UNIVERSITY OF CALIFORNIA SAN DIEGO

A New Way To Image Objects With Multiple Sensors

A thesis submitted in partial satisfaction of the
requirements for the degree
Master of Science

in

Electrical and Computer Engineering

by

Keshav Rungta

Committee in charge:

Professor Dinesh Bharadia, Chair
Professor Henrik Christensen
Professor Truong Nguyen

2021

The thesis of Keshav Rungta is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2021

DEDICATION

[TODO: Need to write something here]

EPIGRAPH

Two roads diverged in a wood, and I -

I took the one less traveled by,

And that has made all the difference

—Robert Frost

TABLE OF CONTENTS

Thesis Approval Page . . . iii

Dedication . . . iv

Epigraph . . . v

Table of Contents . . . vi

List of Figures . . . viii

List of Tables . . . x

Acknowledgements . . . xi

Vita and Publications . . . xii

Abstract of the Thesis . . . xiii

Chapter 1 Introduction . . . 1
1.1 Contributions . . . 4
1.2 Related Works . . . 7
1.2.1 Camera Only Methods . . . 8
1.2.2 LiDAR Only Methods . . . 9
1.2.3 Radar Only Methods . . . 9
1.2.4 Fusion based Approaches . . . 10
1.2.5 Datasets . . . 11
1.2.6 Clustering . . . 11

Chapter 2 Background and Challenges . . . 13
2.1 Radar Primer . . . 13
2.1.1 Challenges in Radar point clouds . . . 15
2.2 Data Representation and its Impacts . . . 16
2.3 Different Types of Fusion . . . 18

Chapter 3 Methodology . . . 20
3.1 Prerequisites of Robust Fusion . . . 20
3.1.1 Decoupled Feature Extraction . . . 20
3.1.2 BEV Feature Representation . . . 21
3.2 Hard Fusion . . . 22
3.2.1 System Architecture . . . 22
3.2.2 Clustering . . . 23
3.2.3 BEV Generation . . . 24
3.2.4 Bounding Box Detection . . . 26
3.3 Soft Fusion . . . 27
3.3.1 System Architecture . . . 27
3.3.2 Semantic Point Grid Encoding . . . 28
3.3.3 Bounding Box Detection . . . 31

3.4 Loss functions . . . 32

Chapter 4 Experimental Setup . . . 34
4.1 Hard Fusion . . . 34
4.1.1 Image segmentation network . . . 34
4.1.2 Training Details . . . 34
4.2 Soft Fusion . . . 35
4.2.1 Image segmentation network . . . 35
4.2.2 Training Details . . . 36
4.2.3 RADIATE . . . 37
4.2.4 Bad Weather augmentation . . . 38

Chapter 5 Evaluation and Results . . . 39
5.1 Metrics . . . 39
5.2 Dataset Details . . . 41
5.2.1 Astyx . . . 41
5.2.2 RADIATE . . . 41
5.3 Baselines . . . 42
5.3.1 Perspective view . . . 42
5.3.2 Multi-view aggregation Baseline . . . 42
5.4 Hard Fusion . . . 43
5.4.1 Quantitative results . . . 43
5.4.2 Performance on Visually Occluded objects . . . 44
5.4.3 Ablation studies . . . 46
5.4.4 Qualitative results . . . 47
5.5 Discussion . . . 47
5.6 Soft Fusion . . . 48
5.6.1 BEV bounding box prediction . . . 48
5.6.2 Performance on RADIATE dataset . . . 52

Chapter 6 Future Work and Next Steps . . . 53
6.1 Pointillism Dataset . . . 53
6.2 V-Loss . . . 53

Bibliography . . . 54

LIST OF FIGURES

Figure 1.1: Performance of radar-camera fusion architectures when camera input is affected with fog. AVOD-fusion [15] gets significantly deteriorated while our method continues to provide robust results even in fog. Filled blue boxes ■ are ground truth and red empty boxes □ are prediction results. 4

Figure 2.1: Point-cloud of a scene with one target car using multiple radars (red triangles and green dots are two different radars, blue points are from LiDAR). Multiple such target vehicles could be present in the scene. Figure shows noisy radar points generated due to clutter and multipath effects, and their independence across two radars. 14

Figure 3.1: Overview: An instance segmentation network identifies all object instances in the scene. A clustering block uses radar point cloud and instance masks to generate instance-wise BEV maps for each cluster. Cluster refinement reduces under-segmentation. Each BEV map is then passed through our detection network and predictions are generated for the entire scene. 23

Figure 3.2: Under-segmentation issue while clustering. Top image shows the projection of radar point clouds on different vehicles. Only the points belonging to the vehicle are kept for clarity. Due to lack of depth information while projecting the point cloud, clusters may contian points not belonging to the object. Bottom right figure shows the refined cluster after DBSCAN filtering. 25

Figure 3.3: Overview RadSegNet: Our approach utilizes encodings from our SPG module to detect objects. The encodings are generated from the semantic features from a semantic segmentation network along with radar point-based features and an occupancy grid. These encoded maps are concatenated and passed through the bounding box detection network. 28

Figure 3.4: Adding the semantic channel in SPG encoding helps in identifying points belonging to objects of interest. This figure shows how two cars are represented in the semantic map for the class cars and in the corresponding BEV occupancy grid. Similarly semantic maps can be represented for other classes. 31

Figure 4.1: Effect on segmentation outputs for class car with different adversarial weather. Left and right column shows the RGB and corresponding segmentation output. . 38

Figure 5.1: AP variation with distance for different IoU thresholds (in brackets). CF: CenterFusion. Our approach consistently performs better than baseline for all the distances. 45

Figure 5.2: The figure shows the final outputs of different system along with the radar point cloud. Red is the predicted box while Blue is the ground truth box. We compare our predictions (last column) with centerfusion [26] (2nd column). We also show the output without the DBSCAN refinement (3rd column). Our approach maintains provides a robust prediction performance for multiple scenarios including long-range. Best viewed digitally. FP: False Positive 46

Figure 5.3: Visualization of the output bounding box output of RadSegNet on challenging cases from Astyx Dataset. Black dots represent radar point clouds in BEV. Filled blue boxes ■ are ground truth and red empty boxes □ are prediction results. . . 49

Figure 5.4: Bounding box predicition results of RadSegNet in different weather conditions on RADIATE dataset. Blue box shows the ground truth vehicle and Red box shows our prediciton. 50

LIST OF TABLES

Table 5.1: mAP performance on test set of Astyx dataset. 43
Table 5.2: Detection rate for optimum score threshold 44
Table 5.3: mAP performance gain by using the DBSCAN based refinement. GT (Ground Truth) clustering is the performance when clustering is performed using ground truth labels. 44
Table 5.4: BEV Average Precision scores for different IoU thresholds in brackets. RadSegNet outperforms the state-of-the-art architectures consistently over all IoU thresholds. R: Radar; C: Camera; L: Lidar 48
Table 5.5: Ablation study experiment. We present the effect of each channel on the overall BEV AP performance of RadSegNet at different IoU thresholds. 49
Table 5.6: Effect on AP for Iou threshold 0.5 of different weather conditions on our architecture vs AVOD-fusion. The same input is provided to both the architectures in all the respective experiments. Percentage drop from clear weather performance in parenthesis. 50
Table 5.7: Average Precision Results for good weather and bad weather on RADIATE dataset for IoU threshold=0.5 52

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank my advisor, Prof. Dinesh Bharadia for his guidance, and support throughout my academic career at UC, San Diego. I would also like to thank Kshitiz Bansal, Siyuan Zhu for their help in our numerous collaborations over the last few years. [TODO: Need to finish writing this: properly elaborate on how they helped]

Chapter 3 in part is currently being prepared for a submission to IEEE Conference on Computer Vision and Pattern Recognition, 2022 (CVPR '22). It is a joint work between Kshitiz Bansal and Dr. Dinesh Bharadia

Chapter 4 contains works from 2 different works. The first section is a reprint of my contributions as it appears in the work Pointillism: Accurate 3D Bounding Box Estimation with Multi-Radars. This is co-authored by Kshitiz Bansal, Siyuan Zhu and Dinesh Bharadia. Additionally, the last section is currently being prepared for a submission to IEEE Conference on Computer Vision and Pattern Recognition, 2022 (CVPR '22). It is a joint work between Kshitiz Bansal and Dr. Dinesh Bharadia.

Chapter 5 in part is currently being prepared for a submission to IEEE Conference on Computer Vision and Pattern Recognition, 2022 (CVPR '22). It is a joint work between Kshitiz Bansal and Dr. Dinesh Bharadia

VITA

2020	B. S. in Electrical Engineering *cum laude*, University of California, San Diego
2020	Graduate Teaching Assistant, University of California, San Diego
2021	Interim Engineering Intern, Qualcomm
2021	M. S. in Electrical Engineering, University of California, San Diego

PUBLICATIONS

Kshitiz Bansal, Keshav Rungta, Siyuan Zhu, and Dinesh Bharadia, "Pointillism: accurate 3D bounding box estimation with multi-radars", *In Proceedings of the 18th Conference on Embedded Networked Sensor Systems (SenSys '20)*, Association for Computing Machinery, New York, NY, USA, 340–353.

ABSTRACT OF THE THESIS

A New Way To Image Objects With Multiple Sensors

by

Keshav Rungta

Master of Science in Electrical and Computer Engineering

University of California San Diego, 2021

Professor Dinesh Bharadia, Chair

Perception systems for autonomous driving have seen significant advancements in the last few years. Despite the advancements, these systems struggle to show robustness in extreme weather conditions primarily because primarily used sensors like LiDARs and Cameras see a decline in performance under these conditions. To solve this problem, camera and radar fusion systems provide a unique opportunity for all weather reliable high quality perception. Camera provides rich semantic information while radars can work in all weather. However, we find that the traditional state-of-the-art fusion methods are still heavily reliant on cameras, which essentially results in losing the all-weather reliability they set out to achieve. Contrary to these approaches, we propose two new sequential fusion methods, that treat Radars as the primary sensor in order to truly achieve all-weather reliability. We develop and validate our proposed system on the benchmark Astyx dataset and further verify these results on the RADIATE dataset. Our experimental results show that the proposed approaches achieve a 11% and 27% improvement, respectively, in terms of average precision score, and maintains a significantly better performance in bad weather conditions, compared to the state-of-the-art.

Chapter 1

Introduction

Over the last few years, the demand for robust autonomous driving systems has drastically increased. The main idea behind these systems is that they must be able to detect and track objects that are present in a scene and then plan a path to reach its destination. All of these tasks must be accomplished in real time and must be consistently accurate and precise in all scenarios and under all conditions.

The basis, of these autonomous system's is a perception subsystem. A perception system is responsible for identifying the objects that are present in a scene. Objects are identified by predicting a bounding box around them: estimate the spatial location, dimension and orientation of the object. Rapid research in self-driving perception systems has significantly improved their quality in the last few years. However, despite this quality improvement, no commercial autonomous driving system has been able to break the barrier of Level 3 autonomy. At level 3, systems have the capability to perform most detection and driving tasks, but human intervention can still be required. We see that these systems are bound to Level 3 because they are unable to perform reliably in the lethal edge cases; the same cases where these perception systems fail to detect objects in the scene. The primary reason, behind these systems failing, is their complete dependence on LiDARs, cameras, or

their fusion in order to detect objects.

Complete reliance on LiDARs and cameras has had some lethal consequences. As an example, LiDARs and cameras are unable to perform robustly in extreme weather conditions [4]. This limitation leads to the inability of perception systems to detect some of the objects present in a scene. Individually, cameras have the ability to provide rich textural information of the scene and they capture data with high spatial resolution; but, they are unable to provide any depth information to a perception system. The missing information makes it extremely hard to detect objects in a 3D Euclidean space. LiDARs, on the other hand, generate a high resolution, uniform point cloud, which can be used to, properly, image objects in a scene; but, they operate at a relatively low range and are extremely expensive [TODO: cite from cvpr].

These shortcomings, in current perception systems, have sparked a major interest in automotive radar-based sensing. In addition to robustness in bad weather conditions, radars also provide long range detections, and precise depth and speed estimations for objects in a scene. Lastly, commercial automotive radars are also extremely cheap. These advantages arise from the way that automotive radars function. They use millimeter waves (mmWaves) for sensing an environment. The larger wavelength, when compared to the nano meter level wavelength of LiDARs, allows them to unaffectedly pass through adverse weather conditions and sense reflections at a much longer range. This larger wavelength, however, has a huge downside; it makes radars quite challenging to work with. The all-weather reliability comes at the cost of low resolution data. The wavelength of radar sensors results in their point clouds being sparse and noisy [3, 5]. But, in order to reap the advantages offered by radars, past works try to use them in sensor fusion systems with visual sensors [15, 26, 27].

A camera radar fusion system, ideally, can combine the benefits of both visual and RF-based sensing. A visual sensor can provide 2D spatial resolution and rich textural information

about objects on the streets; while, radars are capable of providing all-weather reliance [3, 5] along with a longer range, precise depth, and speed estimates of the objects on road. Overall a radar-camera fusion is an inexpensive solution to the weather agnostic perception task of bounding box prediction. In this work I propose two different approaches at performing camera-radar fusion. The key difference between the two is the nature of fusion. One is a hard fusion, where we use one sensor modality to correct the data in the other sensor; while, the other method is a softer fusion, where the system learns, on its own, which parts of the inputted data are useful and which parts are not. The first approach is called *RadClustNet*, while the second is called *RadSegNet*. RadSegNet is a collaboration with Kshitiz Bansal, a PhD candidate at the University of California, San Diego.

In RadClustNet, location information and occupancy state of objects of interest are extracted directly from cameras and the sparsity of radar point clouds are addressed locally. RadClustNet's architecture can be divided into two major stages. In the first stage, a novel clustering approach generates instance wise clusters for radar point clouds, using image segmentation and spatial techniques on camera images. After this clustering stage, a deep learning model performs bounding box prediction on individual clusters locally.

In RadSegNet, however, the idea developed in RadClustNet is further refined. The entire system can once again be broken down into two stages: a data encoding stage and a bounding box prediction stage. In the encoding stage, camera semantics are coupled with radar point based features. The resulting information is then passed into a deep learning model to get the predicted boxes for the objects present in the scene currently being inferenced. By performing detections on the entire scene at the same time we saw some improvements over RadClustNet.

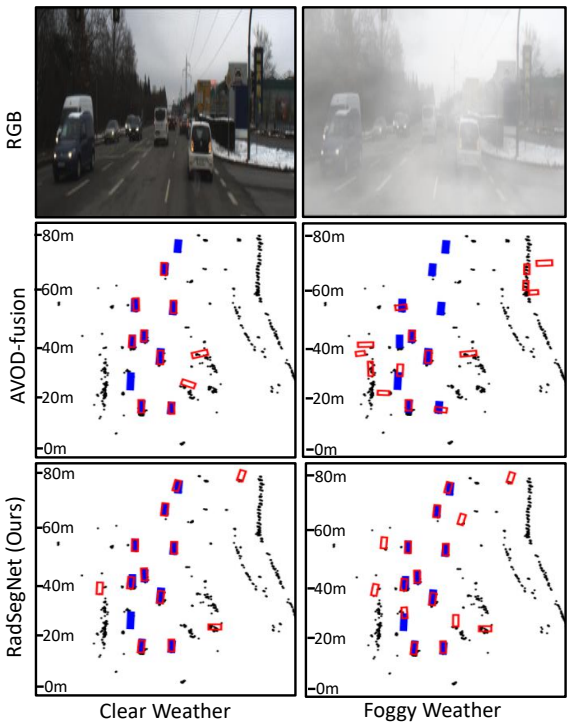

Figure 1.1: Performance of radar-camera fusion architectures when camera input is affected with fog. AVOD-fusion [15] gets significantly deteriorated while our method continues to provide robust results even in fog. Filled blue boxes ■ are ground truth and red empty boxes □ are prediction results.

1.1 Contributions

This work proposes two different approaches to camera-radar fusion. The first work, RadClustNet, utilises the features obtained from cameras in order to propose regions in the radar data that contains an object. The second work, RadSegNet, on the other hand, gives the system the freedom learn which parts of features are useful on its own, RadSegNet directly builds from the insights gained in RadClustNet.

Upon analysing current state of the art, radar-camera fusion, approaches, I noticed that the major setback of these systems is their heavy reliance on cameras. This reliance is the primary reason why their performance takes a severe hit, especially in cases where camera input is degraded: for example in bad weather conditions, as shown in **Figure 1.1**. It is also inherently non-trivial to

extrapolate depth or 3D information from a camera image; however, the data collected by a depth sensor, in the form of a 3D point cloud can be easily projected into either Bird's Eye View (where we look at a scene from the top down) or a camera perspective view. In this work, data from radars are made the center of focus, thereby reducing the inherent dependence on cameras. This shift enables the complete utilisation of the several benefits radar sensing has to offer. Additionally, by choosing automotive radars and industry-grade cameras, as the sensing modalities in the system's sensor suite,the proposed solutions are extremely low-cost and low maintenance because of the robustness of these modalities. On top of that, radars provide all-weather sensing capability, [35] which further adds to the two solution's reliability.

In order to get around the camera-centric issues, previously stated, data, in both of the proposed solutions is fused differently. In RadClustNet, a clustering stage encodes camera semantics into radar point clouds to generate instance wise inputs, while in RadSegNet camera semantics are coupled, as a whole to the radar point cloud. So, in RadSegNet, the entire scene is parsed at the same time while in RadClustNet, detection is done one instance at a time.

Clustering on radar point clouds is generally performed by spatial techniques like DBSCAN that rely on a distance-based threshold in the feature space of radar points [31]. These features include the coordinates, doppler, and intensity information of points. Standalone usage of DBSCAN lacks specific information about an object's locations and occupancy and is highly dependent on the pre-defined threshold. A proper clustering on radar point clouds would require an adaptive threshold, which is quite challenging in the case of sparse radar point clouds. Additionally, due to the radar signal's non-uniform reflection characteristics, these approaches by themselves could not differentiate objects of interest from the background when used without any priors. Contrary to this, in our approach, we use the masks generated by an instance segmentation network on camera images to achieve this clustering by introducing strong object priors.

A camera instance segmentation-based approach allows us to adapt according to different instance sizes. However, naively associating the projected points from radar with the mask they lie on would also lead to erroneous clustering. This is because of sensor mis-alignments and radar point cloud noise. To tackle this problem, the instance segmentation performed on camera images only identify coarse clusters which are further refined using DBSCAN, to achieve robust results.

Each cluster that was generated in the training step, is used to generate a separate BEV map. With the knowledge that each map contains only one object, the task of bounding box prediction using sparse radar point clouds becomes much more tractable. A detection module then predicts a BEV bounding box for each map, using the features encoded in this BEV map.

In RadSegNet, the second proposed solution, the focus shift from cameras to radars is achieved by refining the ideas first explored in RadClustNet. The rich information provided by cameras is appropriately fused with radar point data using a type of fusion called *sequential fusion*. In this approach, the system architects the sequential fusion by decoupling the simultaneous feature extraction from both the cameras and radars, used in state-of-art architectures. Instead, features are sequentially extracted first from the camera and then propagated to radar point clouds, thus retaining reliable radar inputs at all times along with a hint of semantic information from cameras. This shift enables RadSegNet to achieve all-weather perception benefits of radar sensing. Keeping radars as the primary modality ensures reliability in all situations including occlusions, long-range and bad weather. To the best of our knowledge, this is the first work that performs all-weather reliable sensor fusion of radar point clouds and camera images, at long ranges, while keeping radars as the primary sensing modality.

A natural next question is, how do we achieve the sequential fusion for radar-camera fusion. We create a novel semantic-point-grid (SPG) representation that encodes the rich semantic information from camera images into the radar point clouds by creating a combined representa-

tion from both modalities. Inspired from [36], we bank on the significant advancements made in the camera semantic segmentation literature and use the semantic features extracted from images instead of using raw RGB. The semantic information is combined with point-based features from each individual point in a radar point cloud.

The two proposed approaches to radar-camera fusion can be summarised as universal (works with any type of radar), reliable (maintains reliability in all-weather conditions); but only RadSegNet can be classified as complete (completely use all advantages of radar sensing). Both solutions can be readily integrated with other sensor-fusion approaches. Additionally, I also show that even in cases of bad weather, where the input from camera is unreliable, the two systems still maintain a reasonable performance. Figure 1.1 shows the robust performance of RadSegNet in foggy weather conditions compared to a baseline radar-camera fusion approach. For the cases of occlusion, where camera features do not add much value, the two methods still allow detections based on radar features. A comprehensive analysis on the improvements is provided by each radar point based feature in the average precision score. RadClustNet and RadSegNet are evaluated on a publicly available High resolution radar datasets, Astyx [25], while RadSegNet is also evaluated RADIATE [32].

1.2 Related Works

There has been a growing interest in building robust detection systems for the autonomous vehicles of the next generation. This is especially true when we start to encounter more edge cases like hidden or occluded objects, bad weather conditions, or unpredictable movements from agents in the scene. These problems are extremely challenging and there have been numerous attempts to solve them.

1.2.1 Camera Only Methods

Since the advent of deep learning, the one problem that has seen unprecedented success is 2D object detection using monocular camera images. With approaches like [TODO: RCNN, Faster RCNN, YOLO, SSD], among some others, seeing the most amount of success. However, 2D detections are not sufficient for autonomous vehicles. The perception task calls for the prediction of 3D boxes. This spurred a host of papers like [TODO: Mono3D, Deep Manta, 3D-RCNN], etc. While these papers showed some amount of success, predicting 3D boxes from 2D images is not a trivial problem. This is primarily because of the lacking depth information when using monocular cameras and the extreme difficulty in estimating geometric information from these images. As a result, all of these works have constraints associated with them. Some works attempted to get around these constraints by using stereo cameras to create a pseudo-LiDAR representation by first estimating a depth map. Then by back-projecting the resulting information to 3D, the data can be processed by using common LiDAR approaches. Papers like [TODO: Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object de- tection for autonomous driving, Pseudo- lidar++] follow this approach. However, the depth estimation portion of these approaches still result in massive loss in performance especially around the boundary of the objects in question. We took inspiration from some of these approaches in order to extract meaningful information from camera images in the form of semantic details. Being able to use the textural information of visible objects through the high angular resolution of the images themselves was key to our approach as it is useful complementary information to data presented by radars. However, these sensors are plagued by being susceptible to bad weather conditions and occlusions, which render distant objects near impossible to reliably detect.

1.2.2 LiDAR Only Methods

LiDARs have been the center of numerous approaches to solving the object detection problem. Having extremely dense pointclouds make it an extremely viable solution. The abundance of publicly available datasets have also aided in this proliferation. There have typically been two ways that LiDARs have been used. In the first way, works like [TODO: PointRCNN, STD, 3DSSD, PV-RCNN] perform detections directly on the point cloud. Most of these point based networks utilise [TODO: PointNet++] in order to extract features and perform detections. However, performing detections on pointclouds directly is not a trivial task as they are not very structured and uniform. As a result performing convolutions can be quite computationally demanding. On the other hand, works like [TODO: [Pointillism paper], VoxelNet, SA-SSD, PontPillars] alleviate this problem by first voxelising/ pixelising (into pillars) the 3D space. While this does instill some structure to the data, 3D convolutions are still quite computationally expensive. Some of the more recent approaches are taking the voxelising idea to a more extreme and creating a BEV representation of the data. These approaches have shown to be quite effective both in terms of Average Precision and computational efficiency. [TODO: Voxel-RCNN] is a prime example of this. However, none of these approaches tackle the biggest two problems associated with LiDARs: they are extremely expensive; they succumb to extreme weather conditions, rendering them ineffective.

1.2.3 Radar Only Methods

Radars on the other hand have their fair share of advantages and disadvantages. Being one of the very few sensor modalities that is robust to bad weather conditions they have seen a surge in interest in the Automotive field. There have been works that use Radars to solve the Classification problem [TODO: [pointillism paper]] while some other papers takes keys from LiDAR based approaches and utilise similar PointNet++ architectures to extract features or voxelise the space

to instill similar structures. However these approaches do not take into account the sparsity that is present in radar data due to the lower resolution of the sensor. As a result we see a much lower performance in radars. However, the modern sensors like those developed by by [TODO: Astyx] (now owned by Cruise), have a much higher resolution. This does solve the sparsity problem to some extent. But other works utilise a MIMO setup where they combine data from multiple radars to generate one dense pointcloud.

1.2.4 Fusion based Approaches

In an attempt to rectify the faults of any one given sensor with the strengths of another, there have been numerous attempts to fuse two or more sensors. Such ideas have proven especially useful in the cases of camera-radar fusion approaches. Some of these approaches essentially fuse the final predictions predicted by both sensor modalities [TODO: cite late fusion]. Some other approaches utilise ideas like projecting the data from one sensor modality onto another, like in [TODO: cite cvpr early fusion papers]. Both cases have proven to be ineffective because they are properly able to learn the faults of the individual sensor, as in [TODO: late fusion]; or they are trying to solve an intractable problem like depth estimation from monocular images and project camera data onto radars [TODO: early fusion]. There have been some more modern techniques that try to aggregate features from multiple views like in [TODO: cite mid fusion]. However such approaches have also shown to not be effective in cases of bad weather because the features extracted from the sensors are extremely correlated to each other. As a consequence, if one sensor fails then the entire architecture fails.

1.2.5 Datasets

In order for such approaches to be evaluated, there have been numerous different datasets proposed. Works like [TODO: KITTI, CityScapes, India driving, COCO], have been shown to be extremely effective for works pertaining to Cameras and LiDARs, where KITTI has been useful for object detection, tracking and planning related problems while the others have shown pertinence for semantic, image segmentation related tasks. NuScenes, a large scale dataset similar to KITTI, took the first major step in bringing radars into the limelight. However, the quality of the radar used was not upto the mark and it paved the way for datasets like [TODO: CRUW, Oxford, Zendar, RadarScenes]. However, each of these datasets lacked in something that was crucial to problems addressed in this works. RadarScenes was missing calibration information for its sensors in addition to labels; Oxford dataset also lacked labels. CRUW did not contain LiDAR data for us to cross check our work and neither did it contain doppler information for the radars, one of the key pieces of information provided by radars (which was also the case with Zendar). Lastly, [TODO: Astyx and RADIATE] are the two publicly available external datasets that was heavily utilised in this work. Astyx used a Hi-Res radar, while RADIATE used a mechanical radar which allowed the data for both to be quite rich.

1.2.6 Clustering

There have been numerous approaches to try and solve the clustering problem for radar or LiDAR based bounding box prediction systems. Density Based Spatial Clustering of Applications with Noise (DBSCAN) [**?**] is one of the most commonly used clustering approach. Some variations include Grid Based DBSCAN [6] and Adaptive DBSCAN [16]. The biggest problem with this approach is the over-segmentation of a scene into numerous tiny clusters due to lack of enough differentiating features between objects and background specially in case of radar point clouds.

Another stream of clustering algorithms include the Random Sample Consensus (RANSAC) [11] based methods which use various models of the objects (cars) in order to group the points. Some common models of vehicles include the L shape and the line shape [41] and points that don't obey these models are removed as outliers. Such models have also been used in segmentation tasks and also in probabilistic detection of cars to infer spatial uncertainty like in [38, 41] respectively. Due to a predefined shape matching, these approaches do not generalise well and lack robustness in case of sparse and noisy point clouds. More recent approaches combine one or more of the aforementioned models [19, 31, 34] but still fall prey to generalisability or robustness issues.

Chapter 2

Background and Challenges

2.1 Radar Primer

Automotive radars are active sensors that emit millimeter waves for perception. At the core, radars use the same time of flight (ToF) analysis of reflections to generate points, as in LiDARs. However, the key difference lies in the underlying techniques used to measure Time of Flight (ToF) and the Angle of Arrival (AoA) [2]. For radars, the FMCW (Frequency Modulated Continous Wave) technique has become a standard in the automotive radar market for point cloud generation and velocity estimation [**?**, 1]. Additionally, multiple transmit and receive antennas are used for the angle of arrival estimation using beamforming. The reader should refer to [1] for a detailed description of the entire point cloud generation. The resolution in range and angle estimation in FMCW depends on the bandwidth of operation and the number of available antennas, respectively. Due to the millimeter band of transmission, radars offer the following unique advantages:

(a) Longer detection range: This is because radio waves have a lower absorption rate, when reflecting from objects, in comparison to the rates of light waves. This allows radio waves to travel and reflect from longer distances [3].

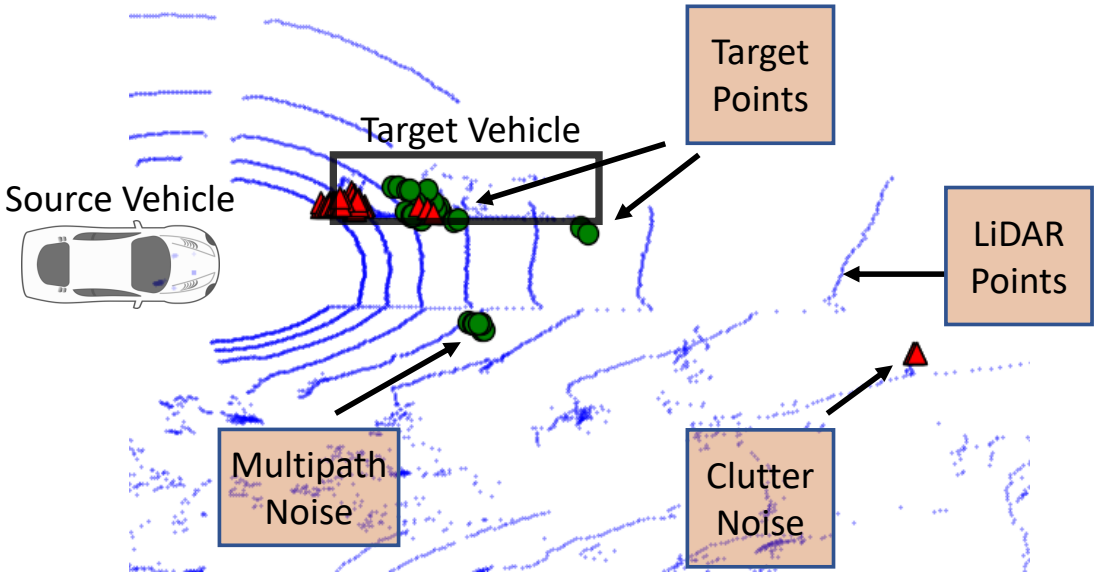

Figure 2.1: Point-cloud of a scene with one target car using multiple radars (red triangles and green dots are two different radars, blue points are from LiDAR). Multiple such target vehicles could be present in the scene. Figure shows noisy radar points generated due to clutter and multipath effects, and their independence across two radars.

(b) See-through occluding vehicles: This is because their signals bounce off the ground allowing them to sense vehicles which are completely occluded [14, 28].

(c) All weather sensor: This is because of the larger wavelength of millimeter waves. The wave are able to pass, unaffected, through adverse conditions like fog, snow and dust, which allows radars to receive reflections from objects in these extreme conditions.

The current market trend is to achieve higher angular resolution by using several co-located antennas and improve bounding box estimation performance. However, radar point clouds are affected by more fundamental challenges that inhibit the bounding box estimation performance on radar data. In the next section, these challenges are discussed.

2.1.1 Challenges in Radar point clouds

Millimeter wave sensing has enabled high resolution radars for automotive sensing but it has its own perils. These challenges are the primary reason why radars are not more ubiquitous in sensor suites. There are three main challenges faced in mmWave sensing [5]:

(a) Specular reflections: For an incident electromagnetic wave on a surface, the size of its wavelength compared to the roughness of the object's surface determines the degree of scattering of the wave. mmWaves undergo a negligible scattering effect, resulting in a specular reflection (angle of incidence = angle of departure) from the surfaces. Consequently, for a small aperture radar, a lot of reflected signal does not make its way back to the sensor, causing blindness of the objects. This blindness is even independent of the resolution capabilities of the sensor.

(b) Radar clutter and noise: Radar detections are commonly known to be polluted by signals from clutter, noise, and multipath effects. Radar clutter is defined as the unwanted echos from the ground or other objects like insects that can be confused with the objects under consideration. In a congested environment like cities, a signal emitted by a radar sensor could suffer multiple reflections before coming back to the sensor. The result is the formation of ghost objects, which are reflections of actual objects in some reflector formed because of multipath. **Figure 2.1** shows the different types of noises experienced by radar sensing.

(c) Sparsity: Outdoor scene point clouds are inherently sparse due to the empty volume between objects, which are at a substantial distance from each other. Additionally, due to different interaction properties of mmWaves with different objects (non-uniform interactions), this effect is compounded in the case of mmWave radars. The result is a sparse and non-uniform point cloud.

All of these challenges essentially means that the resulting point cloud is very sparse and noisy. Additionally, in radar point clouds, a point cluster originating from a wall has a similar spatial spread as that of a cluster originating from a car due to this sparsity. On the other hand,

light signals from LiDARs have very small wavelengths (≈1000 nm), at which most surfaces act as Lambertian, allowing the signals to go through diffused scattering. This leads to a very high resolution. Thanks to this resolution, LiDARs provide a uniformity in the generated point cloud. An average automotive radar generates about a thousand points from a scene. In contrast, a typical 16 channel Lidar generates around 35 thousand points for the same scene. This effect makes it challenging to learn any shape based features directly from radar point clouds to differentiate objects of interest (Cars, pedestrians, etc) from background objects. Figure 1.1 shows the sparsity and non-uniformity present in radar point clouds.

2.2 Data Representation and its Impacts

There have been numerous works in the past, like the one by Wang et. al. [37], that demonstrate the importance of data representation. Huge gains have been seen by simply changing the representation of the same data. In this work, we are dealing with 2 sensors: Cameras can be represented in the Perspective view, while radars can be represented in a Point Cloud or a Birds Eye View (BEV) image. I first present all possible ways of representing them and then explain which view we will be using for both approaches in this work.

Perspective View The perspective view is basically the data that is represented in the camera's image plane; when we look at a camera image, it is in perspective view. It is the ideal way to quickly grasp the composition of a scene, but it has a lot of cons associated to it. The first and most obvious con is that, if an object is occluded, there is no way to gather its data [37]. Additionally, perspective view also injects a scale ambiguity into its data: it is hard to distinguish between a large object, at a greater distance, from a smaller object, closer to the sensor. This ambiguity is the main reason why local computations like 2D convolutions cause different objects, at these different depths, to be

correlated with the same kernel, making the task of object detection significantly harder.

3D Point Cloud A Point Cloud, as the name suggests, is a representation where each of the reflections received by the depth sensor is stored as a point in 3D space. It is an excellent way to store the spatial information of different points in a scene and also correlate additional contextual information related to that point. In the case of radars, we see features like doppler, which tells us about the speed of the object relative to the sensor, or like intensity of reflection. However, the major downside associated with this representation is that there is no inherent order associated to the data: two points stored one after the other could belong to two completely unrelated objects at very different locations in space. One approach to rectify this problem is to perform a voxelisation, but because radars are so sparse, it is inefficient to do so. All of these together, make it challenging to apply convolutions. Additionally, 3D convolutions are expensive, which makes it less than ideal to perform object detection directly on Point Clouds [18,30] [TODO: figure out the citations (second)].

Bird's Eye View This last view is what we are focusing on in this work. Bird's Eye View (BEV) essentially is a view in which we look at the same scene from the top down. For sensors like Radars and LiDARs, we simply collapse the height information to project the entire point cloud onto one x-z plane which allows us to perfectly look at the depth and the horizontal coordinate of the point. Pixelising this space makes it very easy to apply convolutions. BEV has a clear advantage in the problems posed by perspective view. This is because looking at a scene from the top down it makes it much easier to separate objects from one another, allowing us to properly identify occluded objects. Additionally, the scale ambiguity due to depth is no longer valid, because the scale and depth information is both encoded in the generation of the BEV image. However, as previously mentioned in [TODO: ref early fusion], it is not trivial to project a camera image onto BEV. In

this work we are able to get around this problem by instead utilising the semantic features offered by cameras. [TODO: ref section talking about semantic segmentations].

2.3 Different Types of Fusion

The disadvantages associated with radars is primarily why we look at fusion based systems. A vision sensor like camera is perfect to supplement the disadvantages, mentioned in the previous section. A fusion between camera and radar modalities, historically, is performed at a late, early or middle stage [10]. Past works have shown that middle fusion has had the most amount of success, but just for completeness, lets take a look at all three types. This will allow me to establish where the two proposed solutions fit into the conversation.

Late Fusion: Late fusion approaches combine individual detections from multiple modalities, but can not learn any joint representation or perform joint feature extraction [10]. This fusion has proved to be the most ineffective because the flaws of the individual sensor still persists in the system and the system does not learn how to correct them using the advantages provided by the other sensor. For our context, this implies that these systems are ineffective in bad weather conditions.

Early Fusion: Early fusion approaches in camera-radar fusion works in two ways. Some project radar points to a camera perspective view [27] but operating in perspective view makes it harder to distinguish between objects at different depths and scales: smaller objects closer to the sensor are effectively correlated the same as large objects far away. Others approaches perform an inverse mapping of camera image to radar's Bird's eye View (BEV) but an inverse mapping of camera to BEV plane is ill-defined problem due to the lack of depth information in camera images [20]. Once again we see that both approaches are very dependent on cameras leading to similar issues as in the late fusion approaches.

Mid Fusion: Finally, the state-of-the-art performance is provided by the middle fusion or feature-level fusion approaches. These approaches either use multi-view feature aggregation from radar and image [15] or augment image features by associating radar points to the detected objects in camera perspective view [26]. They provide state-of-the-art results in camera radar fusion, but we show that these fusion systems fail to realise the complete potential of radar-camera fusion based sensing specially all-weather reliability.

Chapter 3

Methodology

3.1 Prerequisites of Robust Fusion

Before we go into the details pertaining to the two approaches proposed in this work, we should understand the requirements of a successful perception system using radar-camera fusion. We identify two main requirements to achieve a reliable and high quality radar-camera fusion. These are described below.

3.1.1 Decoupled Feature Extraction

To prevent a system from falling prey to adverse weather conditions, radar feature extraction should not rely on cameras. By separating this feature extraction, the deficiencies of the camera, in these extreme weather conditions are not propagated to the radar. If radar features are correlated to camera images, they are highly dependent on the quality of camera images: if camera input is degraded, we also see huge negative impacts on the radar feature extraction, as demonstrated in [TODO: reference results].

3.1.2 BEV Feature Representation

BEV representation is able to clearly separate objects at different depths, offering a clear advantage in cases of partially and completely occluded objects [37]. BEV becomes a necessity for radars as they even get signals from occluded objects due to the radio waves bouncing off ground (section **??**). Instead if we were to project points belonging to these occluded objects, onto a perspective camera view, we would see them overlapping with points belonging to the object occluding the view in a camera image. This overlap causes inaccuracies in detection. Instead of using BEV, if we were to use 3D point clouds, we might also see decent detections. But as mentioned in **Section 2.2**, its inefficient memory and compute requirements make it less than ideal for real time inference.

In this work, both these requirements are fulfilled. The two proposed solutions in this work cannot be classified as any of the three approaches mentioned in **Section 2.3**. The first approach, RadClustNet, is an attempt at an implementation on a 4th type of fusion strategy called *Sequential Fusion*, while, RadSegNet, the second approach, refines this implementation by sequentially fusing semantic information from cameras with the radar point clouds and projecting it onto a BEV map. By using sequential fusion, our solutions decouple the simultaneous feature extraction of the two modalities, as seen in mid fusion. We distill the rich scene semantic information from cameras and then forward this information to radars, which assists them in detecting objects. We are able to avoid the projection errors seen in approaches that use a mid-fusion strategy (**Section 2.3**) because we use semantic information instead of using the camera images directly. By using semantic information instead of camera images, the defects of cameras are not propagated to the feature extraction of radars. This is the key to performing detections in extreme weather conditions. In the first approach, we use camera semantics to help cluster object instances while in the second, this idea is refined into a *Semantic Point Grid* (SPG) encoding that fuses semantics with radar

based features. Both these approaches make use of all the advantages present in radar sensing while limiting the dependence on camera. **Figures 3.1, 3.3** shows the overview of our entire architecture for both approaches.

3.2 Hard Fusion

3.2.1 System Architecture

In order to perform reliable bounding box predictions, our first solution needs to address the specific challenges of non-uniformity and sparsity of radar point clouds by utilising the BEV representation and decoupling feature extraction between the two sensors. State-of-the-art approaches for camera-radar fusion, instead, perform simultaneous feature extraction from both modalities (**Section 1.2**). Simultaneous feature extraction is done by first extracting features individually from camera and radar in perspective and BEV respectively, and then fusing the extracted features to obtain bounding box detections. These approaches have shown an improvement in the overall detection performance as they are not limited by the perspective view. Despite this improvement, these methods struggle in cases of bad weather because the features learned from one sensor are extremely co-dependent on the other sensor. As a consequence, when one sensor sees a degradation in its performance, the entire architecture fails. These approaches limit the capability of radars in terms of all-weather and long distance. In this approach, we tackle these challenges by dividing our solution into 2 stages of: Radar Point Cloud Clustering using instance segmentation followed by Bounding box prediction in BEV. In this section we will go into details of each one of these. **Figure 3.1** shows the overview of our entire architecture.

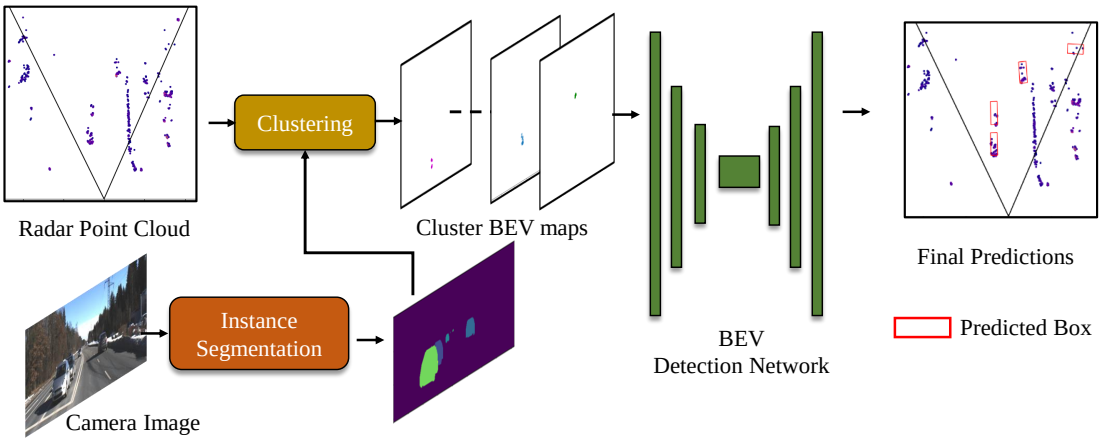

Figure 3.1: Overview: An instance segmentation network identifies all object instances in the scene. A clustering block uses radar point cloud and instance masks to generate instance-wise BEV maps for each cluster. Cluster refinement reduces under-segmentation. Each BEV map is then passed through our detection network and predictions are generated for the entire scene.

3.2.2 Clustering

The sparsity and noise in radar point clouds make identifying objects of interest in a scene a challenging task. This solution addresses this problem by using image segmentation cues from the camera image and propagating that information to radar point clouds. By using semantic information we *cluster* the radar points into points belonging to individual object instances. Clustering is done in two steps – obtaining coarse object-level clusters using image segmentation and a spatial correction step to refine the obtained clusters.

Clustering with Image segmentation Network

The idea behind this clustering step is to group all the radar points that belong to an instance of an object like vehicle and pedestrian and discard the points belonging to the background. To obtain such clusters, we need information about all the objects present in the scene. This information is extracted from a camera image by making use of a robust image segmentation model, Mask R-CNN [12]: masks are generated corresponding to each individual instance of the numerous

objects present in a scene (**Figure 3.1**). This is done using a pre-trained network (trained on COCO [23] dataset). After obtaining these masks, the radar points are projected onto the corresponding image plane (mask plane). An association step correlates the instance mask to the radar points in order to obtain a cluster of points for each object. Note that this process does not associate the same radar point to multiple objects as the masks given by instance segmentation network are mutually exclusive, as shown in **Figure 3.1**.

Spatial and doppler Correction Point cloud clusters generated as a result of instance mask can contain some additional points due to errors in projection. This error emerges as we collapse the depth information while projecting the point cloud onto the image plane. As a result, a cluster may contain points that do not belong to the actual location of the object as shown in **Figure 3.2** causing under-segmentation. My insight here is that we can leverage the spatial relationship between points of the cluster to identify and remove cases of under-segmentation. The idea is that points belonging to the actual object would be close in spatial location as well as doppler values. Specifically, we can use DBSCAN clustering [**?**] to further divide an under-segmented cluster into micro-clusters. Out of these new micro-clusters, we can pick the one with the maximum number of points as the final cluster corresponding to that object instance. By leveraging the spatial and doppler characteristics of radar points, We see a significant improvement in the bounding box prediction performance of the overall system.

3.2.3 BEV Generation

After obtaining individual clusters from radar point clouds the next step is to predict a bounding box for each of these clusters. Performing the clustering step provides us with the necessary priors required to perform bounding box prediction on radar point clouds.

Each clusters obtained from the clustering step consists of points from only a single object.

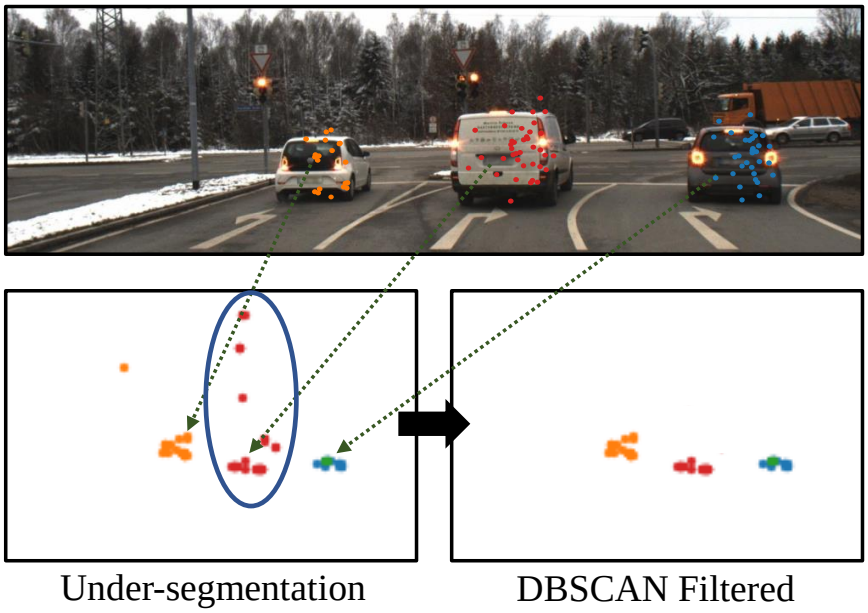

Figure 3.2: Under-segmentation issue while clustering. Top image shows the projection of radar point clouds on different vehicles. Only the points belonging to the vehicle are kept for clarity. Due to lack of depth information while projecting the point cloud, clusters may contian points not belonging to the object. Bottom right figure shows the refined cluster after DBSCAN filtering.

While, a simple solution for bounding box prediction is to pass the radar point cloud along with the clustering information through a PointNet [30] encoder. But, this would be structurally inefficient due to the non-uniformity of radar point cloud data. This effect is only made worse by the un-ordered format of point cloud data [30], leading to computational inefficiencies. In this solution, BEV Maps are used, that, by design, preserve the spatial relationships between the points in the clusters [18]. More details can be found in **Section 2.2**. Each of the instance-wise clusters is then projected onto a 2D plane by collapsing the height dimension. The projected cluster point cloud is converted into a grid map by pixelising the 2D space. This generates an occupancy grid, where each pixel is an indicator function that gets a value of 1 if it contains a point otherwise it is represented as 0. This BEV occupancy grid preserves the spatial relationships between the different points of an un-ordered point cloud and stores radar data in a more structured and CNN-friendly format [18].

This projection is also used to store doppler and intensity values. So, for each pixel, in the BEV map, the feature set includes an occupancy grid, doppler values and RCS values:

$$s := (\mathcal{I}((u, v)), d, r) \tag{3.1}$$

Here, $\mathcal{I}$ represents the occupancy grid and can be formulated as an indicator function defined as $\mathcal{I}((u, v))$ which indicates if there exists any point $p \in c_i$ that lies in the (u, v) pixel of the BEV map. c_i is the instance cluster for which the BEV map is being generated. d and r represents the average doppler and intensity value of all points, $p \in c_i$, projected to (u, v) pixel of the BEV map. They help identify objects based on their speeds and reflection characteristics.

3.2.4 Bounding Box Detection

Instance wise training [TODO: discuss with prof bharadia on how to proceed] While converting the radar point cloud to BEV, one could simply combine all the instance-wise BEV maps into a single BEV map. However, my original intuition was that is a sub-optimal solution as it would make the detection task harder by introducing ambiguity in the number of objects that are present in the scene. A solution could be to somehow encode the information about number of objects in the scene while detection, but it would not guarantee a one-to-one mapping between the predicted boxes and all the objects present in the scene. For example, multiple boxes could be predicted for the same object which would make the detection performance worse. Hence, training is done on individual BEV maps generated for each cluster. With this approach, the task of detection becomes much more tractable. In addition, the chances of false positives being generated significantly decreases as there is a one to one mapping between objects and BEV maps. Because each input into the system is guaranteed to only contain one object at a time, this approach gets rid of the NMS (non-maximal-separation) step [33], at the end of the detection. In order to predict a bounding box, we

can simply choose the box with the highest confidence score.

Feature Extraction and detection

Each of the BEV maps generated from clusters are passed into a deep neural network for feature extraction and bounding box prediction. For the feature extraction stage a deep U-Net [21] type encoder-decoder network is used. This specific implementation consists of 4 stages of down-sampling layers with 3 convolutional layers at each stage to extract features of different sizes during the encoding stage and then combine all the intermediate features during the up-sampling stage by using skip connections to generate the final set of features. The input into this stage is characterized as a tensor of shape $(W, H, C = 3)$, where the channels are given by the binary BEV Image, doppler and RCS values for each occupied pixel. I use an anchor box-based detection [24] architecture to generate predictions using a classification head which is followed by a regression head. The classification head predicts a confidence score for the output boxes and a regression head learns to refine their dimensions. As mentioned in the previous subsection, the final prediction that is inferred is the box that contains the highest confidence score. But, in order to train the network, we compute a loss value for each predicted box that meets a certain threshold of overlap over the ground truth bounding box.

3.3 Soft Fusion

3.3.1 System Architecture

The idea first developed in **Section 3.2** is further refined in this second approach which is a collaboration between Kshitiz Bansal and I. The reason behind this second approach is because, the first proposed approach still contained limited dependence on cameras. Radars, in addition to extreme weather robustness and long range detects also provide occlusion free sensing. The objective

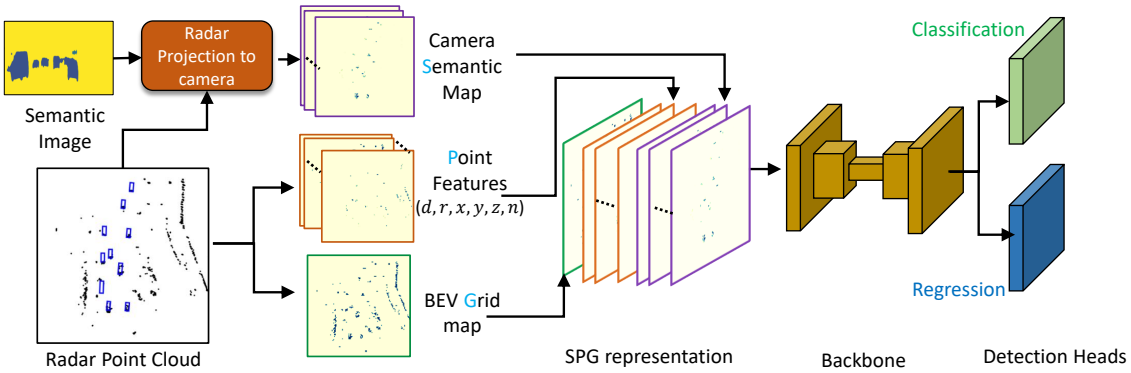

Figure 3.3: Overview RadSegNet: Our approach utilizes encodings from our SPG module to detect objects. The encodings are generated from the semantic features from a semantic segmentation network along with radar point-based features and an occupancy grid. These encoded maps are concatenated and passed through the bounding box detection network.

of sensor fusion approaches and this work in particular is to maximise use of the unique information provided by each of the two sensors. This approach aims to achieve this goal by once again using sequential fusion in BEV, but instead of clustering object instances, we generate one BEV map for the entire scene. By doing so, any projection error caused by occlusions in camera images are not propagated onto radar features. In order to deal with the sub-optimality of the problem discussed in the previous approach, we provide our system with additional features in order to assist the network gauging the number of vehicles that might be present in the scene. We generate these features by passing sensor data through our *Semantic Point Grid* (SPG) module in order to encode sequentially fused features.

3.3.2 Semantic Point Grid Encoding

This module takes the grass root idea introduced in the first approach and expands on it. The SPG encoding based input has 3 different components to it: a radar BEV based occupancy grid map, camera based semantic information and radar points based object information.

BEV Occupancy grid

In order to generate the radar based BEV occupancy grid, similar steps are used as demonstrated in **Section 3.2.3**. However the key difference is that instead of performing this instance wise, the entire radar point cloud is projected onto a 2D grid by collapsing the height dimension. Each pixel once again behaving as a indicator variable and stores 1 when the pixel is occupied.

Camera semantic features

The high resolution in camera images can be used to understand a scene and its objects. Recent advances in camera image segmentation have established their dominance in understanding scene semantics. Inspired from [36] we use a robust pre-trained semantic segmentation network to obtain semantic masks from camera images of the objects present in the scene.

Adding semantics to SPG To associate camera based semantics to radar points, each point in the radar point cloud is, first, projected onto the semantically segmented image using sensor calibration matrices. The class probabilities vector for the corresponding pixel is then appended to the projected point. This appending results in a separate map being created for each of the classes that the segmentation network outputs (**Figure 3.4**). Each map is the same size as the BEV occupancy map. In the case where multiple pixels are associated with the same BEV pixel, the probabilities are averaged over all the points in that pixel. These semantic features effectively help in reducing possible false positive predictions generated by radars due to poor differentiability and non-uniformity (**Section 2.1**) in radar data. **Figure 3.4** shows how these segmentations are encoded with the radar BEV map, into a semantic map, to generate features that help in object detection.

Note that this form of sequential fusion with camera does not filter out any radar points, while making better use of the advantages that both modalities bring to the table. The textural

and high angular resolution information from cameras is condensed into semantic features which assists the all-weather, long range and occlusion robust sensing of radars. In case of bad weather, where the camera based semantic features become less informative, all objects in the scene are still visible to radars.

Radar point features

As we saw in **Section 3.2.3**, the BEV occupancy grid map provides an order to the unordered radar point cloud making it convolution friendly. However, a BEV grid map also discretizes the sensing space into grids which dissolves the useful information required for refinement of bounding boxes. To retain that information we add the point based features to our BEV grid map as additional channels. Specifically, we add the Cartesian coordinates back as additional spatial information. Similar to **Section 3.2.3** we also add doppler and the intensity information as additional contextual features. The final tensor input to the network is defined as follows:

$$s := (\mathcal{I}((u, v)), d, r, x, z, y_{dist}, n, \phi(r, g, b)) \tag{3.2}$$

Here, $\mathcal{I}$ once again represents the occupancy grid and can be formulated as an indicator function defined as $\mathcal{I}((u, v))$, and d and r represent the average doppler and intensity values for pixel (u, v). Additional maps for spatial coordinates are also generated. (x, z) is the average depth and horizontal coordinate for that pixel in the radars coordinate system. However, In order to encode height information, we generate height histograms by binning the height dimension (y) at 7 different height levels. The Cartesian coordinates help in refining the predicted bounding box. The n channel contains the number of points present in that pixel which is proportional to the surface area and helps in identifying foreground objects.

In addition to the new features that are added to the occupancy grid, the key difference between the first approach and this approach lies in the fact that the entire point cloud is projected

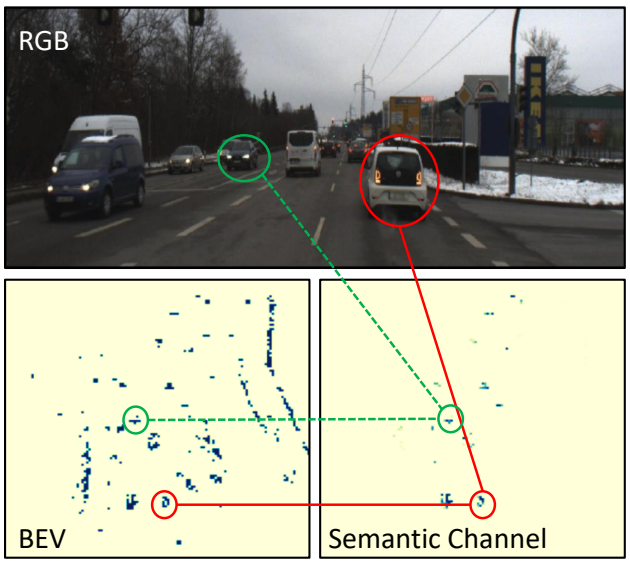

Figure 3.4: Adding the semantic channel in SPG encoding helps in identifying points belonging to objects of interest. This figure shows how two cars are represented in the semantic map for the class cars and in the corresponding BEV occupancy grid. Similarly semantic maps can be represented for other classes.

instead of just the object instances. The camera semantics are encoded as the last set of features, represented by $\phi(r, g, b)$. These are the semantic segmentation scores for each class, which is mapped to each individual radar point and then averaged over all the points belonging to the BEV map pixel. By adding camera semantics like so, we hope that the network is able to gather a sense of the lower bound to the number of cars it must detect. This idea is further analysed in **Section 5.6**.

3.3.3 Bounding Box Detection

Each of the BEV maps generated through our SPG encoding, are passed into a deep neural network for feature extraction and bounding box prediction. For our backbone feature extraction, we use a deep U-Net [21] style encoder-decoder network, just like the one in the first approach. We, once again, use an anchor box-based detection [24] architecture to generate predictions using a classification and a regression head. The classification head predicts a confidence score for the output boxes and a regression head learns to refine their dimensions. The difference between the

first and second approaches, lies in the fact that the regression head and classification heads are slightly deeper in order to give the system a chance to better understand the additional features we pass into it.

3.4 Loss functions

In both of the proposed solutions, a combination of two-loss functions are used as the objective function to train the networks. The classification head uses a focal loss [22]; while for regression head, we use a Smooth L1 loss which combines L1 and L2 losses without their individual shortcomings. The losses are given by:

$$L_{foc} = -\alpha(1 - p_t)^{\gamma} \log(p_t) \tag{3.3}$$

$$L_{Smooth_{L1}} = \begin{cases} 0.5\sigma^2\Delta^2 & |\Delta| < \frac{1}{\sigma^2} \\ |\Delta| - \frac{0.5}{\sigma^2} & else \end{cases} \tag{3.4}$$

$$L_{tot} = L_{foc} + L_{Smooth_{L1}} \tag{3.5}$$

where p_t is the confidence output of classification head, Δ is the refinement value and α, γ, σ^2 are hyper-parameters of loss funtions.

We use focal loss because it provides better results, for sparse radar point clouds, than simple binary cross entropy. This is because the sparsity in radar point clouds essentially means that the generated BEV maps contain a lot of unoccupied pixels. The focal loss parameter, α is used to tune class imbalances: in this case object vs background. Additionally, γ is used to cue the network to focus more on cases it was unable to successfully classify as objects. This translates to giving more stress on the difficult cases that the network was unable to correctly label and less on the ones it has successfully learnt to label.

Smooth L1 loss, on the other hand, contains a hyper parameter that effectively splits the loss into L1 and L2 regions, $\frac{1}{\sigma^2}$. This effectively means that when the prediction is closer to the ground truth (in the range $[0, \frac{1}{\sigma^2})$, we can use L2 loss to allow for a smoother and quicker convergence. However, when the prediction is further away (in the range $[\frac{1}{\sigma^2}, \infty)$, we can utilise the more gradual curve of L1 loss to prevent loss values from exploding [TODO: cite fast rcnn]/ over penalising the network. When combined, the resulting function is continuous and smooth (together with its derivative).

Chapter 4

Experimental Setup

4.1 Hard Fusion

4.1.1 Image segmentation network

In the first proposed solution, we need an image segmentation network to help cluster radar points. A pre-trained Mask R-CNN model was used; this model was used from the model zoo provided by detectron [39]. More specifically, the *X101-FPN* architecture, which is trained on the COCO dataset [23], for instance segmentation task was chosen for its accuracy. However, depending upon usage an alternative model optimized for speed can also be chosen.

4.1.2 Training Details

In order to train and evaluate this system, I used the Astyx Dataset [25]. More details on this dataset can be found in **Section 5.2.1**. The dataset is split using a 9:1 ratio for training and test set. As the training stage for the detection network takes the instance-wise BEV maps as inputs, this increases the size of the training dataset based on the number of object instances present in the total training set. For example, if a scene has 10 cars in it, then the architecture sees different

samples corresponding to the 10 instances. This was a huge perk for the instance wise training. This helps in a better generalization performance of the network. Random flip augmentations were performed with a probability of 0.5 during our training for better regularization. In order to train the network, the 3D ground-truth labels were used to cluster radar points and generate the corresponding BEV maps. However, while evaluating the network, the frames are passed through the clustering module before bounding box predictions can be generated for each instance observed in the clustering step. The network then learns to classify and regress anchor boxes using the same 3D label that was used to generate the BEV map.

The hyper-parameter values were ascertained empirically. The values used are as follows: $\alpha = 0.55, \gamma = 2.0, target_{IoU} = 0.5, \frac{1}{\sigma^2} = 1$. The network is trained using a learning rate of $\lambda = 0.001$ and a weight decay of $\eta = 1e-5$ for the Adam Optimiser. It was trained for around 36 hours using 4 GTX 1080TIs to reach convergence. Additionally, early stopping was employed to evaluate the system using the best model. Since the network uses an SSD as the detection head with fixed anchor proposals as the region proposals, the average dimension of ground truth labels dictated our anchor box sizes: one each for cars and large trucks and person.

4.2 Soft Fusion

4.2.1 Image segmentation network

This second approach also utilises an image segmentation network. But the key difference is that an instance segmentation approach is no longer required, a semantic segmentation network would suffice. This time, a pre-trained semantic segmentation model from the model zoo provided by the official DeeplabV3+ implementation [7, 8] was employed. More specifically, the ResNet-101 architecture [13], which is trained on the Cityscapes dataset [9] for semantic segmentation task, was

used. Once again, this model was chosen for its accuracy and generalizability. However, depending upon usage an alternative model optimized for speed can also be chosen. Our approach is agnostic to the type of network we choose.

4.2.2 Training Details

This solution was also trained and evaluated on the Astyx dataset [25]. This time though, the dataset is split using a 4:1 ratio for the training and test sets. For each frame in the dataset, the radar point cloud is processed to extract the initial feature channels. The input into the network is a (N, C, W, H) tensor where $N = 2, C = 22, W = 128, H = 128$. The channels correspond to semantic segmentation values (9), BEV occupancy grid map (1) and point features (12). 3D ground-truth labels are used to train the classification and regression heads. Since the network is similar to the one used in the first solution, the SSD module once again utilises the anchor box proposals. The same average dimension of the ground truth labels are used as the fixed anchor box sizes. In order to regress the boxes, a target IoU (Intersection over Union) of 0.5 is used determine the positive and negative examples of the anchor boxes for classification. Only the boxes marked as positive examples are used for regression loss.

The hyper-parameter values are ascertained empirically. These are: $\alpha = 0.9, \gamma = 2.0, target_{IoU} = 0.5, \frac{1}{\sigma^2} = 1$, the same as in the first approach. The network is, once again, trained using a learning rate of $\lambda = 0.001$ and a weight decay of $\eta = 1e - 5$ for the Adam Optimiser. However, the network is only trained for around 20 hours using 2 GTX 1080TIs and batch size of 2, to reach convergence. This is primarily because the size of the training set is much smaller. Early stopping is also employed to evaluate the system using the best model. Additionally, k-fold cross validation is also performed to ensure better generalizability.

4.2.3 RADIATE

In addition to the Astyx dataset [25], this second approach was also evaluated on the RADIATE dataset [32]. This was primarily done to evaluate performance in bad-weather conditions. More details about the dataset can be found in **Section 5.2.2**. In the sensor setup, for this dataset, a ZED camera is used, but it only faces in the forward direction. As a result any data that does not lie in the +ve depth direction was filtered out to account for proper data association. As a consequence of this crop, the maximum distance of evaluation is also reduced from the provided 100m to about 70.66m in order to maintain a 1:1 aspect ratio between depth and horizontal axes. The labels are also filtered out to match these new bounds. The original labels are provided as 2D pixel coordinates. Instead, they are first transformed to a world coordinate frame using calibration matrices. However, since there is no height information, we can simply use the height of the sensor as the height coordinate of the object. Once the 3D label in the world coordinate frame are determined, they can be passed into the network to train the different sections. The input into RadSegNet is a point cloud. So in order to obtain this, a CFAR filtering [TODO: cite] is performed to convert the radar intensity map into a 2D point cloud. By placing this point cloud at the sensor height, we can generate a 3D point cloud. However, the single height value implies, that the features corresponding to the height are no longer required and as a result the height distribution maps are no longer passed into the network. Additionally, the dataset did not contain any doppler information as well, so the doppler map, d could not be created. The camera semantics, on the other hand, are obtained by projecting this 3D point cloud onto the camera image using projection matrices and camera extrinsics. So the total number of input channels reduced from 22 to 14.

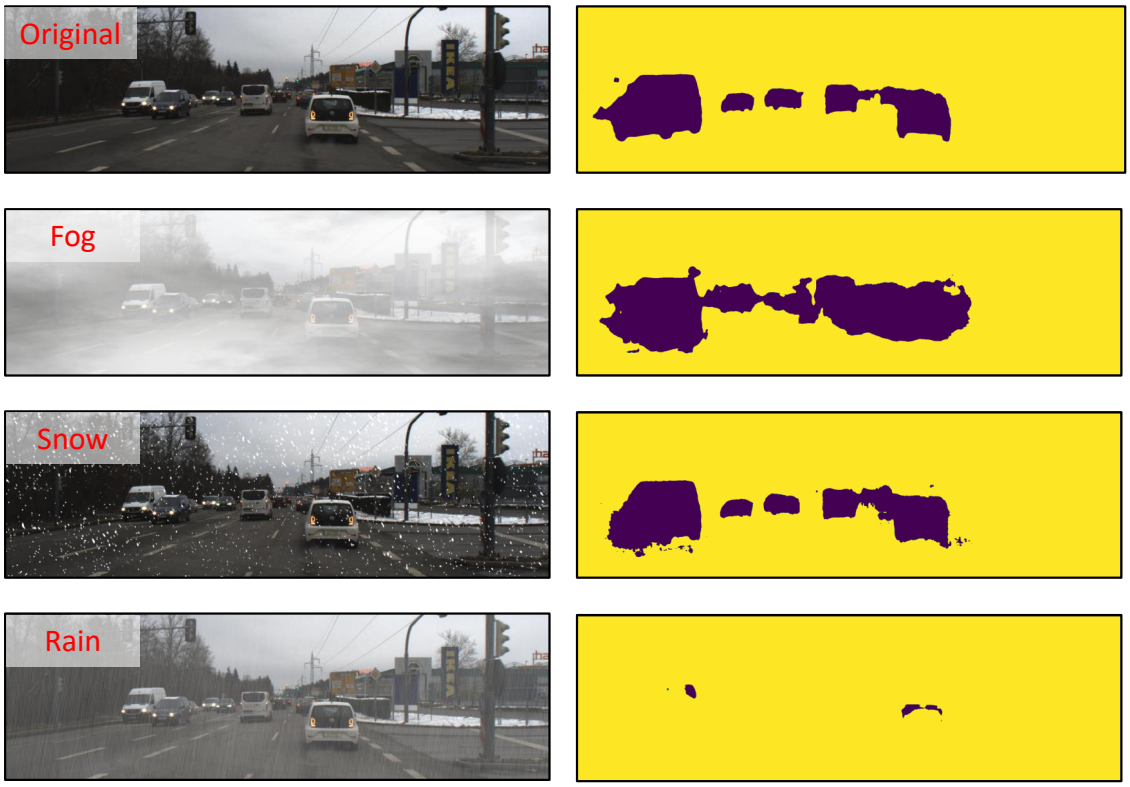

Figure 4.1: Effect on segmentation outputs for class car with different adversarial weather. Left and right column shows the RGB and corresponding segmentation output.

4.2.4 Bad Weather augmentation

In **Section 5.6.1**, an analysis of the performance of different architectures on bad weather data is provided. This analysis is performed, specifically, on the most commonly encountered extreme-weather conditions of fog, snow and rain. To generate these augmentations, the imgaug [1] library is used, which simulates realistic bad weather conditions. For foggy conditions, the $fog()$ method is used, with an argument of $seed = 5$. For rain, the $Rain()$ method with $drop\_size = (0.10, 0.20)$ is used. For snow, $SnowFlakes()$ with $flake\_size = (0.7, 0.95), speed = (0.001, 0.03)$ is used. **Figure 4.1** shows a sample output of each of these augmentations. These parameters were chosen by visual inspection; by virtue of providing the most realistic scenarios in a camera image.

[1] https://imgaug.readthedocs.io/en/latest/source/overview/weather.html

Chapter 5

Evaluation and Results

In this chapter, a comprehensive evaluation of our system is provided. In order to evaluate both the proposed solutions, the publicly available Astyx dataset [25] is used. This chapter provides the overall detection performance of the two approaches and also focuses specifically on cases of occlusions to see the advantage of using BEV representation. Due to the additional refinements made in RadSegNet over RadClustNet, RadSegNet is more holistically evaluated than RadClustNet.

5.1 Metrics

The two proposed solutions are evaluated using the BEV average precision (AP) as the main metric. AP is defined for a particular Intersection over Union (IoU) threshold of a predicted bounding box with the ground truth box. These terms are defined below:

- **IoU**: IoU (Jaccard Index) is a measure of the overlap between the predicted bounding box and the ground truth box. 3D IoU is given by $\frac{\text{Intersection Volume}}{\text{Union Volume}}$. 2D IoU is defined for BEV rectangles of 3D bounding boxes, as $\frac{\text{Intersection Area}}{\text{Union Area}}$. Two equal-sized boxes with 50% overlap would have an IoU of 0.33. Hence, even an IoU of around 0.5 is generally regarded as a good

overlap.

- **AP**: (Average Precision) AP is the area under the precision-recall (PR) curve, which is a measure of the number of actual boxes detected (recall) along with the accuracy of detections (precision). Specifically, precision is obtained for incremental recall values to get PR curve.

$$\begin{aligned} Precision &= TP/(TP + FP) \\ Recall &= TP/(TP + FN) \\ mAP &= Area(\text{precision-recall curve}) \end{aligned} \tag{5.1}$$

  A prediction is regarded as a true positive (TP) if it is above a particular IoU threshold. Note that a higher recall rate can easily be obtained by predicting a large number of boxes, but at the cost of sacrificing precision (more False Positives (FP)) and vice-versa. A higher AP means better performance on both accuracy (precision) and exhaustiveness (recall) of estimation. An FP could also be obtained because of noise. An FP generated due to noise will have a very small (almost 0) IoU with any ground truth box. Hence, in a lower IoU threshold regime, the AP is more sensitive to the amount of noise and allows one to better compare their noise suppression performance.

Generally speaking, both solutions are evaluated using two IoU thresholds, to determine true positives for AP: 0.5 and 0.1, but RadSegNet is also evaluated on an additional 0.3 threshold:

- **0.5 IoU threshold** which is the most commonly used metric for bounding box prediction tasks [**?**]. It is highly sensitive to the accuracy of the dimensions of the predicted boxes.

- **0.1, 0.3 IoU threshold** to understand the ability of the two approaches to detect the presence of different objects in the scene. Detection of objects even with a low IoU and more confidence is beneficial, especially in the case of occlusions when the complete object is not visible.

Additionally, the two approaches are also evaluated based on occlusion. While, RadClustNet is only evaluated on whether the object is occluded or not, RadSegNet is evaluated by splitting Astyx into 3 categories "No occlusion (No-occ)", "Not fully occluded (Medium)" and "Full Dataset" based on the occlusion level of the objects. This classification is based on the occlusion information in [25] provided by the authors of the dataset.

5.2 Dataset Details

5.2.1 Astyx

Astyx Hi-res radar dataset [25] is the only publicly available dataset with a high-resolution MIMO radar setup. It was collected on the roads of Germany under different lighting conditions and with an ego vehicle moving at different speeds. There are a total of 546 frames provided in the set. The dataset contains 3D bounding box labels of vehicles and pedestrians using an on-board LiDAR based point cloud. Each radar point consist of (x,y,z) location, a doppler estimate and an intensity (proportional to the power of reflection) estimate. Each frame in the dataset also has a set of labels associated to it. For each label, in addition to the position, dimension and orientation of the object, the level of occlusion is also provided: this occlusion level helps in creating splits for the different levels of difficulty. All evaluations, on this dataset, are limited to a depth of 80m. This is because the label certainties, beyond this limit, degrade drastically, due to the inability of LiDARs to maintain quality at such large distances.

5.2.2 RADIATE

Only RadSegNet is evaluated on the larger scale radar dataset RADIATE [32], with scenes captured in extreme weather conditions. RADIATE uses an unconventional and expensive mechanical radar, Navtech CTS 350-X, which is different from the commonly used High Res automotive

radar. The data is stored as 2D intensity maps, unlike the more common radar point clouds. This means that there is no height information present in the radar data, which can limit the true potential of radar data. Nevertheless, RadSegNet is evaluated on RADIATE dataset to test its generalizability on different types of radars. The dataset is split into a training set of size 16577 frames and a test set is of size 2073. In order to better understand the implementation of RadSegNet on RADIATE, please refer to **Section 4.2.3**.

5.3 Baselines

5.3.1 Perspective view

We choose CenterFusion [26], the state-of-the-art perspective view based camera radar fusion approach, as one of our baselines. In this approach, the authors create a feature map of radar point clouds and process it along with the corresponding image based feature map to perform detections. We also compare our approach against a camera only approach called CenterNet [42]. CenterNet is essentially CenterFusion without the corresponding radar data. We use the official GitHub implementation of these networks. We saw that the pre-trained network provided by the authors and fine-tune it on the Astyx dataset in order to make it a fair comparison performed better than training it fully on the Astyx dataset. Hence, we provided only the results for the transfer learning approach.

5.3.2 Multi-view aggregation Baseline

We use [15] as our multi-view aggregation based baseline. [15] uses an AVOD [17] architecture to perform radar-camera fusion. Due to the unavailability of official code from [15] we use the official implementation of AVOD and train it on the Astyx dataset to compare the performance. We dub this approach as AVOD-fusion.

Table 5.1: mAP performance on test set of Astyx dataset.

	IoU threshold 0.1	**IoU threshold 0.5**
80m max depth		
Centerfusion [26]	0.45	0.12
Ours	**0.55**	**0.20**
Capped between +-5m lateral		
Centerfusion [26]	0.51	0.13
Ours	**0.71**	**0.34**

5.4 Hard Fusion

5.4.1 Quantitative results

BEV bounding box prediction

Table 5.4 shows the performance of our approach in comparison to the baseline. We evaluate only on the objects that are in the camera field of view up to a maximum depth of 80m. We compare BEV AP for two different thresholds. The Astyx dataset [25] presents a lot of challenging scenarios with vehicles parked inside houses or on roadsides that are not labeled but are detected by our instance segmentation network which explains the low AP values for the 0.5 IoU threshold. This effect is exacerbated due to the inherent sparsity of radar point clouds that makes fitting a tight bounding box on the clusters an extremely challenging task. Despite these edge cases, we still see an improvement of 0.09 mAP over the baseline. When comparing the two over a lower IoU threshold of 0.1, our approach beats the baseline by 22%, which shows its ability to detect objects more reliably than the baseline.

Performance with distance Figure 5.1 shows the AP performance of our approach with distance. We evaluate this starting from a distance of 20m up to 80m, beyond which the radar points become too sparse. As expected, the AP drops as the distance from the sensor is increased due to point clouds getting sparser. The drop in AP can also be attributed to the missing labels in the

Table 5.2: Detection rate for optimum score threshold

Detection Rate		
	IoU >0.1	**IoU >0.5**
	Not occluded	
Centerfusion [26]	0.78	0.39
Ours	**0.82**	**0.56**
	Occluded	
Centerfusion [26]	0.45	0.21
Ours	**0.65**	**0.21**

Table 5.3: mAP performance gain by using the DBSCAN based refinement. GT (Ground Truth) clustering is the performance when clustering is performed using ground truth labels.

	IoU threshold 0.1	**IoU threshold 0.5**
Only Masks	0.47	0.18
DBSCAN correction	**0.55**	**0.20**
GT clustering	0.89	0.50

dataset for vehicles parked on either side of the roads, which are detected by Mask R-CNN and hence get labeled as a False Positive.

Performance on objects on near ego-lane: From a control system point of view, the objects of primary importance are the ones in the same or adjacent lanes [29]. In Astyx dataset, a good proportion of vehicles are parked on the roadsides. Hence, we also evaluate the performance of our approach for the vehicles within +- 5m lateral distance of the ego vehicle (approx. for adjacent lanes). Table 5.4 contains the performance comparison of our approach with the baseline for such objects. In these cases, we see an improvement of more than double AP for 0.5 IoU threshold and 39% improvement for 0.1 IoU threshold.

5.4.2 Performance on Visually Occluded objects

To evaluate the prominence of using BEV representation, we use the occlusion information provided by the Astyx to see the performance on occluded vs non-occluded objects. Table 5.2 shows the detection rate values for our approach compared to the baseline. The detection rate values are

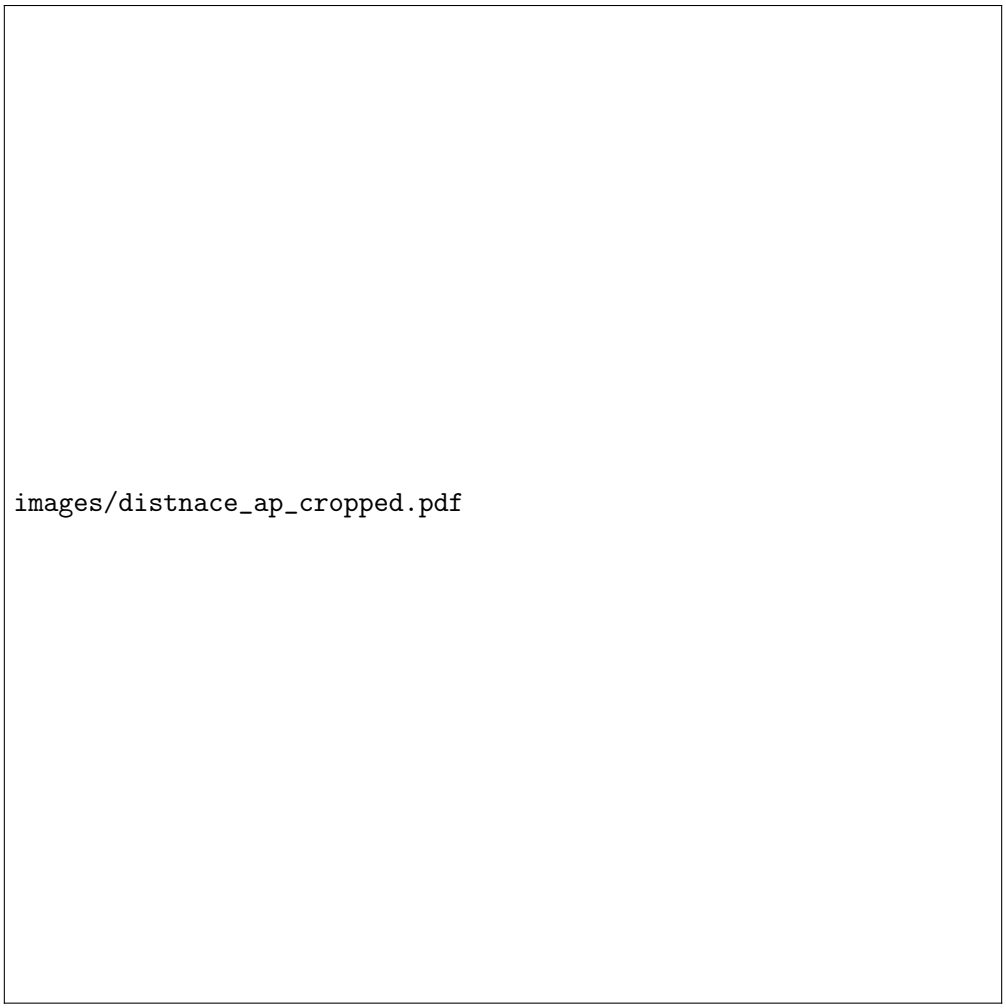

Figure 5.1: AP variation with distance for different IoU thresholds (in brackets). CF: CenterFusion. Our approach consistently performs better than baseline for all the distances.

obtained for the prediction score threshold that provided the most optimal results. For non-occluded objects we beat the baseline by 43.5% and 5.1% at IoU threshold of 0.5 and 0.1 respectively. This shows that our approach that uses BEV representation for detection can detect objects with higher IoU compared to perspective view based baseline. For occluded objects, the number of points on the vehicle are less, hence we do not see improvement for 0.5 IoU threshold. However, for 0.1 IoU threshold, there is an improvement of 44.4%. This result further bolsters the fact that bounding box prediction on radar point clouds in BEV representation is much more robust than compared to a camera perspective view-based approach while detecting occluded objects.

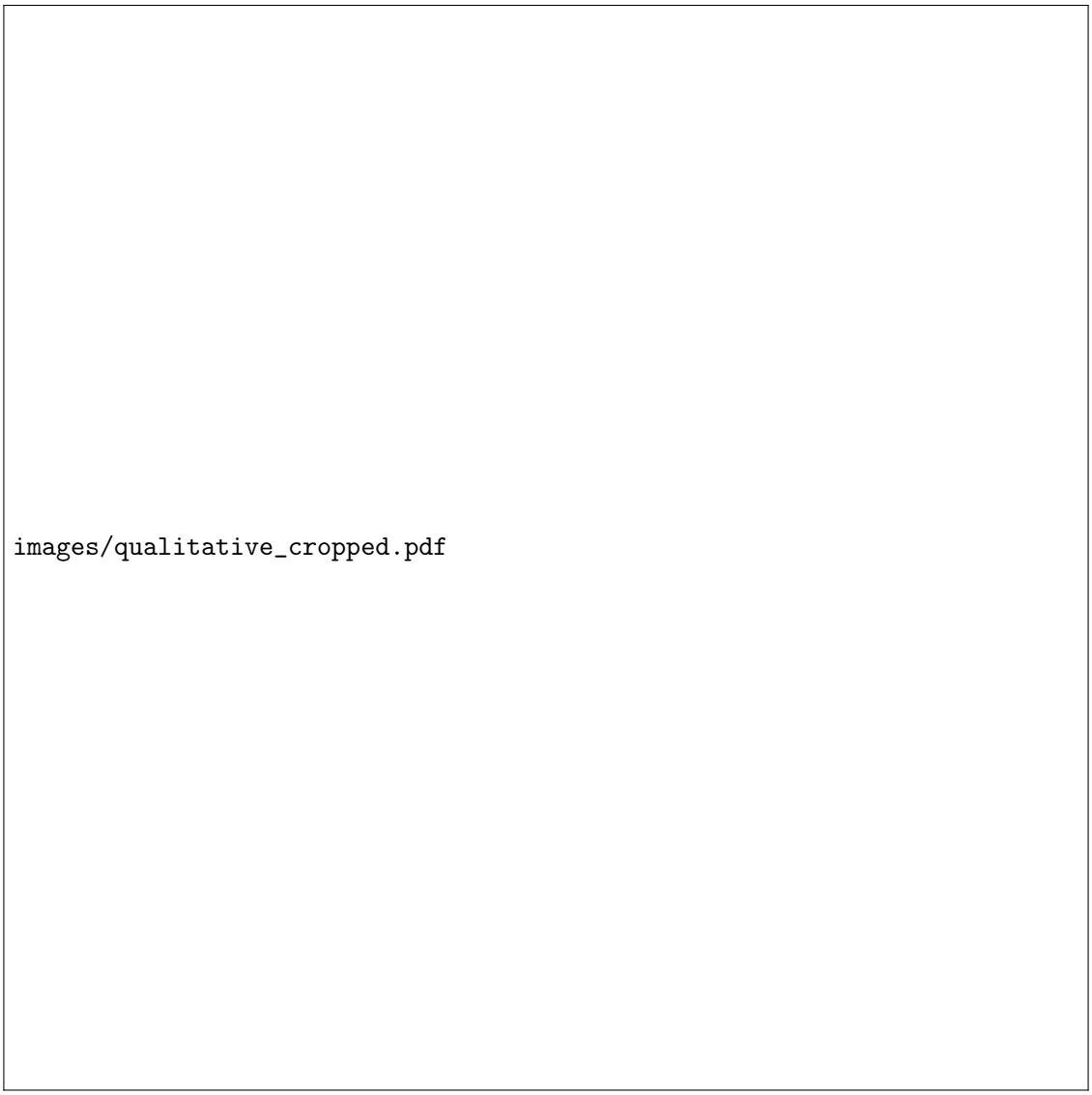

Figure 5.2: The figure shows the final outputs of different system along with the radar point cloud. Red is the predicted box while Blue is the ground truth box. We compare our predictions (last column) with centerfusion [26] (2nd column). We also show the output without the DBSCAN refinement (3rd column). Our approach maintains provides a robust prediction performance for multiple scenarios including long-range. Best viewed digitally. FP: False Positive

5.4.3 Ablation studies

In this section we evaluate different parts of our approach individually. First we see the effect of using DBSCAN based refining on the performance. Table 5.3 shows that by using DBSCAN refinement, the performance increase by 17% for 0.1 IoU threshold and 11% for 0.5 IoU threshold,

showing the ability of refinement to tackle under-segmentation.

Next we evaluate the performance of our detection network. To do so, we use the ground truth labels based clusters to create the BEV maps and use as the test set for our network. Table 5.4 shows the obtained performance. We get a performance of 0.5 AP for IoU threshold of 0.5, which shows the network's ability to generate good bounding box predictions.

5.4.4 Qualitative results

Figure 5.4 shows the qualitative output of our approach. Our approach can maintain a good detection performance for long ranges and occluded scenarios where the baseline struggles. First row shows a case of closely parked cars, where our approach can predict most of the boxes with a high accuracy. The second row shows a more common traffic scenario where far away cars are occluded in the camera view. Performing detection on radar BEV as in our approach shows a clear advantage in this case specifically for the vehicles at farther distances.

5.5 Discussion

Our approach to performing fusion for radar and camera is a sequential approach that uses instance masks to perform clustering, followed by a detection step. However, in a practical scenario, these two steps can be run in parallel by keeping one frame latency between the detection network and the image segmentation network. Past work has explored the possibility of building such a system [36] and similar techniques can be employed in our approach. Secondly, approaches like pseudo-lidar [40] convert monocular images to point cloud representations, but the problem of occlusions remains. Wang et al. [37] have shown how using a cheap LiDAR can possibly enhance performance. Our system offers a unique opportunity for such fusion of camera with radars that are low-cost sensors, provide highly accurate doppler information, and can perform sensing in all-

Table 5.4: BEV Average Precision scores for different IoU thresholds in brackets. RadSegNet outperforms the state-of-the-art architectures consistently over all IoU thresholds. R: Radar; C: Camera; L: Lidar

Method	Modality	AP (0.1)			AP (0.3)			AP (0.5)		
		No-occ	Medium	All	No-occ	Medium	All	No-occ	Medium	All
CenterNet [42]	C	34.94	39.94	39.92	25.65	28.55	28.07	12.40	13.36	13.11
Center Fusion [26]	R + C	34.12	41.75	40.64	25.62	28.87	28.04	9.78	11.22	10.87
AVOD-Fusion [15]	R + C	56.10	64.82	64.90	49.75	52.66	51.33	40.38	37.46	36.11
RadSegNet (Ours)	R + C	**63.38**	**70.82**	**70.54**	**56.78**	**56.80**	**55.80**	**48.14**	**46.82**	**45.88**
RadSegNet-Lidar	L + C	45.16	53.70	51.59	40.00	45.06	43.24	33.75	38.45	36.79

weather scenarios [5].

5.6 Soft Fusion

In this section, we provide a comprehensive evaluation of our system. We use the publicly available Astyx dataset [25] to compare our system with multiple different baselines and showcase our system's generalization capabilities by evaluating it on the RADIATE [32] dataset.

5.6.1 BEV bounding box prediction

Table 5.4 shows the performance of our network in comparison to other state-of-the-art radar camera fusion approaches. We see that the perspective view based approaches do not provide good results. This is because of their heavy dependence on cameras which leads to major performance losses in cases of occlusions and long range objects. RadSegNet beats these perspective view baselines in all categories and at all IoU thresholds. RadSegNet uses BEV representation, that performs significantly better than the perspective view, specifically in cases of occlusions. We provide visualization based comparisons in the supplementary material. AVOD-fusion [15], which also uses BEV representation from radar, performs significantly better than perspective view baselines, further illustrating the negative effects of perspective view. However, RadSegNet also outperforms AVOD-fusion across the board which shows the strength of SPG representation for radar-camera fusion that contains useful information in form of semantic and point features. Figure 5.3 shows the

Table 5.5: Ablation study experiment. We present the effect of each channel on the overall BEV AP performance of RadSegNet at different IoU thresholds.

Radar	Position	Num Pts	Camera	AP (0.1)	AP (0.3)	AP (0.5)
✓				51.45	40.34	32.52
✓	✓			53.95	43.54	33.09
✓	✓	✓		58.91	49.00	36.09
✓	✓	✓	✓	70.54	55.80	45.88

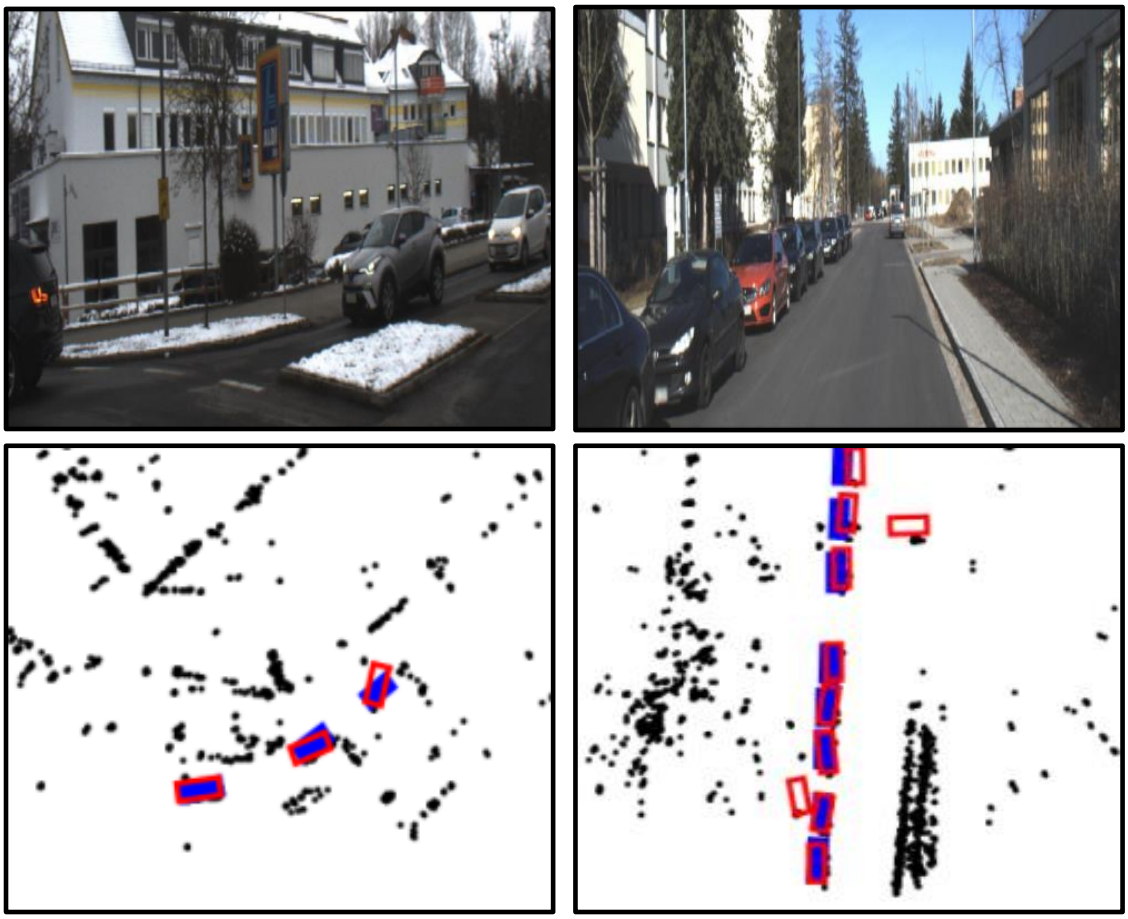

Figure 5.3: Visualization of the output bounding box output of RadSegNet on challenging cases from Astyx Dataset. Black dots represent radar point clouds in BEV. Filled blue boxes ■ are ground truth and red empty boxes □ are prediction results.

bounding box prediction outputs of RadSegNet which shows that our network can predict accurate boxes at long range, closely spaced cars and different orientations.

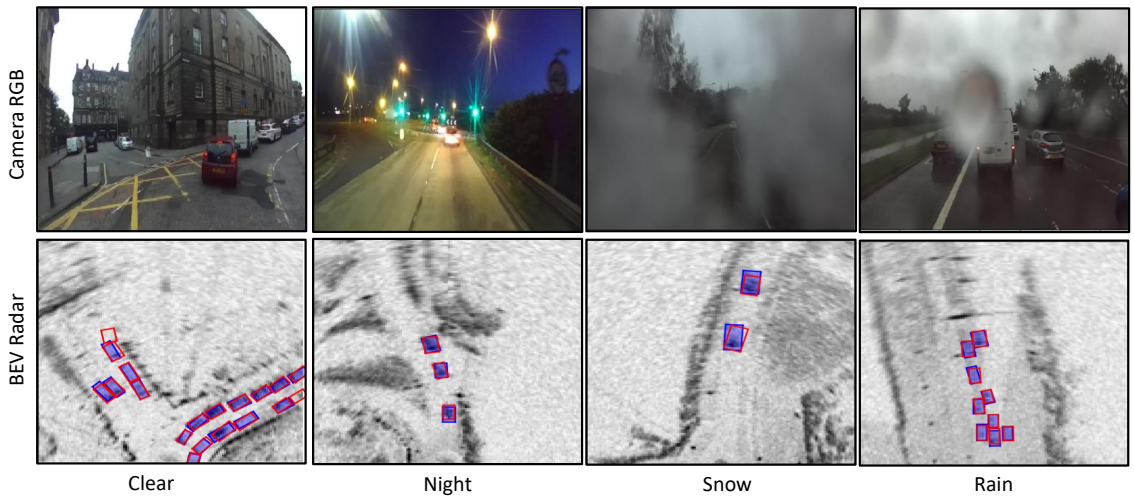

Figure 5.4: Bounding box predicition results of RadSegNet in different weather conditions on RADIATE dataset. Blue box shows the ground truth vehicle and Red box shows our prediciton.

Table 5.6: Effect on AP for Iou threshold 0.5 of different weather conditions on our architecture vs AVOD-fusion. The same input is provided to both the architectures in all the respective experiments. Percentage drop from clear weather performance in parenthesis.

	Clear	Fog	Snow	Rain
AVOD-Fusion [15]	36.11	2.38	33.56	17.41
	–	(-93.41%)	(-7.06%)	(-51.79%)
RadSegNet	45.88	43.24	43.72	32.89
	–	(-5.75%)	(-4.71%)	(-28.31%)

Comparison with LiDAR

In this experiment, we use the LiDAR data provided by Astyx as an input to without changing the architecture, to understand the advantage of using radars over Lidars. Table 5.4 shows that, while the LiDAR based object detection benefits from camera input, they perform poorly when compared to a radar's performance. The reasons for this performance drop can be attributed to the long range and occlusion free sensing of radars. Please refer to supplementary material for visualization based comparisons.

Performance in bad weather conditions

Table 5.6 shows the performance comparison of our work against the AVOD-fusion baseline. For evaluating the effect of bad weather on these systems, we generate artificial weather conditions in the rgb images using imgaug [1] library. Please refer to supplementary material for more details. As shown by past works, radar data does not get affected by these conditions [3, 5]. For each weather condition, we also show the performance drop, as a percentage, from the clear weather performance. We use the network trained on clear weather and evaluate on the test set with the generated weather augmentations. As verified in the results, AVOD-fusion's performance heavily degrades in cases of fog and rain. This is because AVOD-fusion learns feature representations that are heavily dependent on cameras during its region proposal and refinement stage. Our RadSegNet shows that its performance is almost unhindered in foggy and snowy conditions and has a reasonable performance in rainy conditions. These result shows the ability of sequential fusion to learn camera independent features from radar and perform reliably in all-weather scenarios.

Ablation Studies

In this section, we evaluate the performance gains provided by each element of our SPG encoding. Table 5.5 shows the results of this ablation study. The baseline experiment contains only the BEV map with doppler and intensity features (Radar column). The (x, y, z) (pos) maps provides an improvement of 1.76% in 0.5 IoU AP score. These channels provide the spatial context of each BEV grid pixel in the world coordinate system which is specifically helpful in bounding box refinement. The n (Num Pts) channel provides another 9.05% increase by providing information about the strength and surface area of reflection. Finally, the semantic features from camera provide significant 27.12% increase in performance, which illustrates our claim that the sequential fusion

[1] https://imgaug.readthedocs.io/en/latest/source/overview/weather.html

Table 5.7: Average Precision Results for good weather and bad weather on RADIATE dataset for IoU threshold=0.5

	Clear Weather	Clear + Bad Weather
RadSegNet	73.25	75.82

approach used by RadSegNet can comprehensively make use of the advantages of both modalities.

5.6.2 Performance on RADIATE dataset

In this experiment, we evaluate RadSegNet on a large scale radar dataset RADIATE [32], with scenes captured in extreme weather conditions. RADIATE uses an unconventional and expensive mechanical radar, Navtech CTS 350-X, which is different from the commonly used High Res automotive radar. The data is stored as 2D intensity maps, unlike the more common radar point clouds. This means that there is no height information present in the radar data, which can limits the true potential of radar data. Nevertheless, we evaluate on RADIATE dataset to test the generalizability of RadSegNet on different types of radars. We use a training set of size 16577 frames and a test set is of size 2073. Please refer to supplementary materials for details on our implementation on RADIATE.

Table 5.7 shows the break down for the performance of our network on this dataset. RadSegNet is able to achieve excellent performance with an average precision of 73+ in both clear and bad weather scenarios. Note that the frames in bad weather are collected at a different time from the clear weather frames. Hence the performance numbers can not be compared directly. Regardless, this experiment shows that RadSegNet is agnostic to the type of radar used for detection and can work with any type of radar and in all weather conditions. Figure 5.4 provides sample outputs of our network in different weather conditions. The results show that RadSegNet generalizes quite well to the new dataset and provides almost perfect detections in challenging scenarios.

Chapter 6

Future Work and Next Steps

6.1 Pointillism Dataset

6.2 V-Loss

[TODO: Add as discussion in future works and tied in with kshitiz's thesis]

Bibliography

[1] FMCW fundamentals in TI radar. https://training.ti.com/mmwave-training-series. 13

[2] Imaging radar using cascaded mmwave sensor reference design. https://www.ti.com/tool/TIDEP-01012. 13

[3] Lidar vs radar: A detailed comparison. http://robotsforroboticists.com/lidar-vs-radar/. 2, 3, 13, 51

[4] Waymo is 99% of the way to self-driving cars. the last 1% is the hardest. https://www.bloomberg.com/news/articles/2021-08-17/waymo-s-self-driving-cars-are-99-of-the-way-there-the-last-1-is-the-hardest. 2

[5] Kshitiz Bansal, Keshav Rungta, Siyuan Zhu, and Dinesh Bharadia. Pointillism: accurate 3d bounding box estimation with multi-radars. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pages 340–353, 2020. 2, 3, 15, 48, 51

[6] Thapana Boonchoo, Xiang Ao, Yang Liu, Weizhong Zhao, Fuzhen Zhuang, and Qing He. Grid-based dbscan: Indexing and inference. *Pattern Recognition*, 90:271–284, 2019. 11

[7] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 35

[8] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018. 35

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 35

[10] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, 2020. 18

[11] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981. 12

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 23

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 35

[14] Martin Holder, Philipp Rosenberger, Hermann Winner, Thomas D'hondt, Vamsi Prakash Makkapati, Michael Maier, Helmut Schreiber, Zoltan Magosi, Zora Slavik, Oliver Bringmann, et al. Measurements revealing challenges in radar sensor modeling for virtual validation of autonomous driving. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2616–2622. IEEE, 2018. 14

[15] Jinhyeong Kim, Youngseok Kim, and Dongsuk Kum. Low-level sensor fusion network for 3d vehicle detection using radar range-azimuth heatmap and monocular image. In *Proceedings of the Asian Conference on Computer Vision*, 2020. viii, 2, 4, 19, 42, 48, 50

[16] Jeong-Hun Kim, Jong-Hyeok Choi, Kwan-Hee Yoo, and Aziz Nasridinov. Aa-dbscan: an approximate adaptive dbscan for finding clusters with varying densities. *The Journal of Supercomputing*, 75(1):142–169, Jan 2019. 11

[17] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018. 42

[18] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019. 17, 25

[19] Pia Lidman and Susan Luu. *Clustering, shape extraction and velocity estimation applied to radar detections: Extracting higher order information of detections from automotive radar for tracking applications*. 2018. 12

[20] Teck-Yian Lim, Amin Ansari, Bence Major, Daniel Fontijne, Michael Hamilton, Radhika Gowaikar, and Sundar Subramanian. 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. 18

[21] Juan-Ting Lin, Dengxin Dai, and Luc Gool. *Depth Estimation from Monocular Images and Sparse Radar Data*. Sep 2020. 27, 31

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 32

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 24, 34

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 27, 31

[25] Michael Meyer and Georg Kuschk. Automotive radar dataset for deep learning based 3d object detection. In *2019 16th European Radar Conference (EuRAD)*, pages 129–132. IEEE, 2019. 7, 34, 36, 37, 39, 41, 43, 48

[26] Ramin Nabati and Hairong Qi. Centerfusion: Center-based radar and camera fusion for 3d object detection. *arXiv preprint arXiv:2011.04841*, 2020. viii, 2, 19, 42, 43, 44, 46, 48

[27] Felix Nobis, Maximilian Geisslinger, Markus Weber, Johannes Betz, and Markus Lienkamp. A deep learning-based radar and camera sensor fusion architecture for object detection. In *2019 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pages 1–7. IEEE, 2019. 2, 18

[28] Andras Palffy, Julian FP Kooij, and Dariu M Gavrila. Occlusion aware sensor fusion for early crossing pedestrian detection. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1768–1774. IEEE, 2019. 14

[29] Jonah Philion, Amlan Kar, and Sanja Fidler. Learning to evaluate perception models using planner-centric metrics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14055–14064, 2020. 44

[30] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 17, 25

[31] Nicolas Scheiner, Nils Appenrodt, Jürgen Dickmann, and Bernhard Sick. A multi-stage clustering framework for automotive radar data. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2060–2067. IEEE, 2019. 5, 12

[32] Marcel Sheeny, Emanuele De Pellegrin, Saptarshi Mukherjee, Alireza Ahrabian, Sen Wang, and Andrew Wallace. Radiate: A radar dataset for automotive perception. *arXiv preprint arXiv:2010.09076*, 3(4):7, 2020. 7, 37, 41, 48, 52

[33] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019. 26

[34] Martin Stolz, Mingkang Li, Zhaofei Feng, Martin Kunert, and Wolfgang Menzel. High resolution automotive radar data clustering with novel cluster method. 04 2018. 12

[35] Shunqiao Sun, Athina P Petropulu, and H Vincent Poor. Mimo radar for advanced driver-assistance systems and autonomous driving: Advantages and challenges. *IEEE Signal Processing Magazine*, 37(4):98–117, 2020. 5

[36] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4604–4612, 2020. 7, 29, 47

[37] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8445–8453, 2019. 16, 21, 47

[38] Zining Wang, Di Feng, Yiyang Zhou, Lars Rosenbaum, Fabian Timm, Klaus Dietmayer, Masayoshi Tomizuka, and Wei Zhan. *Inferring Spatial Uncertainty in Object Detection.* Aug 2020. 12

[39] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019. 34

[40] Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. *arXiv preprint arXiv:1906.06310*, 2019. 47

[41] Xiao Zhang, Wenda Xu, Chiyu Dong, and John Dolan. *Efficient L-Shape Fitting for Vehicle Detection Using Laser Scanners.* Jul 2017. 12

[42] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 42, 48