 1]$.

3D Object Detection: With a fused point cloud representation of 3+1D radar data and monocular image, it is possible to use any point cloud based 3D object detection network for the task. Similar to our previous work [1] we use PointPillars for generating 3D bounding boxes given the stacked point cloud described in the previous subsection as input. It is to be noted that the ground truth labels are only used at this stage for training the object detection network, and that any point cloud based network can be chosen here.

IV. EXPERIMENTS

A. Dataset and network models

Our recent work, the View-of-Delft (VoD) dataset [1], is the only public dataset of reasonable size to date (∼8700 annotated frames) that includes a 3+1D radar along with a high-end 64-layer LiDAR and stereo camera data. A well balanced distribution among specific classes of interest c_q i.e., cars (∼27k), pedestrians (∼26k) and cyclists (∼11k) even makes it ideal for the experimental setup of this research. To further demonstrate the capabilities and benefits of our modular approach, and to avoid over-fitting on the VoD dataset, we use another dataset of (∼29k) un-annotated frames recorded with the same sensor suite presented in [1] to train the RAID network in a fully cross-sensor supervision scheme. To generate monocular depth maps and panoptic masks, we use the official code release of HR-Depth [60] pre-trained on KITTI Depth Completion dataset [62] and Efficient-PS [61] pre-trained on Cityscapes dataset [63]. No fine tuning is performed for these networks on VoD dataset prior to our experiments.

For the 2D backbone of depth refinement network in Stage-1, we choose PyTorch implementation of MobileNet-V3 backbone with a Lite R-ASPP (reduced design of the Atrous Spatial Pyramid Pooling) segmentation head [65]. During training, we initialize this backbone with weights pre-trained on [66]. To ensure a fair comparison, we use the same 3D object detection network - PointPillars [18] in our experiments trained using various point clouds (from radar, LiDAR, image and sensor fusion). All networks are trained for 80 epochs, with a voxel grid resolution of $[0.16 \times 0.16 \times h_{span}]$ where h_{span} represents the vertical span of the point cloud which varies for point clouds containing radar targets due to presence of targets below ground. Further, for radar only point cloud, the maximum number of points per pillar is limited to 10 which is otherwise 32 for all other networks. Due to the presence of Doppler signal, augmentation methods for training with fused point clouds containing radar targets are limited to flipping the data along $'X'$ axis and random scaling between the range $[0.95, 1.05]$. On the other hand, for point clouds - either uni-modal or fused - that do not use radar points, we also perform

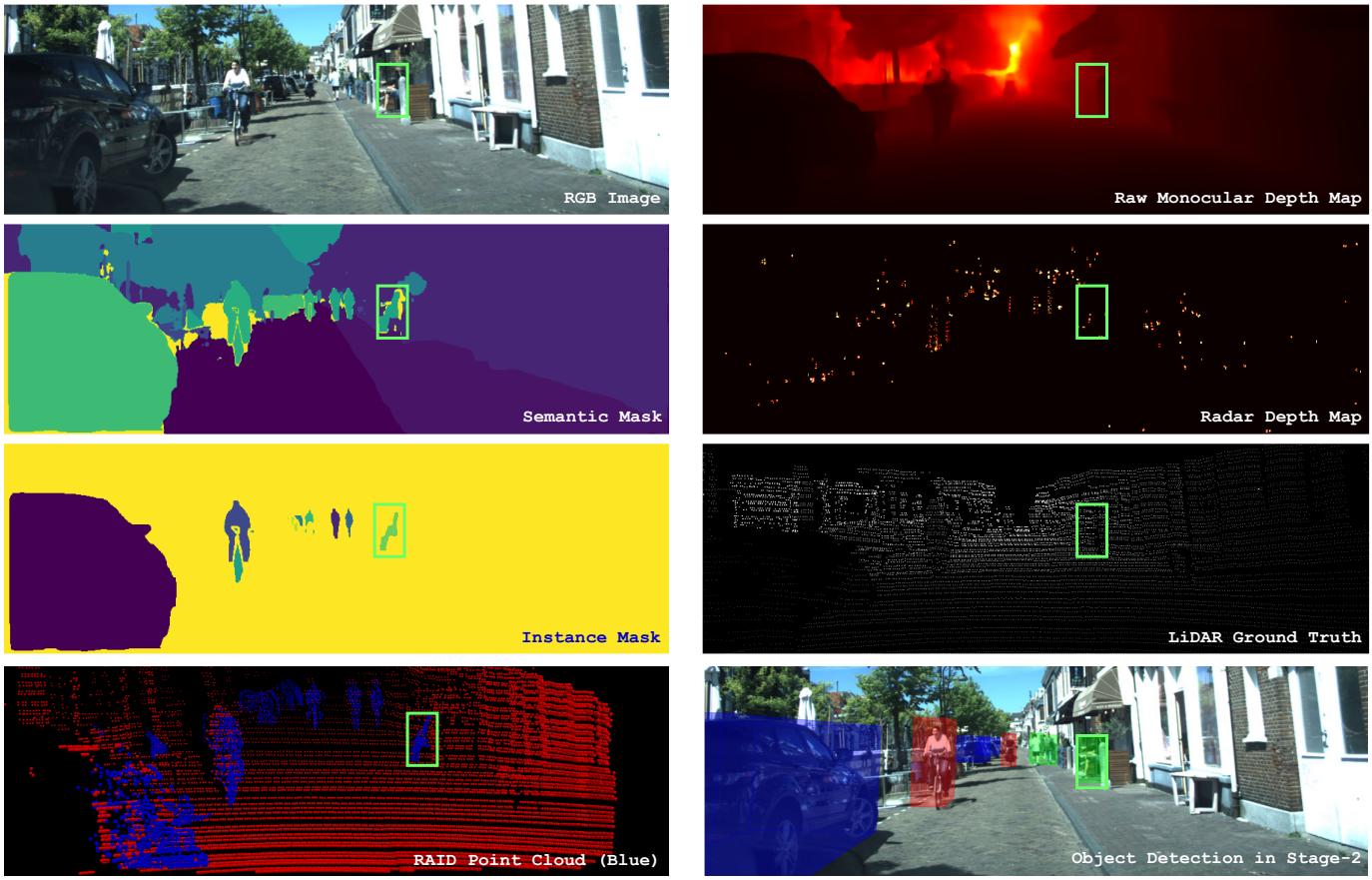


Fig. 3: Visualization of the inputs, supervision signal, and the output of Stage-1 (RAID), and the final 3D object detections of Stage-2 for a sample test frame. Top three rows: As indicated by the overlay text, RGB image input, its inferred monocular depth, semantic and instance masks from pre-trained networks, the radar depth map after pre-processing and the LiDAR depth map as ground truth signal for depth refinement. Bottom row, left: Point cloud generated from RAID, Stage-1 (blue), overlaid on LiDAR point cloud (red). For visualization purposes, we only show the re-projected pixels of the RAID point cloud corresponding to the road users. Green boxes highlight a challenging pedestrian instance in the scene - both occluded and static, basically invisible in the monocular depth map. Bottom right: PointPillars [18] using the fused point cloud in Stage-2 is able to successfully detect road users in the scene including the marked hard instance.

Fig. 4: Detection performance of our proposed method on two busy scenes in the test set. 3D Bounding boxes in red and green show successful detections of cyclists and pedestrians respectively. Yellow boxes indicate where instances are not detected (False Negative), while the cyan box on the right highlights a false detection (False Positive).

Method	Input to Detection Network	Entire annotated area					In Driving Corridor				
		Car	Pedestrian	Cyclist	mAP	mAOS	Car	Pedestrian	Cyclist	mAP	mAOS
AVOD [49]	Radar BEV-image; RGB-image	35.7	1.8	9.5	15.7	15.9	58.1	3.0	6.9	22.7	23.2
PP - RAID (painted)	Painted PCL from RAID	20.1	7.6	10.0	12.5	27.1	53.7	12.3	24.3	30.1	47.4
PP - mono fusion	Painted PCL from mono + radar targets	49.0	47.6	56.0	50.8	43.1	81.2	60.7	75.9	72.6	70.0
PP - radar (painted)	Painted radar targets	45.8	47.7	55.9	49.8	40.1	80.7	67.8	83.0	77.2	69.1
PP - RAID fusion	Painted PCL from RAID + radar targets	**51.8**	**50.4**	**56.9**	**53.0**	**48.6**	**89.6**	**68.6**	**83.3**	**80.5**	**77.7**
PP - mono (painted)	Painted PCL from mono	15.5	5.3	11.3	10.7	23.6	38.3	7.0	18.6	21.3	41.7
PP - radar (5 scans)	Radar targets	41.1	41.0	54.5	45.6	38.3	77.7	59.3	76.4	71.1	64.1
PP - RAID fusion	Painted PCL from RAID + radar targets	51.8	50.4	56.9	53.0	48.6	89.6	68.6	83.3	80.5	**77.7**
PP - LiDAR	LiDAR point cloud	75.6	55.1	55.4	62.1	49.4	90.8	71.4	82.5	81.6	70.3
PP - LiDAR (painted)	Painted LiDAR point cloud	**77.2**	**62.2**	**57.6**	**65.7**	**51.1**	**90.9**	**79.2**	**85.6**	**85.3**	70.6

TABLE I: Results for both 3D object detection experiments in the entire annotated area and within the "Driving Corridor" only. Top: Comparison of various approaches for fusion between radar point cloud and monocular camera. Bottom: Proposed fusion method compared with the performance of other sensors. All class-specific columns represent AP calculated with a 3D IoU of 0.5 for car and 0.25 for pedestrian/cyclist. Numbers highlighted in bold indicate the best performance in the section within the column. In bottom section, underlined numbers are within 5 mAP/mAOS points of the best performer (in boldface). "Painted" refers to appending semantic class to point features in one-hot-encoding and "PCL" is abbreviation for point cloud. The prefix "PP" indicates that the method is trained with PointPillars [18].

global "rotation" about the vertical axis. All networks are trained in a multi-class fashion for the detection task using OpenPCDet [67].

B. Evaluation metric

In view of the main goal for this research, i.e., road user detection in urban scenarios, we consider 3D object detection performance as the evaluation metric. To be specific, we use two performance measures - Average Precision (AP) and Average Orientation Similarity (AOS) following the KITTI benchmark [68]. AP is calculated based on the intersection over union (IoU) in 3D space between the predicted and ground truth bounding boxes. We consider a 50% overlap for car, and 25% overlap for pedestrian and cyclist classes as in [68]. However, we also present detection performance over multiple overlap thresholds for a more comprehensive evaluation in main comparisons. Mean AP (mAP) and mean AOS (mAOS) are calculated by averaging class-wise results. We also report results for two regions: the entire annotated region and 2) a safety-critical "Driving Corridor" following the approach in [1].

Evaluation of detection performance for our proposed method is carried out in two stages. We first look at how the proposed fusion of 3+1D radar and monocular camera compares to applicable state-of-the-art methods for fusing a point cloud and RGB image for detection task in our scenario. Then, we also compare how our proposed fusion approach compares to performance of networks that take uni-modal input from camera, radar or LiDAR, and also a state-of-the-art LiDAR-camera fusion method.

C. RAID fusion vs radar-camera fusion baselines

In this subsection, we compare our proposed two-stage RAID fusion described in section III with three radar-camera fusion baselines. First, we re-implement the approach of Astyx [48], the only research to our knowledge to fuse a 3+1D radar and mono-camera. It is a straightforward implementation of AVOD network [49] where LiDAR input is replaced with 3D point cloud from radar. While the authors of [48] only presented work on single-class (car) detection, we trained AVOD on all the three classes of interest since we have sufficient class representation.

Second, we implement semantic painting of radar targets $PP{-}radar(painted)$ similar to the "point painting" paradigm in LiDAR-camera fusion domain. Methods following this approach [13] to fuse LiDAR and camera consistently lead the 3D object detection benchmarks like KITTI [68] and NuScenes [10].

In addition to the full RAID fusion $PP-RAID\ fusion$, we also evaluate both stages of our proposal in a stand-alone approach. To be specific, first, we evaluate the detection performance on the point cloud output from RAID depth after semantic painting $PP-RAID\ (painted)$, i.e. Stage-1 alone. Second, to evaluate stage-2 individually, a point cloud generated from monocular depth is painted with semantic class and stacked with radar point cloud ($PP-mono\ fusion$) to validate our hypothesis that both spatial and semantic cues are crucial for radar-camera fusion in 3D object detection.

See Table I for overview and the performance of each of these methods for the entire annotated area and within the "Driving Corridor" region. Our proposed method clearly surpasses all the above mentioned radar-camera fusion baselines.

D. RAID fusion vs sensor baselines

To demonstrate the redundant capability of our proposed sensor fusion as a perception module, we compare its performance with that of point clouds generated from different sensors. More specifically, we trained and compared several PointPillar models for several sensor data combinations. $PP-radar\ (5\ scans)$ takes 5 scans of accumulated radar data. $PP-LiDAR$ is trained with LiDAR point cloud only, while $PP-Mono\,(painted)$ is trained with semantic painted point cloud generated from the raw monocular depth. Finally, we also train a network $PP-LiDAR\,(painted)$ where the LiDAR points features are appended with semantic classes similar to [13]. Table I gives an overview of these methods and shows that uni-modal performance of monocular camera and radar is lower than LiDAR: $PP-mono(painted)$ vs. $PP-radar(5scans)$ vs. $PP-LiDAR$ (10.7 vs. 45.6 vs.

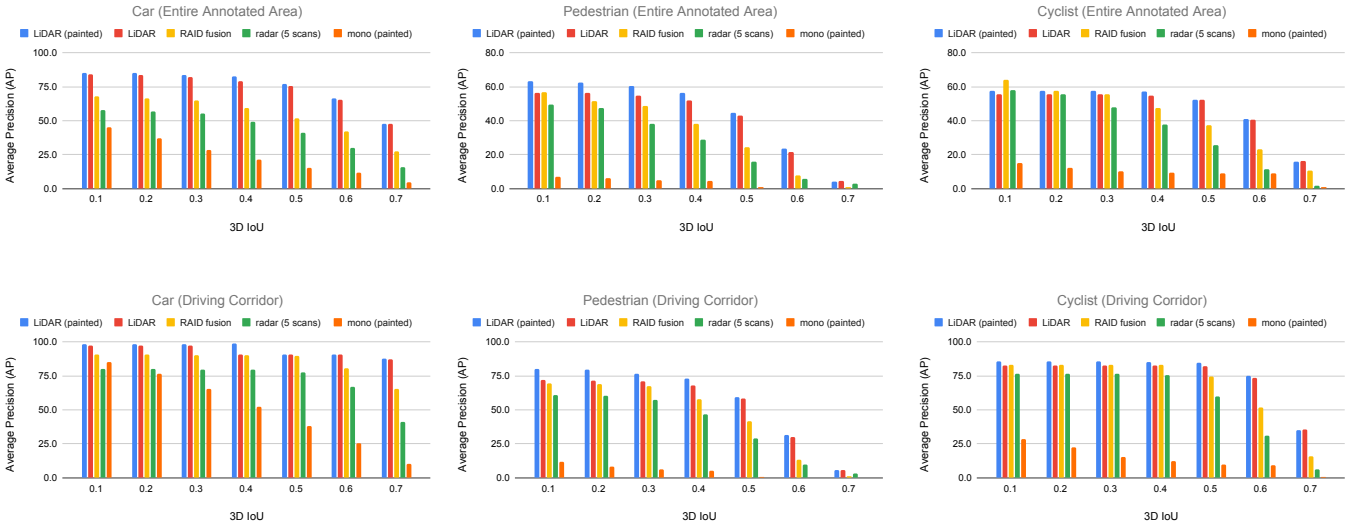


Fig. 5: Detection performance per class evaluated at multiple 3D IoU thresholds.

62.1) mAP in entire annotated area. However, our radar-camera fusion demonstrates comparable performance with the LiDAR baselines in the driving corridor in general ($PP-LiDAR(painted)$ vs. $PP-LiDAR$ vs. $PP-RAIDfusion$: 85.3 vs. 81.6 vs. 80.5), and for the cyclist class in entire annotated area (mAP: 57.6 vs. 55.4 vs. 56.9).

V. DISCUSSION

From the perspective of fusing the two modalities - radar and monocular camera, the first set of experiments provide a comparative evaluation for RAID fusion. From the results in Table I, our proposed 2-stage multi-level fusion approach outperforms all the baselines in the entire annotated area: the state of the art radar-camera fusion baseline AVOD, the state-of-the-art LiDAR-camera fusion paradigm "point painting" on radar point clouds $PP-radar\ (painted)$, and the straightforward late fusion $PP-mono\ fusion$. AVOD network is able to provide detections on cars, but its performance on VRUs is poor. In fact, AVOD is outperformed even by the PointPillar network trained on radar point cloud alone $PP-radar$. The BEV (bird's eye view) input feature maps in AVOD are generated by projecting slices of the point cloud (sliced by a set of planes parallel to the input ground plane) onto the horizontal plane of the camera coordinate frame. This approach relies heavily on the quality of input ground plane estimation and also the assumption that horizontal plane of camera is somewhat parallel to the actual ground plane. We generated ground planes for VoD dataset as explained in Appendix section E by LiDAR point cloud, semantic masks and RANSAC algorithm to provide a reasonable input to AVOD network. However, the camera coordinate frame of our sensor setup has a pitch angle of more than 7 degrees which might have impacted the network's ability. Furthermore, Astyx implementation of AVOD does not use Doppler information at all, which was repeatedly shown to be a strong classification feature in literature [1] [4]. It is also not trivial to directly extend methods that use BEV representation as input to effectively encode Doppler information, unlike methods using point-representation (e.g. PointPillars).

The performance gain of $PP-RAID\ fusion$ over $PP-radar\ (painted)$ shows the benefits of providing a point cloud representation to image via RAID - a superior representation with both semantic and spatial cues. Painted radar target representation on the other hand is limited by the sparseness of the point cloud and also the accuracy of azimuth and elevation. A 3D radar target projected onto the image for sampling semantic class may not "fall" on the reflected object due to measurement inaccuracies and ambiguities. Such misalignment could result in a radar target being painted with incorrect semantics, which in turn feeds incorrect information into the downstream pipeline.

While $PP-RAID\ (painted)$ performs better than $PP-mono\ (painted)$, the detection performance is still inferior to models trained on radar and LiDAR point clouds. This suggests that it is not enough just to exploit radar's ranging capabilities via a refined depth map - Doppler and RCS futures are also needed. On the other hand, a standalone Stage-2 implementation i.e., $PP-mono\ fusion$ does use a point cloud representation from monocular depth map with semantic painting, but this is generated without any learning from radar depth cues. The significant gain of $PP-RAID\ fusion$ over $PP-mono\ fusion$ shows that the RAID depth refinement helped propagate object depth cues from radar to produce a better point cloud, and that such depth refinement is also beneficial for the object detection task.

We also compared object detection performance on point clouds generated with different sensors. As expected, the performance of networks using radar or camera data only (i.e., $PP-radar\ (5\ frames)$, $PP-mono\ (painted)$) was poor compared to those trained on high-end LiDAR data with 64 layers, confirming the need for fusion. Our proposed method significantly outperformed both single modality approaches, demonstrating that it was able to utilize complementary information from the two sensors.

Comparing $PP-RAID\ fusion$ and $PP-LiDAR$ shows

that the proposed multi-level fusion of radar and camera performs comparably to the detections from a 64-layer LiDAR point cloud especially in the Driving Corridor. In the entire annotated region, performance is still comparable for pedestrians and even slightly outperforms it for cyclists. However, LiDAR still significantly surpasses our fusion in the detection of cars.

As expected, fusion of LiDAR and camera data via painting $PP - LiDAR\ (painted)$ has the highest performance in most metrics. However, our fusion method even showcases comparable performance to this LiDAR-camera combination for certain classes and regions, e.g., cyclists both in the whole annotated area and in the Driving Corridor, and cars in the driving corridor, see underlined values in Table I. Our fusion of radar and camera demonstrated comparable detection performance with LiDAR-camera fusion in driving corridor. This indicates the capability of radar-camera fusion as a redundant sensor setup for object detection in urban scenarios, i.e., it could substitute a single LiDAR sensor or LiDAR-camera fusion in case of sensor failure or adverse conditions.

An extensive evaluation on multiple 3D IoU thresholds presented in Figure 5 shows that the proposed fusion approach brings consistent improvement not only for the selected IoU thresholds (25% for pedestrians and cyclists, 50% for cars), but also for multiple overlap thresholds from 10% to 90%, for all classes and for both evaluated regions (i.e., entire annotated area and driving corridor).

Furthermore, our approach outperformed both the LiDAR networks in orientation estimation performance by a large margin in driving corridor. As explained in [1], additional radial velocity information of radar provides strong orientation cues that are not available with LiDAR. Further coupled with strong spatial cues from image based point cloud generated via RAID, the fused point cloud is rich with orientation cues, as seen e.g. for cars in the driving corridor, where the proposed fusion significantly improved compared to the radar only detection performance.

The overall better performance of the two networks using LiDAR can be attributed to the much higher point density of the specific type of 64-layer LiDAR sensor used (average number of points in the annotated area: LiDAR: 21344, radar: 216 for a single scan, or $\sim$ 1000 points for five). Furthermore, the high viewpoint of the LiDAR sensor, on the roof of the car, incurs less occlusion while scanning and thereby benefits object detection performance. This could be a possible reason for significantly higher performance of LiDAR based networks on cars in the entire annotated area. The combination of radar and camera sensors comes, however, with clear advantages in terms of cost and ease of integration in stock vehicles.

The modularity of the proposed pipeline provides several starting points for improvement. For example, it allows experimentation with different or better (pre-)trained networks at all levels: monocular depth estimation and panoptic segmentation networks for improved input signals, 2D backbones for the depth refinement network, and 3D object detection networks to process point cloud information. Thus, we can use both upcoming state of the art research and large datasets even if they do not feature radar data. In addition, we would like to emphasize that the RAID - depth refinement network is cross-modal supervised, hence it does not require any kind of manual annotation, making it possible to quickly acquire and use large amounts of training data.

Besides exploiting the modularity, RAID network performance can be improved further by looking into possibility of class specific losses. For example, by introducing class-specific spatial constraints for the sizes of classes from different classes, we could improve depth consistency within instance masks. Also, the probabilistic output from the ordinal regression layer could be better utilized to estimate the uncertainty of the predicted depth of a pixel and passed on for subsequent processing steps as a feature of the lifted pixels. Alternatively, we can also "sample" depth at different depth bins estimated at different confidence thresholds per pixel.

Furthermore, re-introducing RGB image as additional channel input to the RAID network may help in the depth refinement process as the network can learn features in addition to depth and semantics. The increased input channels may also require a larger backbone network. However, thanks to our modular approach, this is a straightforward swap.

Moreover, the semantic painted point cloud generated via RAID can be used for other perception tasks, such as free-space estimation and ground plane estimation. It can be easily implemented by replacing the object detection network in Stage-2 with the appropriate task head.

While modular network design provides flexibility and agility, it also brings the concern of domain gap. However, we can use self-supervision and cross-modal supervision schemes wherever applicable to alleviate this problem. For example, monocular depth estimation network can be fine-tuned or trained using video or stereo image pair or LiDAR data as groundtruth signal to provide a better input to the RAID network.

As mentioned earlier, the LiDAR sensor in the dataset has a better viewpoint, which can benefit object detection, especially at longer distances. This can be experimentally tested by removing LiDAR points that are occluded when projected to camera image plane. This would essentially simulate the same LiDAR sensor not being on top of the vehicle, but at the position of the camera sensor. The two filtering methods introduced for dealing with conflicting depth cues in image plane can be directly used for this task.

VI. CONCLUSIONS

In this paper we introduced a modular, generic, and multi-level fusion of 3+1D radars and monocular camera for the purpose of road user detection in urban scenarios. The method consists of two stages. In Stage-1, a novel "depth-refinement network" called RAID takes monocular depth map, panoptic masks from pre-trained networks along with radar depth map to generate a refined depth output and lift pixels to 3D metric space. We used state-of-the-art methods for the sub-tasks in our proposed fusion architecture while also exploiting large uni-modal and non-annotated datasets. The resulting point cloud is then painted with semantic information and fed as an input to stage 2, where it is fused again with the 3+1D point cloud by concatenating the two lists of point cloud. elaborate about the pointpillar network training

In extensive experiments on the View-of-Delft dataset, we have demonstrated that the proposed approach significantly outperforms the radar-camera fusion based 3D detection methods (proposed method vs. best baseline: 53.6 mAP vs. 50.8 mAP). Furthermore, it yielded comparable performance to LiDAR and state-of-the-art LiDAR-camera fusion based detectors, especially when evaluated in the safety critical Driving Corridor (80.5 mAP vs. 81.6 mAP).

The modular setup of the pipeline makes it flexible to incorporate upcoming state of the art methods for the camera based inputs, and also allows to further train/fine-tune Stage-1 in a self-supervised/cross-sensor supervised scheme without any need for manual annotations. The proposed fusion approach can thus help alleviate the problem of limited multi-modal datasets from the perspective of size and annotations. The architecture is generic and can also accommodate LiDAR point cloud in stage 2 as input for researchers interested in different combination of sensors for the downstream detection task.

Also, class specific loss functions and pixel sampling may help improve the performance on harder classes like pedestrians.

Furthermore, the output of Stage-1, i.e., the semantic painted point cloud from RAID has the potential for further use cases like free road estimation, ground plane estimation, etc.

Lastly, the point cloud representation of monocular image and 3+1D radar provides the opportunity for extensive research into different point cloud fusion paradigms in Stage-2. Instead of the proposed early fusion between the 3+1D radar point cloud and RAID generated point cloud, feature level fusion of BEV maps or sparse voxel features is research worthy.

APPENDIX A
PRE-PROCESSING OF THE RADAR POINT CLOUD

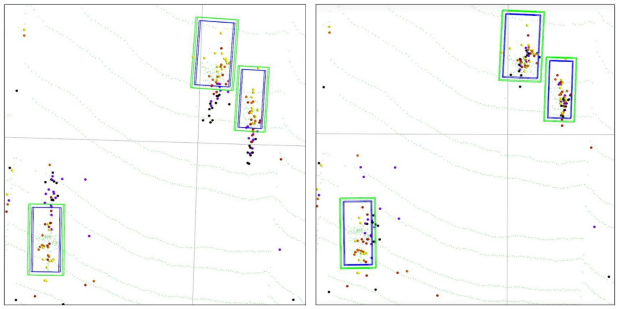
LiDAR point clouds (even the ones of lower tier sensors, such as a 16-layer Velodyne) provide a relatively noise free, accurate scan of the environment with a consistent pattern. This makes them highly suitable for depth completion tasks by fusion with mono-camera camera input. Compared to LiDAR, a 3+1D radar point cloud is noisy as it suffers from interference and multi-path propagation. Operating in the mm-wavelength spectrum leads to specular reflection which in turn results in a small fraction of signal to reach the receiving antenna. This means that the availability of points depends on the radar cross section (RCS) and shape properties of the object. Due to this and the "lack" of a scanning pattern, the points returned in each scan are sparse with heterogeneous distribution. Often researchers accumulate multiple previous sweeps to improve the density and scene information. However, we observed that after performing ego-motion compensation of the past frames, a "trail" like phenomenon (refer to Figure 6a) can be observed in the point cloud due to the motion of corresponding objects in-between the scans, which can lead to "outdated" depth cues and thus, faulty depth estimation. In the following, we present a pipeline of pre-processing steps that are aimed to attain a reliable and dense point cloud from radar with the intention to provide consistent depth cues for RAID network by addressing both the challenge of sparsity and the challenge of moving objects.

Specifically, we first perform "Target Propagation" to compensate for the (radial) motion of the object between consequent radar scans. This is followed by "Spatial-Temporal Filtering" to deal with noise and inconsistent targets. To further improve the density of the point cloud, we then do "Sensor-aware up-sampling". Finally, for dynamic targets (above a certain Doppler threshold) we also carry out "Vertical sampling" with the intuition that these targets correspond to road users and thereby selectively increasing point cloud density at instance level. We emphasize that the listed steps are followed in the order stated and should not be shuffled.

Target Propagation: Let v_{rc} be the compensated radial velocity of a target h from scan $N-a$, where N is the current scan's time id, and a indicates the preceding count. Given the radar point's position vector $\tilde{p}_h$ in radar coordinate frame and a scanning rate f, the target h can be propagated to its "corrected/predicted" position $\tilde{p}'_h$ in a straightforward approach:

$$\tilde{p}'_h = -\tilde{p}_h + \left(v_{rc} \frac{\tilde{p}_h}{\mid p_h \mid} \left(\frac{a}{f} \right) \right). \tag{5}$$

Refer to Figure 6 for visualization of radar targets before and after target propagation for cyclists. It is to be noted that this approach only compensates for the motion of object in radial direction, and the lateral shift is still relevant. However, moving cars and cyclists mostly travel in the longitudinal direction of the ego-vehicle and hence compensation along radial direction alleviates the concern significantly for these classes. It may not be as effective for pedestrian class due to their stochastic moving patterns.



(a) Without target propagation (b) With target propagation

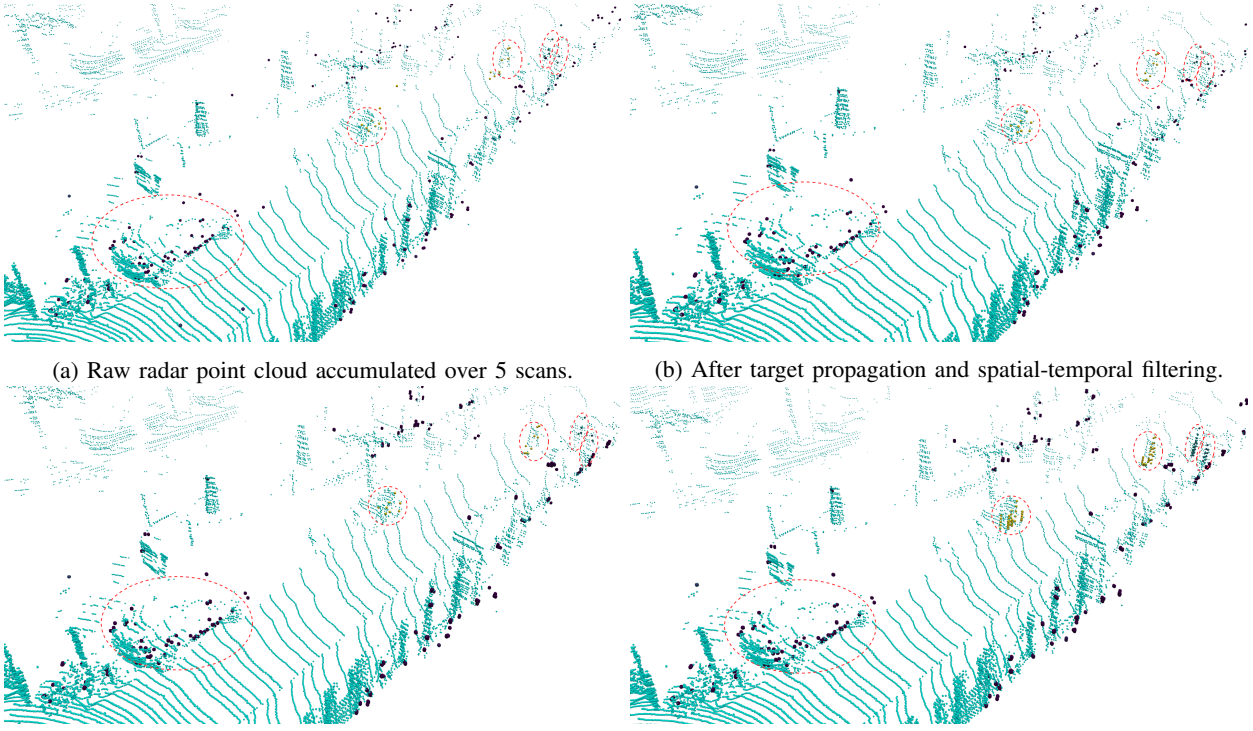
Fig. 6: Visualization of objects' trails (a) due to their motion between accumulated scans and after correction with target propagation (b). Blue boxes marks the ground truth annotation for three cyclists in the scene, while green ones are the same boxes "enlarged" by 0.2 meters along each dimension. Different colors of the targets indicate different source scans. Grey points represent the LiDAR point cloud. After compensation, most targets fit the ground truth boxes.

Spatial-Temporal Filtering: In Pointillism [69], authors filter the raw radar point cloud by checking for spatial coherence in output from two spatially separated radars, and for temporal coherence by "tracking" the individual targets across scans using Kalman filter. However, our sensor setup consists of a single frontal radar. Yet, inspired by their underlying intuition, we propose a simple vote based approach to filter targets based on spatial and temporal consistency.

First, we fit a k-dimensional tree (with a radius r) on the input radar point cloud to enable a neighbor search. For each radar target, we count the number of neighbor targets irrespective of the source frame to get the "spatial" vote and the number of unique scans that these neighboring targets originate from represent the "temporal" vote. For a target to qualify the filter, the sum of both the votes should be above a certain threshold $\mathcal{M}$. r and $\mathcal{M}$ are hyper-parameters and for the experimental results presented in this work, values of 0.5 meters and 3 are chosen respectively. For a visual representation, see Figure 7b

Sensor-aware Up-sampling: While the range accuracy of radar is excellent (0.02 meters), the angular accuracy for azimuth and elevation is limited, especially at the edges of field of view. The ZF radar fitted in our ego-vehicle reports (in center of FoV) an elevation accuracy of 0.3 degree for elevation and 0.15 degree for azimuth. For every radar target that passed the spatial-temporal filter, we sample 3 new points from a 2D (azimuth, elevation) Gaussian distribution with the mean and standard deviation as the azimuth/elevation angle and accuracy respectively. See Figure 7c for a visual representation of this step.

Vertical Expansion of Radar Targets: To deal with the sparse nature of radar output and the lack of elevation information, researchers have projected radar targets to the camera frame as vertical pillars [7] or as circles [11] for fusion of 2+1D radar with monocular image. However, such methods may act as noise by providing conflicting depth cues between foreground and background objects. Thanks to the



(a) Raw radar point cloud accumulated over 5 scans.

(b) After target propagation and spatial-temporal filtering.

(c) After sensor-aware up-sampling

(d) After vertical sampling i.e., fully pre-processed radar targets

Fig. 7: Radar point cloud visualization at various stages of pre-processing. Points in cyan represent LiDAR points to comprehend the scene context. Radar targets are colored based on the compensated Doppler value. Note that the cyclists and pedestrians are moving while the car is parked (all the instances are highlighted in dotted ellipses).

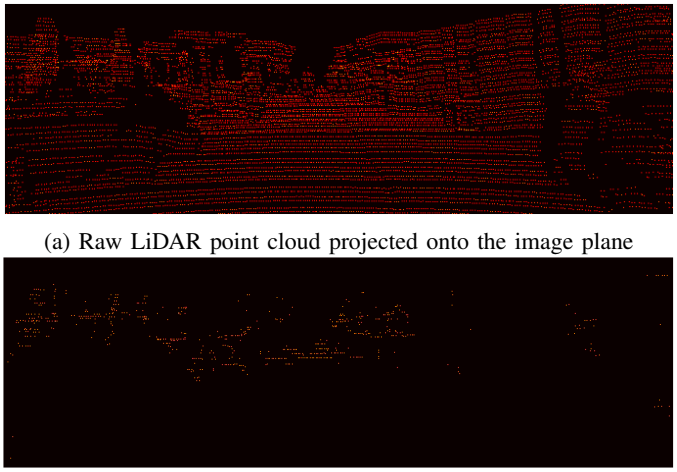
elevation information in the 3+1D radar, the core idea can be implemented without causing such signal noise.

To be specific, for each dynamic target, a fixed number (hyper-parameter) of points are sampled along the Z-axis of the radar co-ordinate frame between the ground and the target itself. The ground plane is assumed to be 0.5 meters below the X-Y plane of the radar co-ordinate system based on our ego-vehicle setup. This step is done to inject further depth cues for moving objects and assumes that there are no "floating" objects, i.e., an elevated radar target's depth information is relevant for pixels below it (and above the ground plane).

Figure 7d shows the radar point cloud at the end of the four pre-processing pipelines discussed above.

Appendix B
Handling Conflicting Depth Cues

In our proposed RAID network, we use LiDAR point cloud as the supervising signal by projecting it onto the image plane. However, due to the positioning of this sensor on the roof of our ego-vehicle, it generates conflicting depth cues when seen from camera view, i.e., some points of the background are projected onto foreground objects in the camera view. To illustrate this phenomenon, refer to Figure 8. Even the radar depth maps as input channels suffer from this phenomenon if not addressed, because accumulation of frames occurs at different ego-vehicle positions (when in motion) and thus previously visible, farther points could be occluded when viewed from an other, subsequent position.



(a) Raw LiDAR point cloud projected onto the image plane

(b) Points removed by the kernel and then DBSCAN cluster filter

Fig. 8: LiDAR points propagating conflicting depth cues due to perspective-view difference between camera and LiDAR scans.

In order to tackle this problem, we first run a "kernel" based filter of size $J \times J$ (J is a hyper-parameter) which iterates throughout the depth images and looks for a conflict in the Z-order among the pixels within the kernel with a LiDAR/radar projection. The conflict is a simple outlier detection based on depth value stored in these pixels. If a conflict arises over any kernel patch of the image, it is resolved by nullifying

the pixel(s) reporting a "farther" depth compared to their neighbors in the kernel.

While the kernel filter does detect and resolve some conflicts, due to small kernel size, the filtering is only done considering a "local" neighborhood. However, at an instance level, the concern still persists. Further, the instance level losses in RAID network depend on inferred panoptic masks (from a pre-trained network) to obtain pixels corresponding to each instance. As shown in Figure 9 for both radar and LiDAR point clouds, both noisy masks and perspective-view projection result in poor supervision signal from LiDAR. In other words, for a given inferred instance mask, it is highly likely that the corresponding ground truthLiDAR scan contains points that belong to background scene. Since the instance level loss penalizes performance for all pixels where ground truth is available in an instance, the network is penalized even when regressing the instance depth correctly due to these background LiDAR points on the instances.

To alleviate this concern, we extract "sub-"point clouds falling on each instance of target class and run a DBSCAN clustering algorithm to filter the outliers. Thanks to the semantic mask information, we can set class specific clustering parameters, i.e., radial threshold of 0.4, 0.6 and 0.7 meters for pedestrians, cyclists and cars with a minimum of three neighbors needed to retain a point/target. Figure 9 shows the output of this filtering step. The result of this clustering is at least one cluster of points per instance mask, of which we keep only the largest, since we assume that it belongs to the road user.

Appendix C
RAID: Loss Function

In continuation of section III-B, the mathematical expressions for $\Psi(w_i, h_i, \chi, \Theta)$ - per pixel ordinal loss, $\hat{l}$ - predicted discrete ordinal label and $\hat{d}$ - the decoded depth from $\hat{l}$ are provided here; all introduced first in DORN [24].

Ψ is defined as a negative log-likelihood estimation function:

$$\sum_{k=0}^{K-1} Z^k_{(w,h)} \log\left(\mathcal{P}^k_{(w,h)}\right) + \left(1 - Z^k_{(w,h)}\right)\left(1 - \log\left(\mathcal{P}^k_{(w,h)}\right)\right), \tag{6}$$

where $Z^k_{(w,h)}$ is defined as a binary flag indicating that the ground truth depth at pixel (w, h) falls beyond the ordinal bin k, while $\mathcal{P}^k_{(w,h)}$ represents the inferred probability of the same:

$$Z^k_{(w,h)} = \left(l_{(w,h)} > k \mid d_{(w,h)}\right), \tag{7}$$

$$\mathcal{P}^k_{(w,h)} = P\left(\hat{l}_{(w,h)} > k \mid \chi, \Theta\right). \tag{8}$$

While generating $Z^k_{(w,h)}$ from ground truth depth map is straightforward, $\mathcal{P}^k_{(w,h)}$ is calculated from the output of ordinal regression layer in the following way:

$$\mathcal{P}^k_{(w,h)} = \frac{e^{y_{(w,h,2k+1)}}}{e^{y_{(w,h,2k)}} + e^{y_{(w,h,2k+1)}}}. \tag{9}$$

Note that this loss in the original DORN publication is written in a different way by removing $Z^k_{(w,h)}$ and splitting the summation into two parts. We formulate the loss here in a more generic way for better comprehension. Defining the ground truth and prediction probabilities this way enforces an "ordinal relation" in the negative log-likelihood function, which, in its native implementation, would have been a classification task without any correlation among the distance bins/classes.

During inference, the ordinal depth label $\hat{l}$ is generated by counting the bins up to which the network is confident above the threshold 0.5 that the depth of the pixel is beyond a given bin:

$$\hat{l}_{(w,h)} = \sum_{k=0}^{K-1} \eta\left(\mathcal{P}^k_{(w,h)} >= 0.5\right). \tag{10}$$

The metric depth can then be decoded from the SID intervals (t) by the average of the bin boundaries in the following way:

$$\hat{d}_{(w,h)} = \frac{t_{\hat{l}_{(w,h)}} + t_{\hat{l}_{(w,h)+1}}}{2} - \xi, \tag{11}$$

where ξ is an offset introduced to the bins' start and end values while generating the intervals.

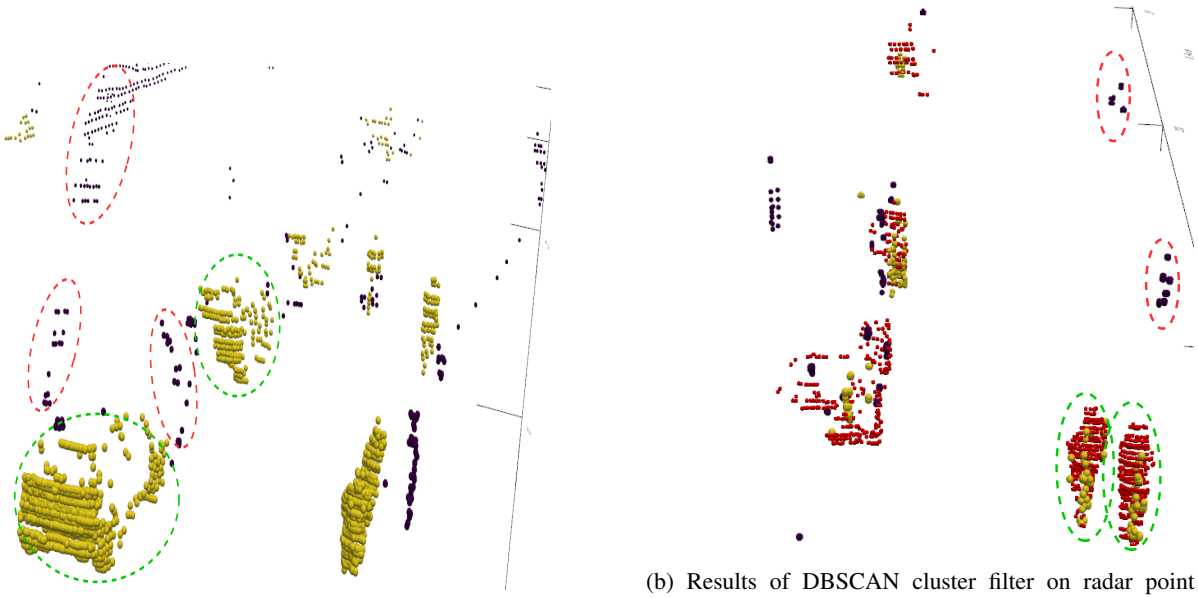
Appendix D
RAID: Qualitative Results

RAID effectively addresses the often mentioned [70][59] local misalignment problem in point clouds generated from image pixels. Earlier works on giving point cloud representation to images such as Mono3D-PL [70] and Pseudo-LiDAR++ [59] approached this problem using bounding box consistency and deterministic depth propagation from sparse LiDAR point cloud respectively. In contrast, our method extract depth cues from the radar point cloud in a deep learning approach. As seen in Figure 10, pixels lifted through RAID consistently overlap with the LiDAR point cloud. Images in the first row correspond to two car instances, bottom left shows a cyclist in BEV, and finally bottom right shows two pedestrians 40 meters away in front of the ego-vehicle. Note that while the monocular depth failed to infer useful (i.e., nearby) depth for any of the pixels for these pedestrians, RAID succeeded in lifting these instances to 3D space.

Appendix E
Ground-plane Estimation for AVOD

AVOD is the chosen point-cloud and monocular camera fusion network for 3+1D radar-camera fusion by Astyx [48], which, to the best of our knowledge, is the only attempt in literature. To implement this network on the VoD dataset, ground planes are expected as an input in addition to the sensor data. To address this need, we used a simple and effective approach.

Using the inferred semantic masks, LiDAR points corresponding to the semantic class "road" are extracted. Subsequently, a RANSAC algorithm is used to fit a plane to the extracted LiDAR points, see Figure 11. Qualitative evaluation on multiple frames demonstrated the efficacy of this method.



(a) LiDAR points in a scene corresponding to specific instances of interest. Yellow points represent the retained clusters after DBSCAN while those in black are outliers. For two car instances on the left, points highlighted in dotted green ellipses indicate the retained points after filtering while ellipses in red indicate the occluded points in image view that are now filtered out.

(b) Results of DBSCAN cluster filter on radar point cloud in another scene. Red indicates filtered LiDAR points, yellow and black indicate retained and outlier radar targets respectively. For the two pedestrian instances on the right, ellipses in green indicate the retained points after the DBSCAN filtering and the points in red ellipses correspond to the background scene that are removed which otherwise would have propagated conflicting depth cues to these instances when projected onto image plane.

Fig. 9: DBSCAN cluster based filtering for LiDAR and radar point clouds to tackle conflicting depth cues when projecting the points onto image plane.

Method	**Input to Detection Network**	**Entire annotated area**					**In Driving Corridor**				
		Car	**Pedestrian**	**Cyclist**	**mAP**	**mAOS**	**Car**	**Pedestrian**	**Cyclist**	**mAP**	**mAOS**
PP - RAID fusion w/o radar preprocessing	Painted PL from RAID + radar targets	51.1	49.6	56.8	52.5	45.7	**89.8**	67.4	76.7	78.0	74.9
PP - RAID fusion	Painted PL from RAID + radar targets	**51.8**	**50.4**	**56.9**	**53.0**	**48.6**	89.6	**68.6**	**83.3**	**80.5**	**77.7**

TABLE II: Comparison of detection performance between a PointPillar network trained on RAID fusion point cloud with and without radar point cloud pre-processing in Stage-1. Detection performance on cyclists in the driving corridor improved significantly (by more than 7 mAP) and orientation estimation of all the three classes improved by more than 2 mAOS in driving corridor, and in the entire annotated region.

APPENDIX F
FURTHER ABLATION EXPERIMENTS

In a further ablation study, the role of the radar pre-processing pipeline discussed in section A. We trained a network on RAID fusion point cloud where no such pre-processing of the 3+1D radar point cloud is done before passed as input to the RAID network. Table II shows that the pre-processing pipeline indeed helps the overall detection performance, especially for cyclists in the driving corridor.

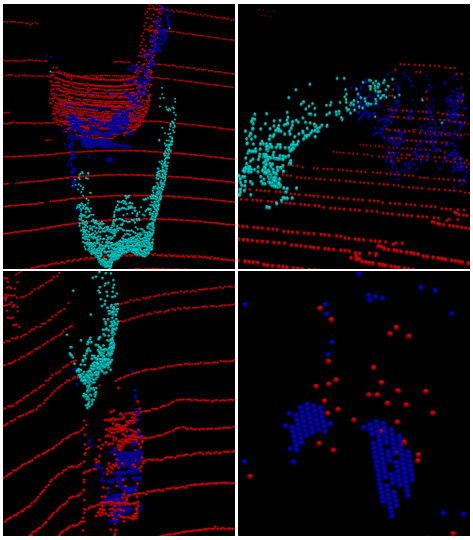


Fig. 10: Localization provided by point cloud from RAID (blue) vs. point cloud generated from monocular depth (cyan) vs. LiDAR point cloud (red) for two cars (top row), a cyclist (bottom left), and two pedestrians (bottom right) visualized in arbitrary view points. RAID matches the ground truth LiDAR point clouds significantly better than the monocular depth based point cloud.

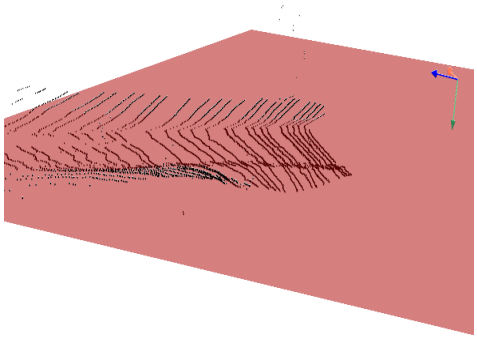


Fig. 11: Ground plane estimation using a LiDAR point cloud and a semantic mask. The orthogonal axes represent the position and orientation of camera fitted on the ego-vehicle. The visualized point cloud is “extracted” by keeping only the LiDAR points that fall on pixels with the semantic class“road” when projected onto image plane. The red plane represents the RANSAC-fitted ground plane using the extracted set of points.

REFERENCES

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

[2] T. Wang, Z. Xinge, J. Pang, and D. Lin, "Probabilistic and geometric depth: Detecting objects in perspective," in *Conference on Robot Learning*. PMLR, 2022, pp. 1475–1485.

[3] L. Jing, R. Yu, H. Kretzschmar, K. Li, R. Qi, H. Zhao, A. Ayvaci, X. Chen, D. Cower, Y. Li, Y. You, H. Deng, C. Li, and D. Anguelov, "Depth estimation matters most: Improving per-object depth estimation for monocular 3d detection and tracking," in *IEEE International Conference on Robotics and Automation*, May 2022.

[4] A. Palffy, J. Dong, J. F. P. Kooij, and D. M. Gavrila, "CNN Based Road User Detection Using the 3D Radar Cube," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1263–1270, 2020.

[5] F. Engels, P. Heidenreich, M. Wintermantel, L. Stäcker, M. Al Kadi, and A. M. Zoubir, "Automotive radar signal processing: Research directions and practical challenges," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 4, pp. 865–878, 2021.

[6] R. Nabati and H. Qi, "Centerfusion: Center-based radar and camera fusion for 3d object detection," in *IEEE Winter Conference on Applications of Computer Vision*, 2021, pp. 1526–1535.

[7] F. Nobis, M. Geisslinger, M. Weber, J. Betz, and M. Lienkamp, "A deep learning-based radar and camera sensor fusion architecture for object detection," in *2019 Sensor Data Fusion: Trends, Solutions, Applications*, 2019, pp. 1–7.

[8] Y. Wang, Z. Jiang, Y. Li, J.-N. Hwang, G. Xing, and H. Liu, "Rodnet: A real-time radar object detection network cross-supervised by camera-radar fused object 3d localization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 4, pp. 954–967, 2021.

[9] S. Campbell, N. O'Mahony, L. Krpalcova, D. Riordan, J. Walsh, A. Murphy, and C. Ryan, "Sensor technology in autonomous vehicles : A review," in *Irish Signals and Systems Conference*, 2018, pp. 1–4.

[10] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 11 618–11 628, 2020.

[11] Y. Long, D. Morris, X. Liu, M. Castro, P. Chakravarty, and P. Narayanan, "Radar-camera pixel depth association for depth completion," in *Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 502–12 511.

[12] M. Meyer and G. Kuschk, "Automotive radar dataset for deep learning based 3d object detection," in *European Radar Conference*, 2019, pp. 129–132.

[13] S. Vora, A. H. Lang, B. Helou, and O. Beijbom, "PointPainting: Sequential Fusion for 3D Object Detection," in *Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4603–4611.

[14] S. Xu, D. Zhou, J. Fang, J. Yin, Z. Bin, and L. Zhang, "Fusionpainting: Multimodal fusion with adaptive attention for 3d object detection," in *IEEE International Intelligent Transportation Systems Conference*, 2021, pp. 3047–3054.

[15] L. Wang, T. Chen, C. Anklam, and B. Goldluecke, "High Dimensional Frustum PointNet for 3D Object Detection from Camera, LiDAR, and Radar," in *IEEE Intelligent Vehicles Symposium*, 2020, pp. 1621–1628.

[16] X. Wu, L. Peng, H. Yang, L. Xie, C. Huang, C. Deng, H. Liu, and D. Cai, "Sparse fuse dense: Towards high quality 3d detection with depth completion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2022, pp. 5418–5427.

[17] T. Yin, X. Zhou, and P. Krähenbühl, "Multimodal Virtual Point 3D Detection," *Conference and Workshop on Neural Information Processing Systems*, 2021.

[18] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 12 689–12 697, 2019.

[19] F. Nobis, E. Shafiei, P. Karle, J. Betz, and M. Lienkamp, "Radar voxel fusion for 3d object detection," *Applied Sciences*, vol. 11, no. 12, p. 5598, 6 2021.

[20] C. Reading, A. Harakeh, J. Chae, and S. L. Waslander, "Categorical depth distribution network for monocular 3d object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8551–8560.

[21] X. Ye, L. Du, Y. Shi, Y. Li, X. Tan, J. Feng, E. Ding, and S. Wen, "Monocular 3d object detection via feature domain adaptation," in *European Conference on Computer Vision*. Berlin, Heidelberg: Springer-Verlag, 2020, p. 17–34.

[22] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8437–8445.

[23] C. Godard, O. M. Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6602–6611.

[24] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2002–2011.

[25] E. Simsar, E. P. Örnek, F. Manhardt, H. Dhamo, N. Navab, and F. Tombari, "Object-aware monocular depth prediction with instance convolutions," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5389–5396, 2022.

[26] X. Wang, W. Yin, T. Kong, Y. Jiang, L. Li, and C. Shen, "Task-aware monocular depth estimation for 3d object detection," in *AAAI Conference on Artificial Intelligence*, 2020, pp. 12 257–12 264.

[27] T. Roddick, A. Kendall, and R. Cipolla, "Orthographic feature transform for monocular 3d object detection," *British Machine Vision Conference*, 2019.

[28] T. Wang, X. Zhu, J. Pang, and D. Lin, "Probabilistic and geometric depth: Detecting objects in perspective," in *Conference on Robot Learning*, 2021.

[29] ——, "FCOS3D: Fully Convolutional One-Stage Monocular 3D Object Detection," in *IEEE International Conference on Computer Vision Workshops*, 2021, pp. 913–922.

[30] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia, "Geonet: Geometric neural network for joint depth and surface normal estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 283–291.

[31] J. H. Lee, M.-K. Han, D. W. Ko, and I. H. Suh, "From big to small: Multi-scale local planar guidance for monocular depth estimation," *arXiv preprint arXiv:1907.10326*, 2019.

[32] L. Huynh, P. Nguyen-Ha, J. Matas, E. Rahtu, and J. Heikkilä, "Guiding monocular depth estimation using depth-attention volume," in *European Conference on Computer Vision*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., Cham, 2020, pp. 581–597.

[33] S. Lee, J. Lee, B. Kim, E. Yi, and J. Kim, "Patch-wise attention network for monocular depth estimation," *AAAI Conference on Artificial Intelligence*, vol. 35, no. 3, pp. 1873–1881, May 2021.

[34] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2021, pp. 4009–4018.

[35] Y. Cao, Z. Wu, and C. Shen, "Estimating Depth From Monocular Images as Classification Using Deep Fully Convolutional Residual Networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 11, pp. 3174–3182, 2018.

[36] S. Lee, S. Im, S. Lin, and I. S. Kweon, "Learning monocular depth in dynamic scenes via instance-aware projection consistency," *AAAI Conference on Artificial Intelligence*, vol. 35, no. 3, pp. 1863–1872, 5 2021.

[37] M. Klingner, J.-A. Termöhlen, J. Mikolajczyk, and T. Fingscheidt, "Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance," in *European Conference on Computer Vision*. Berlin, Heidelberg: Springer-Verlag, 2020, p. 582–600.

[38] M. Ochs, A. Kretz, and R. Mester, "Sdnet: Semantically guided depth estimation network," in *DAGM German Conference on Pattern Recognition*. Berlin, Heidelberg: Springer-Verlag, 2019, p. 288–302.

[39] S. Qiao, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3996–4007.

[40] Z. Zhang, Z. Cui, C. Xu, Y. Yan, N. Sebe, and J. Yang, "Pattern-affinitive propagation across depth, surface normal and semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4101–4110.

[41] J.-T. Lin, D. Dai, and L. V. Gool, "Depth estimation from monocular images and sparse radar data," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 10 233–10 240.

[42] C.-C. Lo and P. Vandewalle, "Depth estimation from monocular images and sparse radar using deep ordinal regression network," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 3343–3347.

[43] A. Palffy, J. F. P. Kooij, and D. M. Gavrila, "Occlusion aware sensor fusion for early crossing pedestrian detection," in *IEEE Intelligent Vehicles Symposium*, 2019, pp. 1768–1774.

[44] D. Kang and D. Kum, "Camera and radar sensor fusion for robust vehicle localization via vehicle part localization," *IEEE Access*, vol. 8, pp. 75 223–75 236, 2020.

[45] V. John, M. K. Nithilan, S. Mita, H. Tehrani, R. S. Sudheesh, and P. P. Lalu, "So-net: Joint semantic segmentation and obstacle detection using deep fusion of monocular camera and radar," in *Image and Video Technology*, J. J. Dabrowski, A. Rahman, and M. Paul, Eds. Cham: Springer International Publishing, 2020, pp. 138–148.

[46] A. Sengupta, A. Yoshizawa, and S. Cao, "Automatic radar-camera dataset generation for sensor-fusion applications," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2875–2882, 2022.

[47] J. Ku, A. Harakeh, and S. L. Waslander, "In defense of classical image processing: Fast depth completion on the cpu," in *Conference on Computer and Robot Vision*, 2018, pp. 16–22.

[48] M. Meyer and G. Kuschk, "Deep learning based 3D object detection for automotive radar and camera," *European Radar Conf.*, pp. 133–136, 2019.

[49] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–8.

[50] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3d object detection," in *European Conference on Computer Vision*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 663–678.

[51] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D Object Detection Network for Autonomous Driving," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6526–6534.

[52] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun, "Multi-task multi-sensor fusion for 3d object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7337–7345.

[53] X. Du, M. H. Ang, S. Karaman, and D. Rus, "A general pipeline for 3d detection of vehicles," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 3194–3200.

[54] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 918–927.

[55] Z. Wang and K. Jia, "Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 1742–1749.

[56] L. X. 0003, C. Xiang, Z. Yu, G. Xu, Z. Yang, D. Cai, and X. H. 0001, "Pi-rcnn: An efficient multi-sensor 3d object detector with point-based attentive cont-conv fusion module," in *AAAI Conference on Artificial Intelligence*, 2020, pp. 12 460–12 467.

[57] W. Zheng, H. Xie, Y. Chen, J. Roh, and H. Shin, "Pifnet: 3d object detection using joint image and point cloud features for autonomous driving," *Applied Sciences*, vol. 12, no. 7, 2022.

[58] C. Wang, C. Ma, M. Zhu, and X. Yang, "Pointaugmenting: Cross-modal augmentation for 3d object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 789–11 798.

[59] Y. You, Y. Wang, W.-L. Chao, D. Garg, G. Pleiss, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-LiDAR++: Accurate Depth for 3D Object Detection in Autonomous Driving," in *ICLR*, 2020.

[60] X. Lyu, L. Liu, M. Wang, X. Kong, L. Liu, Y. Liu, X. Chen, and Y. Yuan, "Hr-depth: High resolution self-supervised monocular depth estimation," *AAAI Conference on Artificial Intelligence*, vol. 35, no. 3, pp. 2294–2301, May 2021.

[61] R. Mohan and A. Valada, "Efficientps: Efficient panoptic segmentation," *International Journal of Computer Vision*, 2021.

[62] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger, "Sparsity invariant cnns," in *International Conference on 3D Vision*, 2017.

[63] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Conference on Computer Vision and Pattern Recognition*, 2016.

[64] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *IEEE International Conference on Computer Vision*, 2015, pp. 2650–2658.

[65] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *IEEE International Conference on Computer Vision*, 2019, pp. 1314–1324.

[66] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

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

[68] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[69] K. Bansal, K. Rungta, S. Zhu, and D. Bharadia, *Pointillism: Accurate 3D Bounding Box Estimation with Multi-Radars*. New York, NY, USA: Association for Computing Machinery, 2020, p. 340–353.

[70] X. Weng and K. Kitani, "Monocular 3D Object Detection with Pseudo-LiDAR Point Cloud," in *IEEE International Conference on Computer Vision Workshop*, 2019, pp. 857–866.