RadarNet: Exploiting Radar for Robust Perception of Dynamic Objects

Bin Yang[1,2⋆], Runsheng Guo[3⋆], Ming Liang[1], Sergio Casas[1,2], Raquel Urtasun[1,2]

[1]Uber Advanced Technologies Group, [2]Univeristy of Toronto, [3]University of Waterloo
{byang10,ming.liang,sergio.casas,urtasun}@uber.com,
r9guo@edu.uwaterloo.ca

Abstract. We tackle the problem of exploiting Radar for perception in the context of self-driving as Radar provides complementary information to other sensors such as LiDAR or cameras in the form of Doppler velocity. The main challenges of using Radar are the noise and measurement ambiguities which have been a struggle for existing simple input or output fusion methods. To better address this, we propose a new solution that exploits both LiDAR and Radar sensors for perception. Our approach, dubbed RadarNet, features a voxel-based early fusion and an attention-based late fusion, which learn from data to exploit both geometric and dynamic information of Radar data. RadarNet achieves state-of-the-art results on two large-scale real-world datasets in the tasks of object detection and velocity estimation. We further show that exploiting Radar improves the perception capabilities of detecting faraway objects and understanding the motion of dynamic objects.

Keywords: Radar; Autonomous Driving; Object Detection.

1 Introduction

Self-driving vehicles (SDVs) have to perceive the world around them in order to interact with the environment in a safe manner. Perception systems typically detect the objects of interest and track them over time in order to estimate their motion. Despite many decades of research, perception systems have not achieved the level of reliability required to deploy self-driving vehicles at scale without safety drivers.

Recent 3D perception systems typically exploit cameras [36, 5, 42], LiDAR [45, 34, 17], or their combination [32, 20, 6] to achieve high-quality 3D object detection. While cameras capture rich appearance features, LiDAR provides direct and accurate 3D measurements. The sparsity of LiDAR measurements (e.g., at long range) and the sensor's sensitivity to weather (e.g., fog, rain and snow) remain open challenges. In addition to detecting and recognizing objects, estimating their velocities is also of vital importance. In some safety critical situations,

⋆ Equal contribution. Work done during RG's internship at Uber ATG.

for example a child running out of occlusion in front of the SDV, the SDV needs to estimate velocities from a single measurement cycle in order to avoid collision. This estimation is often inaccurate (or even impossible) when using LiDAR or cameras alone as they provide static information only. While for pedestrians we may infer the motion from its pose with large uncertainty, for rigid objects like vehicles we can not make reasonable predictions from their appearance alone.

An appealing solution is to use sensors that are robust to various weather conditions and can provide velocity estimations from a single measurement. This is the case of Radar, which uses the Doppler effect to compute the radial velocities of objects relative to the SDV. Radar brings its own challenges, as the data is very sparse (typically much more so than LiDAR), the measurements are ambiguous in terms of position and velocity, the readings lack tangential information and often contain false positives. As a result, previous methods either focus on the ADAS by fusing Radar with cameras [4, 30, 29, 8], where the performance requirements are relatively low; or fuse Radar data at the perception output level (e.g., tracks) [7, 10, 9], thus failing to fully exploit the complementary information of the sensors.

In this paper, we take a step forward in this direction and design a novel neural network architecture, dubbed *RadarNet*, which can exploit both LiDAR and Radar to provide accurate detections and velocity estimates for the actors in the scene. Towards this goal, we propose a multi-level fusion scheme that can fully exploit both geometric and dynamic information of Radar data. In particular, we first fuse Radar data with LiDAR point clouds via a novel *voxel-based early fusion* approach to leverage the Radar's long sensing range. Furthermore, after we get object detections, we fuse Radar data again via an *attention-based late fusion* approach to leverage the Radar's velocity readings. The proposed attention module captures the uncertainties in both detections and Radar measurements and plays an important role in transforming the 1D radial velocities from Radar to accurate 2D object velocity estimates.

We demonstrate the effectiveness of RadarNet on two large-scale driving datasets, where it surpasses the previous state-of-the-art in both 3D object detection and velocity estimation. We further show that exploiting Radar brings significant improvements in perceiving dynamic objects, improving both motion estimation and long range detection.

2 Related Work

Exploiting LiDAR for Perception: As a high-quality 3D sensor, LiDAR has been widely used for 3D object detection in self-driving. Previous methods mainly differ in two aspects: the detection architecture and the input representation. While single-stage detectors [47, 45, 17] have the advantages of simplicity and fast inference, two-stage methods [6, 15, 34] are often superior in producing precisely localized bounding boxes. Different representations of LiDAR point clouds have been proposed: 3D voxel grids [18], range view (RV) projections [19, 27, 6], bird's eye view (BEV) projections [47, 45, 44], and point sets [34, 32, 35]

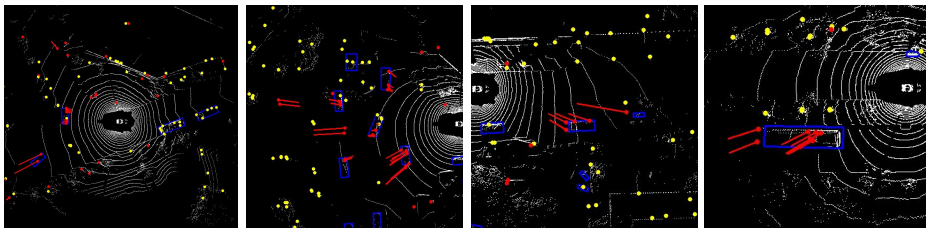

Fig. 1. LiDAR and Radar sensor data: We show LiDAR data in white, dynamic Radar returns (with radial velocity) in red, static Radar returns in yellow, and object labels in blue.

are amongst the most popular. While 3D voxel grids are slow and wasteful to process due to the size of the volume which is mainly sparse, range view projections are dense representations by nature. However, RV images suffer from the large variance in object size and shape due to the projection. BEV projections achieve a better trade-off between accuracy and speed. Voxel features represented with either simple statistics [45, 6] or learned representations [17] have been proposed. In this paper, we use a single-stage detector with BEV representation for its simplicity, effectiveness and efficiency.

Exploiting Radar for Perception: Radar has long been used in ADAS for adaptive cruise control and collision avoidance due to its cost and robustness to severe weather conditions. Recently, Radar has been exploited in many other applications, spanning across free space estimation [38, 25], object detection [4, 29, 8], object classification [13, 43, 31] and segmentation [38, 33, 25]. However, most of these methods treat Radar as another 3D sensor, ignoring its high-fidelity velocity information. In contrast, we exploit both Radar's geometric and dynamic information thanks to a novel specialized fusion mechanism for each type of information.

Sensor Fusion with Radar: In many self-driving perception systems, Radar data has been fused at the perception output level in the form of object tracks [7, 10, 9]. Kalman Filter [39] or IMM [2] trackers are popular approaches to digest Radar data, and the resulting tracks are then fused with object tracks from other sensors. However, sensor fusion is not exploited during the process of generating those object tracks. Recent works also look at fusion between Radar and cameras within the perception system. Different Radar representations are proposed to facilitate fusion: spectrogram images [22], sparse locations in image space [29], pseudo-image by projecting to image space [30, 4], BEV representation [28] and object detections [16]. However, these methods do not have high accuracy in terms of 3D perception. Instead, here we choose to fuse Radar with LiDAR and design a multi-level fusion mechanism that outperforms the state-of-the-art in self-driving.

Sensor Modality	Detection Range	Range Accuracy	Azimuth Resolution	Velocity Accuracy
LiDAR	100 m	2 cm	$0.1^\circ \sim 0.4^\circ$	-
Radar	250 m	10 cm near range 40 cm far range	$3.2^\circ \sim 12.3^\circ$ near range 1.6° far range	0.1 km/h

Table 1. Hardware comparison between LiDAR and Radar sensors.

3 Review of LiDAR and Radar Sensors

We first provide a review of LiDAR and Radar sensors and introduce our notation. We hope this short review can help readers better understand the intuitions behind our model designs, which will be described in the next section.

LiDAR (light detection and ranging) sensors can be divided into three main types: spinning LiDAR, solid state LiDAR, and flash LiDAR. In this paper we focus on the most common type: spinning LiDAR. This type of LiDAR emits and receives laser light pulses in 360° and exploits the time of flight (ToF) to calculate the distance to the obstacles. As a result, LiDAR data is generated as a continuous stream of point clouds. We denote each LiDAR point as a vector $P = (x, y, z, t)$, encoding the 3D position and the capture timestamp. In practice we often divide the LiDAR data into consecutive 360° sweeps for frame-wise point cloud processing. LiDAR is the preferred sensor for most self-driving vehicles due to its accurate 3D measurements. The main drawbacks are its sensitivity to dirt (which leads to poor performance in fog, rain and snow), cold (that causes exhaust plumes) as well as the lack of reflectivity of certain materials (such as windows and certain paints). Furthermore, its density decreases with range, making long range detection challenging.

Radar (radio detection and ranging) sensors work similarly as LiDAR, but transmit electromagnetic waves to sense the environment. The Radar outputs can be organized in three different levels: raw data in the form of time-frequency spectrograms, clusters from applying DBSCAN [12] or CFAR [37] on raw data, and tracks from performing object tracking on the clusters. From one representation to the next, the data sparsity and abstraction increases, while the noise in the data decreases. In this paper we focus on the mid-level data form, Radar clusters, for its good balance between information richness and noise. In the following we refer to these clusters as Radar targets. We denote each Radar target as a vector $Q = (\mathbf{q}, v_\parallel, m, t)$, where $\mathbf{q} = (x, y)$ is the 2D position in BEV, $v_\parallel$ is a scalar value representing the radial velocity, m is a binary value indicating whether the target is moving or not, and t is the capture timestamp. The main advantages of Radar are that it provides instantaneous velocity measurements and is robust to various weather conditions. However, its drawbacks are also significant. It has a low resolution and thus it is difficult to detect small objects. There are ambiguities (a modulo function) in range and velocity due to Radar aliasing, as well as false positive detections from clutter and multi-path returns.

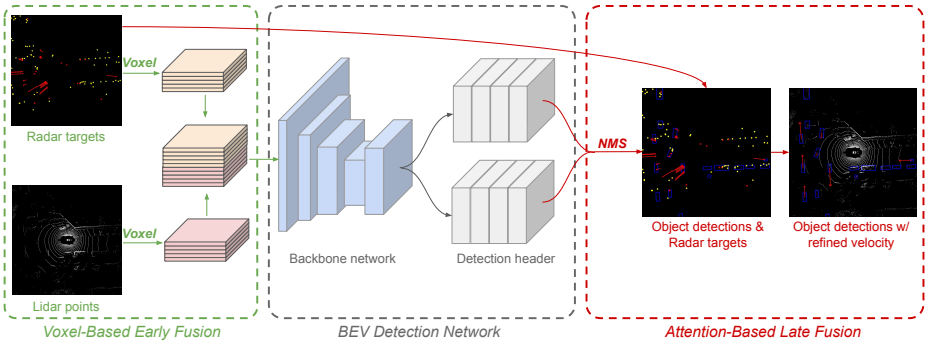

Fig. 2. RadarNet: Multi-level LiDAR and Radar fusion is performed for accurate 3D object detection and velocity estimation.

It is also worth noting that the objects' real-world velocities (2D vectors in BEV) are ambiguous given only the radial velocity. Therefore we need to additionally estimate the tangential velocity or the 2D velocity direction in order to properly utilize the radial velocity.

We compare LiDAR and Radar data both quantitatively and qualitatively. We visualize both sensors' data from the nuScenes dataset [3] in Fig. 1, and we compare their technical specifications in Table 1. Note that LiDAR outperforms Radar in both accuracy and resolution by over an order of magnitude. The accurate 3D surface measurements makes LiDAR the first choice for high-precision 3D object detection. Radar can provide complementary information in two aspects: more observations at long range and instantaneous velocity evidence from the Doppler effect. We thus argue that since these sensors are very complementary, their combination provides a superior solution for self-driving.

4 Exploiting LiDAR and Radar for Robust Perception

In this section we present our novel approach to 3D perception, involving 3D object detection and velocity estimation. We refer the reader to Fig. 2 for an illustration of the overall architecture of our approach. To fully exploit the complementary information of the two sensor modalities and thereby benefit both object detection and velocity estimation, we propose two sensor fusion mechanisms, namely *early fusion* and *late fusion*, that operate at different granularities. More specifically, while early fusion learns joint representations from both sensor observations, late fusion refines object velocities via an attention-based association and aggregation mechanism between object detections and Radar targets.

4.1 Exploiting Geometric Information via Early Fusion

LiDAR Voxel Representation: We take multiple sweeps of LiDAR point clouds (those within the past 0.5 seconds) as input so that the model has enough

information to infer the objects' motion while still being able to run in real-time. All point cloud sweeps are transformed to the ego-vehicle's centric coordinates at the current frame. Note that this is easy to do as sensors are calibrated and the vehicle pose is estimated by the localization system. Following FAF [26], we adopt a bird's eye view (BEV) representation and concatenate multiple height slices and sweeps together along the channel dimension. We use a weighted occupancy value as each voxel's feature representation. Specifically, for each voxel, if no point falls in it, the voxel's value is 0. If one or more points $\{(x_i, y_i, z_i), i = 1 \dots N\}$ fall into it, the voxel's value is defined as $\sum_i (1 - \frac{|x_i - a|}{\mathrm{d}x/2})(1 - \frac{|y_i - b|}{\mathrm{d}y/2})(1 - \frac{|z_i - c|}{\mathrm{d}z/2})$, where (a, b, c) is the voxel's center and $(\mathrm{d}x, \mathrm{d}y, \mathrm{d}z)$ is the voxel's size.

Radar Voxel Representation: Similar to how we accumulate multiple sweeps of LiDAR data, we also take multiple cycles of Radar data as input, in the same coordinate system as LiDAR. We keep only the (x, y) position of Radar targets and ignore the height position as it is often inaccurate (if it ever exists). As a result, each cycle of Radar data can be voxelized as one BEV image. We concatenate multiple cycles along the channel dimension and use a motion-aware occupancy value as the feature for each voxel. Specifically, for each BEV voxel, if no Radar target falls into it, the voxel's value is 0. If at least one moving Radar target (i.e., $m = 1$) falls into it, the voxel's value is 1. If all Radar targets falling into it are static, the voxel's value is -1.

Early Fusion: We use the same BEV voxel size for LiDAR and Radar data. Thus their voxel representations have the same size in BEV space. We perform early fusion by concatenating them together along the channel dimension.

4.2 Detection Network

We adopt a single-stage anchor-free BEV object detector with additional velocity estimation in the detection header.

Backbone Network: We adopt the same backbone network architecture as PnPNet [21]. The backbone network is composed of three initial convolution layers, three consecutive multi-scale inception blocks [40], and a feature pyramid network [23]. The three initial convolution layers down-sample the voxel input by 4 and output 64-D feature maps. The inception block consists of three branches, each with a down-sampling ratio of $1\times$, $2\times$ and $4\times$ implemented by stride of the first convolution. The number of convolution layers in each branch is 2, 4 and 6, and the number of feature channels in each branch is 32, 64 and 96. The feature pyramid network merges multi-scale feature maps from the inception block into one, with 256 channels for each layer. The final output of the backbone network is a 256-D feature map with a $4\times$ down-sampling ratio compared to the voxel input.

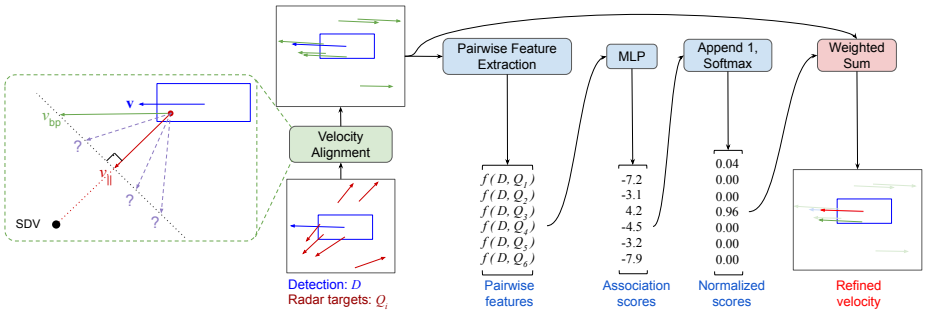

Fig. 3. Attention-based late fusion of object detection and Radar targets: In the figure we show an example of fusing Radar with one detection, while in practice this is applied to all detections in parallel. We first align the radial velocities of Radar targets with the detection's motion direction, then predict pairwise association scores for all detection-Radar pairs. The refined velocity is computed as a weighted sum of all Radar evidences as well as the original velocity estimate.

Detection Header: We apply a fully-convolutional detection header [24] for anchor-free dense detection, which consists of a classification branch and a regression branch, each with 4 convolution layers and 128 channels. The detection is parameterized as $D = (c, x, y, w, l, \theta, \mathbf{v})$, which represents the confidence score, the object's center position in BEV, its width, length and orientation, and its 2-D velocity $\mathbf{v} = (v_x, v_y)$ in BEV. The classification branch predicts the confidence score c, while the regression branch predicts all the other terms $(x - p_x, y - p_y, w, l, \cos(\theta), \sin(\theta), m, v_x, v_y)$, where p_x and p_y are the 2D coordinates of every voxel center and m is an additional term that indicates the probability of moving. During inference, we set the 2-D velocity to $(0, 0)$ if the predicted probability of moving is smaller than 50%.

4.3 Exploiting Dynamic Information via Late Fusion

While early fusion exploits the position and density information of Radar targets, late fusion is designed to explicitly exploit the Radar's radial velocity evidence. Due to the lack of tangential information, the actual object velocity vector is ambiguous given the radial velocity alone. To address this issue, we propose to use the velocity estimation in object detections to align the radial velocity, which is simply back-projecting the radial velocity to the motion direction of the detection. We refer the reader to Fig. 3 for an illustration. It is thus apparent that the radial velocity is more confident when the angle between the radial direction and motion direction is small, as when it is close to 90°, a very small variance in radial velocity will be exaggerated by back-projection.

Given a set of object detections and Radar targets, the key of fully exploiting Radar data lies in solving the following two tasks: (1) *association* of each Radar target with the correct object detection for velocity alignment; (2) *aggregation* to combine the velocity estimates from detection and associated Radar targets

robustly. Both tasks are non-trivial to solve. The association is not a one-to-one mapping as there are many objects without any associated Radar targets, and there are also objects with multiple Radar targets. False positives and noisy positions of Radar targets also make association difficult. For the aggregation problem, it is hard to estimate the uncertainty of the Radar velocity as it also depends on the associated detection.

In this paper, we propose an attention-based mechanism that learns from data to both associate and aggregate. This is illustrated in Fig. 3. Specifically, given pairwise features defined between each object detection and Radar target, we first compute pairwise association scores via a learnable matching function. We then aggregate each detection with all Radar targets according to the normalized association scores to get the refined velocity estimate. Note that late fusion is performed on *dynamic* Radar targets only.

Pairwise Detection-Radar Association: Given an object detection denoted as $D = (c, x, y, w, l, \theta, \mathbf{v})$ and a Radar target denoted as $Q = (\mathbf{q}, v_\parallel, m, t)$, we first define their pairwise feature as follows:

$$f(D, Q) = (f^{\text{det}}(D), f^{\text{det-radar}}(D, Q)) \tag{1}$$

$$f^{\text{det}}(D) = (w, l, \|\mathbf{v}\|, \frac{v_x}{\|\mathbf{v}\|}, \frac{v_y}{\|\mathbf{v}\|}, \cos(\gamma)) \tag{2}$$

$$f^{\text{det-radar}}(D, Q) = (\mathrm{d}x, \mathrm{d}y, \mathrm{d}t, v^{\text{bp}}) \tag{3}$$

$$v^{\text{bp}} = \min(50, \frac{v_\parallel}{\cos(\phi)}) \tag{4}$$

where $(\cdot, \cdot)$ indicates the concatenation operator, γ is the angle between D's motion direction and D's radial direction, ϕ is the angle between D's motion direction and Q's radial direction, v^{bp} is the back-projected radial velocity (capped by 50 m/s to avoid very large values), and $(\mathrm{d}x, \mathrm{d}y, \mathrm{d}t)$ are the offsets in BEV positions and timestamps of D and Q.

We then compute the pairwise association score by feeding the above feature to a learnable matching function:

$$s_{i,j} = \text{MLP}_{\text{match}}(f(D_i, Q_j)) \tag{5}$$

In our case the matching function is parameterized as a Multi-Layer Perceptron (MLP) with five layers with 32, 64, 64, 64 and 1 channels respectively.

Velocity Aggregation: We compute the association scores for all detections and Radar target pairs and refine the velocity estimate of each detection D_i by aggregating information from all Radar targets. Towards this goal, we first normalize the association scores of all Radar targets to sum to 1. We append an additional score of 1 before normalization to handle cases with no association.

$$\mathbf{s}_i^{\text{norm}} = \texttt{softmax}((1, s_{i,:})) \tag{6}$$

We then refine the velocity magnitude by summing all the candidates (the detection itself as well as all Radar targets) weighted by their normalized scores:

$$v'_i = \mathbf{s}_i^{\text{norm}} \cdot (\|\mathbf{v}_i\|, v_{i,:}^{\text{bp}})^\top \tag{7}$$

The 2D velocity estimate is then computed as the refined velocity magnitude:

$$\mathbf{v}' = v' \cdot (\frac{v_x}{\|\mathbf{v}\|}, \frac{v_y}{\|\mathbf{v}\|}) \tag{8}$$

where the detection index i is omitted for brevity.

4.4 Learning and Inference

We trained the proposed LiDAR and Radar fusion model with a multi-task loss defined as a weighted sum of the detection loss, velocity loss on the detection output, as well as the velocity loss on the late fusion output:

$$\mathcal{L} = (\mathcal{L}_{\text{cls}}^{\text{det}} + \alpha \cdot \mathcal{L}_{\text{reg}}^{\text{det}}) + \beta \cdot (\mathcal{L}_{\text{cls}}^{\text{velo}} + \mathcal{L}_{\text{reg}}^{\text{velo}}) + \delta \cdot \mathcal{L}_{\text{reg}}^{\text{velo\_attn}} \tag{9}$$

where $\mathcal{L}_{\text{cls}}^{\text{det}}$ is the cross-entropy loss on classification score c, $\mathcal{L}_{\text{reg}}^{\text{det}}$ is the smooth ℓ_1 loss summed over the position, size and orientation terms, $\mathcal{L}_{\text{cls}}^{\text{velo}}$ is the cross-entropy loss on moving probability m, $\mathcal{L}_{\text{reg}}^{\text{velo}}$ is the smooth ℓ_1 loss on $\mathbf{v}$, and $\mathcal{L}_{\text{reg}}^{\text{velo\_attn}}$ is the smooth ℓ_1 loss on $\mathbf{v}'$. α, β and δ are scalars that balance different tasks. Note that we do not require explicit supervision to learn object and Radar association, which is an advantage of the attention-based late fusion module where the association is implicitly learned.

We use the Adam optimizer [14] with batch normalization [11] after every convolution layer and layer normalization [1] after every fully-connected layer (except for the final output layer). For detection we use hard negative mining. $\mathcal{L}_{\text{reg}}^{\text{det}}$, $\mathcal{L}_{\text{cls}}^{\text{velo}}$ and $\mathcal{L}_{\text{reg}}^{\text{velo}}$ are computed on positive samples only, and $\mathcal{L}_{\text{reg}}^{\text{velo\_attn}}$ is computed on true positive detections only. We apply the same post-processing to generate final detections during training and testing phases, where the top 200 detections per class are kept and NMS is applied thereafter.

5 Experimental Evaluation

5.1 Datasets and Evaluation Metrics

nuScenes: We validate the proposed method on the nuScenes dataset [3]. This dataset contains sensor data from 1 LiDAR and 5 Radars, with object labels at 2Hz. Velocity labels are computed as finite difference between consecutive frames. Since we focus on dynamic objects, we evaluate on two challenging object classes: cars and motorcycles, as their velocities have high variance. We follow the official training/validation split with 700/150 logs each. We report the model performance on object detection and velocity estimation. Average Precision (AP) is used as the detection metric, which is defined on center distance in BEV

between the detection and the label. The final AP is averaged over four different distance thresholds (0.5m, 1m, 2m and 4m). Average Velocity Error (AVE) is used as the velocity metric, which is computed as the ℓ_2 velocity error averaged over all true positive detections (at 2m threshold). Cars are evaluated within 50m range, while motorcycles are evaluated within 40m range. Labels with 0 LiDAR and Radar points are ignored.

DenseRadar: One advantage of Radar over LiDAR is its longer sensing range. To showcase this, we further evaluate our model on a self-collected dataset, called *DenseRadar*, with vehicle labels within 100m range for 5002 snippets. Velocity labels are estimated by fitting a kinematic bicycle model to the trajectory, which produces smoother velocities compared with the finite difference procedure employed in nuScenes. We use similar metrics as nuScenes. For detection we compute AP at 0.7 IoU in BEV. For velocity we report Average Dynamic Velocity Error (ADVE) on *dynamic* objects only. We make a training/validation split with 4666/336 logs each.

5.2 Implementation Details

We train a two-class model on nuScenes with a shared backbone network and class-specific detection headers. Global data augmentation is used during training, with random translations from [-1, 1]m in the X and Y axes and [-0.2, 0.2]m in the Z axis, random scaling from [0.95, 1.05], random rotation from [-45°, 45°] along the Z axis, and random left-right and front-back flipping. We do not apply augmentation at test time. To alleviate the class imbalance, we duplicate training frames that contain motorcycles by 5 times. The model is trained for 25 epochs with a batch size of 32 frames on 8 GPUs. We use an input voxel size of 0.125m in the X and Y axes, and 0.2m in the Z axis. We use $\alpha = 1$ and $\beta = \delta = 0.1$. Hyper-parameter tuning is conducted on the train-detect/train-track split.

We train a single-class model on DenseRadar. Since the dataset is much larger, we do not apply data augmentation. We use an input voxel resolution of 0.2m in all three axes due to the extra computation due to the longer detection range. We use $\alpha = 1$ and $\beta = \delta = 0.5$. The model is trained for 1.5 epochs.

5.3 Comparison with the State-of-the-Art

We compare our LiDAR and Radar fusion model with other state-of-the-art perception models on nuScenes and show the evaluation results in Table 2. Specifically, we compare with the camera-based method MonoDIS [36], the LiDAR-based methods PointPillar [17], PointPillar+ [41], 3DSSD [46], CBGS [48], and the LiDAR and camera fusion method PointPainting [41]. RadarNet outperforms all methods significantly in both detection AP and velocity error. Compared with the second best on cars/motorcycles, our model shows an absolute gain of 2.2%/2.3% in detection AP and a relative reduction of 7%/21% in velocity error.

Method	Input	Cars			Motorcycles		
		AP↑	AP@2m↑	AVE↓	AP↑	AP@2m↑	AVE↓
MonoDIS [36]	I	47.8	64.9	-	28.1	37.7	-
PointPillar [17]	L	70.5	76.1	0.269	20.0	22.8	0.603
PointPillar+ [41]	L	76.7	80.5	0.209	35.0	38.6	0.371
PointPainting [41]	L+I	78.8	82.9	0.206	44.4	48.1	0.351
3DSSD [46]	L	81.2	85.8	0.188	36.0	39.9	0.356
CBGS [48]	L	82.3	85.9	0.230	50.6	52.4	0.339
RadarNet (Ours)	L+R	**84.5**	**87.9**	**0.175**	**52.9**	**55.6**	**0.269**

Table 2. Comparison with the state-of-the-art on nuScenes validation set.

5.4 Ablation Study

We conduct an ablation study on the nuScenes and DenseRadar datasets to validate the effectiveness of our two-level fusion scheme. To better verify the advantage of the proposed attention-based late fusion, we build a strong baseline with carefully designed heuristics. Recall that our attention-based late fusion consists of two steps: association and aggregation. As a counterpart, we build the baseline fusion method by replacing each step with heuristics. In particular, for each detection candidate, we first use a set of rules to determine the Radar targets associated with it. Given a set of associated Radar targets (if any), we then take the median of their aligned velocities (by back-projecting to the motion direction of the detection) as the estimate from Radar and average it with the initial velocity estimate of the detection. If there are no associated Radar targets, we keep the original detection velocity.

Below we define the set of rules we designed for determining the associated Radar targets. Given the features in Eq. 2 and Eq. 3, a Radar target is considered as associated if it meets all of the following conditions:

$$\sqrt{(\mathrm{d}x)^2 + (\mathrm{d}y)^2} < 3 \text{ m} \tag{10}$$

$$\gamma < 40^\circ \tag{11}$$

$$\|\mathbf{v}\| > 1 \text{ m/s} \tag{12}$$

$$v^{\mathrm{bp}} < 30 \text{ m/s} \tag{13}$$

We define these rules to filter out unreliable Radar targets, and the thresholds are chosen via cross-validation.

Evaluation on nuScenes: We show ablation results on nuScenes in Table 3. Note that our LiDAR only model already achieves state-of-the-art performance. Adding early fusion improves detection of motorcycles by 1.9% absolute AP, as the LiDAR observations are sparse and therefore Radar data serves as additional evidence. Early fusion does not affect the velocity performance much as only density information is exploited at present. When it comes to late fusion, our

Model	LiDAR	Radar		Cars		Motorcycles	
		Early	Late	AP@2m↑	AVE↓	AP@2m↑	AVE↓
LiDAR	✓	-	-	87.6	0.203	53.7	0.316
Early	✓	✓	-	+0.3	-2%	+1.9	-0%
Heuristic	✓	✓	heuristic	+0.3	-9%	+1.9	-4%
RadarNet	✓	✓	attention	**+0.3**	**-14%**	**+1.9**	**-15%**

Table 3. Ablation study on nuScenes validation set.

Model	LiDAR	Radar		Vehicles AP ↑			ADVE ↓
		Early	Late	0-40m	40-70m	70-100m	
LiDAR	✓	-	-	95.4	88.0	77.5	0.285
Early	✓	✓	-	+0.3	+0.5	+0.8	-3%
Heuristic	✓	✓	heuristic	+0.3	+0.5	+0.8	-6%
RadarNet	✓	✓	attention	**+0.3**	**+0.5**	**+0.8**	**-19%**

Table 4. Ablation study on DenseRadar validation set.

approach achieves over 14% velocity error reduction, significantly outperforming the heuristic baseline especially in motorcycles, where we typically have few Radar targets and therefore more noise.

Evaluation on DenseRadar: Ablation results on DenseRadar are depicted in Table 4. We show detection APs in near range (0-40m), mid range (40-70m) and long range (70-100m) respectively. Early fusion helps long-distance object detection, bringing 0.8% absolute gain in the 70-100m range detection AP. When late fusion is added, larger improvements are achieved than on nuScenes (from 14% to 19%). Two reasons may account for this: (1) DenseRadar uses higher-end Radar sensors that produce denser returns; (2) we evaluate in longer range (100m vs. 50m), which is more challenging and therefore there is more room for improvement. However, the heuristic baseline still gets lower than 10% gain, showing the advantage of the proposed attention-based mechanism which can learn from noisy data.

5.5 Fine-Grained Analysis

To better understand in which aspects the velocity estimation performance is improved by exploiting Radar we conduct fine-grained evaluation on the larger-scale DenseRadar dataset with respect to different subsets of object labels. In particular, we create different subsets of labels by varying the object distance to the ego vehicle, number of observed LiDAR points, angle γ between motion direction and radial direction, and the velocity magnitude.

We compare three model variants: LiDAR only, our model with heuristic late fusion and our model in Fig. 4. From the results we see that the heuristic

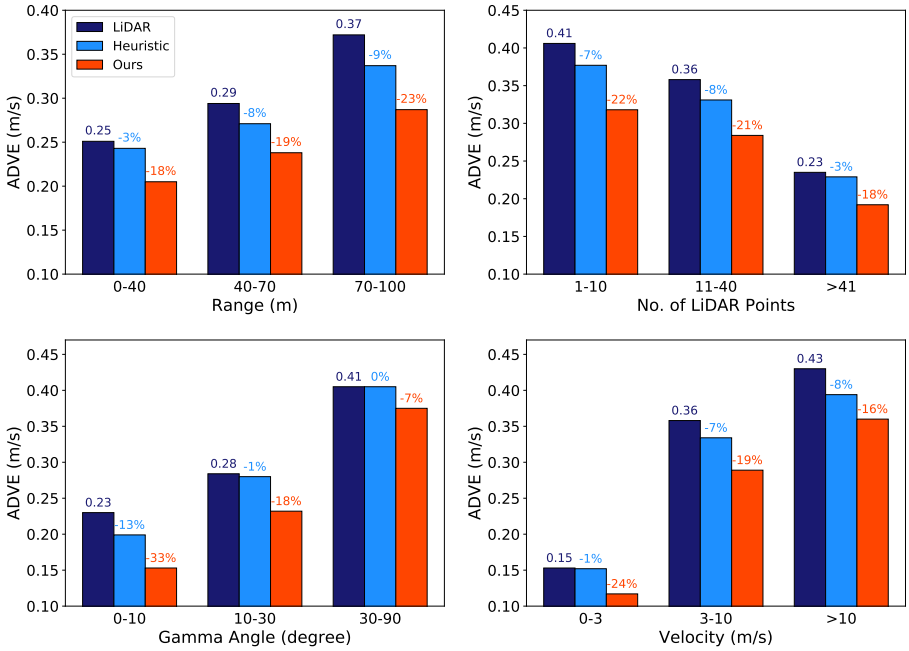

Fig. 4. Fine-grained evaluation of velocity estimation on DenseRadar validation set.

model brings negligible gains when $\gamma > 10°$ or $\|\mathbf{v}\| < 3$ m/s. This justifies the 40° and 1 m/s thresholds in our heuristics as these are cases where Radar data contain large uncertainty. In contrast, our attention-based model consistently and significantly outperforms the heuristic model under all conditions, showing its effectiveness in capturing sensor uncertainties and exploiting both sensors.

5.6 Qualitative Results

In Fig. 5 we show the learned detection and Radar associations. Results are shown in sequence for each object to illustrate the temporal change in the association. From the results we observe that: (1) the association is sparse in that only relevant Radar targets are associated; (2) the association is quite robust to noisy locations of the Radar targets; (3) the model captures the uncertainty of Radar targets very well. For example, when the radial direction is near tangential to the object's motion direction, the model tends to not associate any Radar targets as in such cases the Radar evidence is often very unreliable.

6 Conclusion

We have proposed a new method to exploit Radar in combination with LiDAR for robust perception of dynamic objects in self-driving. To exploit geometric

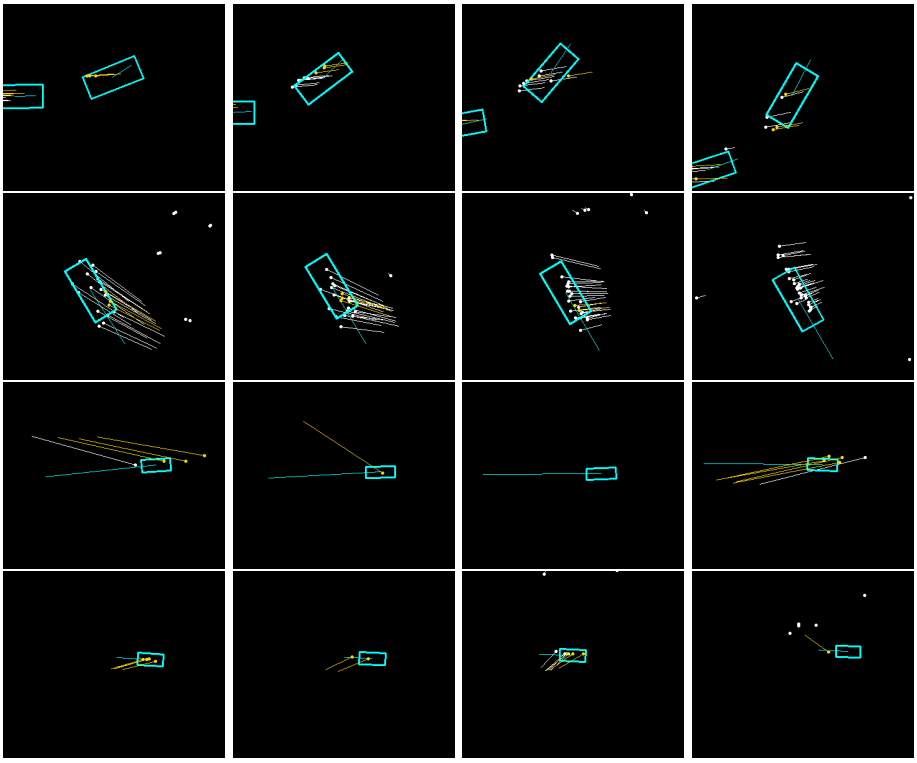

Fig. 5. Qualitative Results: Visualization of learned detections and Radar associations for cars (row 1 & 2) and motorcycles (row 3 & 4) on nuScenes validation set. Each row corresponds to the same object across time. We draw object detections in cyan, Radar targets within past 0.5s in white, and associated Radar targets with > 0.1 normalized score in yellow.

information from Radar, we use a voxel-based early fusion approach, which is shown to improve long-distance object detection due to Radar's longer sensing range. To exploit dynamic information, we propose an attention-based late fusion approach, which addresses the critical problem of associating Radar targets and objects without ground-truth association labels. By learning to associate and aggregate information, a significant performance boost in velocity estimation is observed under various conditions.

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)

2. Blom, H.A., Bar-Shalom, Y.: The interacting multiple model algorithm for systems with markovian switching coefficients. IEEE transactions on Automatic Control (1988)
3. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: CVPR (2020)
4. Chadwick, S., Maddetn, W., Newman, P.: Distant vehicle detection using radar and vision. In: ICRA (2019)
5. Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., Urtasun, R.: Monocular 3d object detection for autonomous driving. In: CVPR (2016)
6. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3d object detection network for autonomous driving. In: CVPR (2017)
7. Cho, H., Seo, Y.W., Kumar, B.V., Rajkumar, R.R.: A multi-sensor fusion system for moving object detection and tracking in urban driving environments. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). pp. 1836–1843. IEEE (2014)
8. Danzer, A., Griebel, T., Bach, M., Dietmayer, K.: 2d car detection in radar data with pointnets. In: ITSC (2019)
9. Göhring, D., Wang, M., Schnürmacher, M., Ganjineh, T.: Radar/lidar sensor fusion for car-following on highways. In: ICRA (2011)
10. Hajri, H., Rahal, M.C.: Real time lidar and radar high-level fusion for obstacle detection and tracking with evaluation on a ground truth. International Journal of Mechanical and Mechatronics Engineering (2018)
11. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML (2015)
12. Kellner, D., Klappstein, J., Dietmayer, K.: Grid-based dbscan for clustering extended objects in radar data. In: IEEE Intelligent Vehicles Symposium (2012)
13. Kim, S., Lee, S., Doo, S., Shim, B.: Moving target classification in automotive radar systems using convolutional recurrent neural networks. In: 26th European Signal Processing Conference (EUSIPCO) (2018)
14. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
15. Ku, J., Mozifian, M., Lee, J., Harakeh, A., Waslander, S.: Joint 3d proposal generation and object detection from view aggregation. In: IROS (2018)
16. Kuang, H., Liu, X., Zhang, J., Fang, Z.: Multi-modality cascaded fusion technology for autonomous driving. In: 4th International Conference on Robotics and Automation Sciences (ICRAS) (2020)
17. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: CVPR (2019)
18. Li, B.: 3d fully convolutional network for vehicle detection in point cloud. In: IROS (2017)
19. Li, B., Zhang, T., Xia, T.: Vehicle detection from 3d lidar using fully convolutional network. RSS (2016)
20. Liang, M., Yang, B., Wang, S., Urtasun, R.: Deep continuous fusion for multi-sensor 3d object detection. In: ECCV (2018)
21. Liang, M., Yang, B., Zeng, W., Chen, Y., Hu, R., Casas, S., Urtasun, R.: Object trajectory evolution for end-to-end perception and prediction. In: CVPR (2020)
22. Lim, T.Y., Ansari, A., Major, B., Fontijne, D., Hamilton, M., Gowaikar, R., Subramanian, S.: Radar and camera early fusion for vehicle detection in advanced driver assistance systems. In: Machine Learning for Autonomous Driving Workshop at the 33rd Conference on Neural Information Processing Systems (2019)

23. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR (2017)
24. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV (2017)
25. Lombacher, J., Laudt, K., Hahn, M., Dickmann, J., Wöhler, C.: Semantic radar grids. In: IEEE Intelligent Vehicles Symposium (IV) (2017)
26. Luo, W., Yang, B., Urtasun, R.: Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In: CVPR (2018)
27. Meyer, G.P., Laddha, A., Kee, E., Vallespi-Gonzalez, C., Wellington, C.K.: Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In: CVPR (2019)
28. Meyer, M., Kuschk, G.: Deep learning based 3d object detection for automotive radar and camera. In: 16th European Radar Conference (EuRAD) (2019)
29. Nabati, R., Qi, H.: Rrpn: Radar region proposal network for object detection in autonomous vehicles. In: ICIP (2019)
30. Nobis, F., Geisslinger, M., Weber, M., Betz, J., Lienkamp, M.: A deep learning-based radar and camera sensor fusion architecture for object detection. In: Sensor Data Fusion: Trends, Solutions, Applications (SDF) (2019)
31. Patel, K., Rambach, K., Visentin, T., Rusev, D., Pfeiffer, M., Yang, B.: Deep learning-based object classification on automotive radar spectra. In: RadarConf (2019)
32. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3d object detection from rgb-d data. In: CVPR (2018)
33. Schumann, O., Hahn, M., Dickmann, J., Wöhler, C.: Semantic segmentation on radar point clouds. In: FUSION (2018)
34. Shi, S., Wang, X., Li, H.: Pointrcnn: 3d object proposal generation and detection from point cloud. In: CVPR (2019)
35. Shi, W., Rajkumar, R.: Point-gnn: Graph neural network for 3d object detection in a point cloud. In: CVPR (2020)
36. Simonelli, A., Bulo, S.R., Porzi, L., López-Antequera, M., Kontschieder, P.: Disentangling monocular 3d object detection. In: ICCV (2019)
37. Skolnik, M.I.: Radar handbook second edition. McGrawHill (1990)
38. Sless, L., El Shlomo, B., Cohen, G., Oron, S.: Road scene understanding by occupancy grid learning from sparse radar clusters using semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 0–0 (2019)
39. Sun, S.L., Deng, Z.L.: Multi-sensor optimal information fusion kalman filter. Automatica **40**(6), 1017–1023 (2004)
40. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR (2015)
41. Vora, S., Lang, A.H., Helou, B., Beijbom, O.: Pointpainting: Sequential fusion for 3d object detection. In: CVPR (2020)
42. Weng, X., Kitani, K.: Monocular 3d object detection with pseudo-lidar point cloud. In: ICCVW (2019)
43. Wöhler, C., Schumann, O., Hahn, M., Dickmann, J.: Comparison of random forest and long short-term memory network performances in classification tasks using radar. In: Sensor Data Fusion: Trends, Solutions, Applications (SDF) (2017)
44. Yan, Y., Mao, Y., Li, B.: Second: Sparsely embedded convolutional detection. Sensors (2018)

45. Yang, B., Luo, W., Urtasun, R.: Pixor: Real-time 3d object detection from point clouds. In: CVPR (2018)
46. Yang, Z., Sun, Y., Liu, S., Jia, J.: 3dssd: Point-based 3d single stage object detector. In: CVPR (2020)
47. Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: CVPR (2018)
48. Zhu, B., Jiang, Z., Zhou, X., Li, Z., Yu, G.: Class-balanced grouping and sampling for point cloud 3d object detection. arXiv preprint arXiv:1908.09492 (2019)