Detecting Road Boundaries and Drivable Regions in Challenging Weather Conditions

by

Dipkumar Patel

A thesis submitted to the
School of Graduate and Postdoctoral Studies in partial
fulfillment of the requirements for the degree of

Master of Applied Science in Electrical and Computer Engineering

Department of Electrical, Computer and Software Engineering

Faculty of Engineering and Applied Science

University of Ontario Institute of Technology (Ontario Tech University)

Oshawa, Ontario, Canada

February 2022

THESIS EXAMINATION INFORMATION

Submitted by: **Dipkumar Patel**

Master of Applied Science in **Electrical and Computer Engineering**

Thesis title: Detecting Road Boundaries and Drivable Road Regions in Challenging Weather Conditions

An oral defense of this thesis took place on February 7, 2022 in front of the following examining committee:

Examining Committee:

Chair of Examining Committee	Dr. Zeinab El-Sayegh
Research Supervisor	Dr. Khalid Elgazzar
Examining Committee Member	Dr. Moustafa El-Gindy
Thesis Examiner	Dr. Jing Ren

The above committee determined that the thesis is acceptable in form and content and that a satisfactory knowledge of the field covered by the thesis was demonstrated by the candidate during an oral examination. A signed copy of the Certificate of Approval is available from the School of Graduate and Postdoctoral Studies.

Abstract

Road detection is a core component of self-driving vehicle perception, where it covers detecting road boundaries and drivable road regions. It can also help human drivers to drive safely in lower visibility. The majority of current road detection techniques use camera and lidar sensors. These sensors struggle in inclement weather conditions. MMwave radar works well in all weather conditions. However, due to the low resolution of the radar, it is currently limited to object detection for cruise control applications. This thesis investigates the impact of bad weather on vision-based systems and introduces a camera and radar-based method for efficient road detection. We propose a novel approach to overcome the sparse resolution of mmwave-radars and use it in the segmentation task. We augment the nuScenes dataset with fog and rain and use it for our validation. We achieve 20% and 18% better road boundary and drivable region detection in inclement weather.

Keywords: road detection; autonomous vehicle; camera radar fusion; perception; mmwave radar

AUTHOR'S DECLARATION

I hereby declare that this thesis consists of original work of which I have authored. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize the University of Ontario Institute of Technology (Ontario Tech University) to lend this thesis to other institutions or individuals for the purpose of scholarly research. I further authorize University of Ontario Institute of Technology (Ontario Tech University) to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research. I understand that my thesis will be made electronically available to the public.

_____ DIPKUMAR PATEL

STATEMENT OF CONTRIBUTIONS

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication. I have used standard referencing practices to acknowledge ideas, research techniques, or other materials that belong to others. Furthermore, I hereby certify that I am the sole source of the creative works and/or inventive knowledge described in this thesis.

Acknowledgements

This work would not be possible without endless support from my supervisor Dr. Khalid Elgazzar. His continuous support, motivation, and guidance helped me shape this research. I can't thank him enough for his invaluable contribution in fulfilling this dissertation. This thesis dissertation would not have been possible without his advice and continuous assistance. I am also thankful to course instructors in my master's course. The courses offered by Dr. Akramul Azim, Dr. Faizal Qureshi and, Dr. Masood Makerchi were extremely helpful in understanding the basics of the area of research. I want to express my gratitude to the members of the IoT lab whose efforts inspired me to go beyond my comfort zone and who guided me through each stage of my research. Finally, A huge thanks to my family for their sacrifices that enabled me to pursue my academic career through various challenges. I hope this work will make them proud.

Table of Contents

List of Tables

List of Figures

List of Abbreviations and Symbols

ANNs Artificial Neural Networks.

AVS Autonomous vehicles.

CNNs Convolutional Neural Networks.

CPN Convolutional Patch Network.

DARPA Defense Advanced Research Project Agency.

ELM Extreme Learning Machine.

FMCW Frequency Modulated Continuous Wave.

GAN Generative adversarial Network.

IoU intersection over union.

LIDAR Light Detection and Ranging.

MVCNN Multi-view convolutional neural network.

RADAR Radio Detection and Ranging.

ROS Robot Operating System.

SAE Society of Automotive Engineers.

SLAM Simultaneous localization and mapping.

YOLO You Only Look Once.

Chapter 1

Introduction

1.1 Introduction

The operational functionality of Autonomous vehicles (AVS) can be divided into four blocks: perception, localization, planning, and control. Perception refers to providing the current state of an environment using different sensors such as camera, lidar, radar, IMU, GPS and more to detect surrounding objects such as pedestrians, traffic lights, vehicles, roads, buildings, etc. The localization stack locates the vehicle's position and projects it to a centimeter-level accuracy on a map. Having such a high level of accuracy helps AVS to comprehend the surrounding environment. The path planning stack analyzes the trajectories of nearby vehicles and predicts the vehicle's future trajectory. Based on the generated trajectory, the control stack directs the vehicle. Localization and path planning stacks utilize the output of the perception stack to localize the vehicle withing cm level precision and generate a trajectory to follow. The inability of autonomous vehicles to perceive any perceptual operation may result in their failure. The Society of Automotive Engineers (SAE) classifies autonomous vehicles into five distinct categories, level 0-5. Level 0 refers to vehicles

that do not have any automation, while level 5 refers to fully autonomous vehicles. Fully autonomous or level 5 autonomous vehicles require a highly precise perception of the environment to drive in all weather conditions (e.g., rain, fog, storm, snow, etc.) without any human inputs. Level 4-5 autonomy vehicles should have a robust road detection system that can work well in all weather conditions. Road detection is a core component of self-driving vehicle perception, where it provides road boundaries and drivable road regions where a vehicle can drive safely. Road detection has become a hot research topic for the last few years. Researchers use different methods and sensor stacks for road detection. This thesis focuses on applying computer vision and deep learning techniques to detect road boundaries and drivable road regions in inclement weather conditions.

1.2 Motivation

Self-driving vehicles have the potential to fundamentally alter the current transportation system. AVs in conjunction with connected vehicle technologies can contribute to the development of a more efficient transportation system. Road detection not only helps AVs but helps human drivers to drive safely in bad weather conditions when vehicles and the road ahead are obscured. For example, knowing the road boundaries enables snowplow vehicles to clean the road more precisely, thereby increasing the amount of drivable area available during the winter. Current AVs perception methods work well in mapped areas such as highways, but they suffer when navigating in unmapped urban areas or challenging weather conditions. In such environments, road detection becomes a critical part of the perception stack because it provides a drivable area where planning algorithms determine the future trajectories. Bad weather conditions lead to lower visibility and current approaches suffer in such conditions to

provide accurate detection which may lead to wrong real-time decisions and fatal accidents. This is a challenging task due to the wide range of environmental conditions that could affect the quality of detection. Environmental conditions such as rain can either reduce the detection capability or provide false detection as sensor signals get reflected when these signals come in contact with raindrops. In our thesis, we develop a computer vision and deep learning based approach for road detection. We divide our work into two components:

1) road boundary detection - Road boundaries provide the boundaries in which a vehicle should drive

2) drivable road region detection - the drivable road region further detects the space within the road where a vehicle can drive.

1.3 Problem Statement

Many works that we preview in chapter 2 introduce various solutions for road detection. A popular sensor suite for this purpose is a combination of camera and Light Detection and Ranging (LIDAR). This solution provides impressive results in daylight conditions but suffers in inclement weather. The camera, on one hand, provides impressive results in the daytime but suffers in weather conditions such as rain, fog and snow when the sensor gets covered by fog and raindrops. Lidar works well at nighttime, but it is quite expensive as the average cost of lidar is between $5,000 and $50,000. Lidar systems with higher performance, such as Velodyne lidar, can cost up to $75,000 per vehicle [3]. Lidar maps the environment using light pulses. This can cause light signals to be easily affected by the medium that significantly influences Lidar signals. A sensor that can detect effectively in bad weather is needed, and an automotive radar may be an alternative. Radio Detection and Ranging (RADAR)

operates based on radio signals, which have higher bandwidth and are less susceptible to attenuation. This means that radar signals can travel long distances with little interference during inclement weather. This characteristic helps radar in providing better object detection in adverse weather conditions. Compared to lidar, automotive radars are less expensive and their price ranges between $50 and $800 at the high end. Therefore, the use of radar over LIDAR can significantly reduce the cost of manufacturing autonomous vehicles. However, due to the comparatively lower resolution of the radar, it is currently limited to object detection for cruise control applications as high resolution is a fundamental requirement for perception tasks such as road detection. Figure 1.1 shows the resolution comparison between lidar and radar taken from the nuScenes dataset [10] and it is clear that the resolution of automotive radar is significantly lower compared to lidar. In conclusion, lidar has a greater resolution but suffers in inclement weather, whereas radar operates in any weather but has a lower resolution and this makes the road detection task more interesting.

In our thesis, we define the challenges that we address in our solution as follows:

1. Automotive radar's mmwave bandwidth range provides better object detection, but the same narrow range introduces sparse resolution as well. Additionally, the vehicle's speed and the range of radar detection are inversely proportional. For example, as vehicle speed increases, the radar detection range decreases.

2. Optical sensor based approaches are promising during the daytime, but they suffer in adverse weather conditions such as rain, fog, or low visibility at nighttime. Therefore a supportive sensor is required along with optical sensors to provide accurate road boundary detection in all weather conditions.

3. Road region detection requires classifying each pixel of a road image to identify the boundaries in the bad weather conditions. It becomes even more challenging

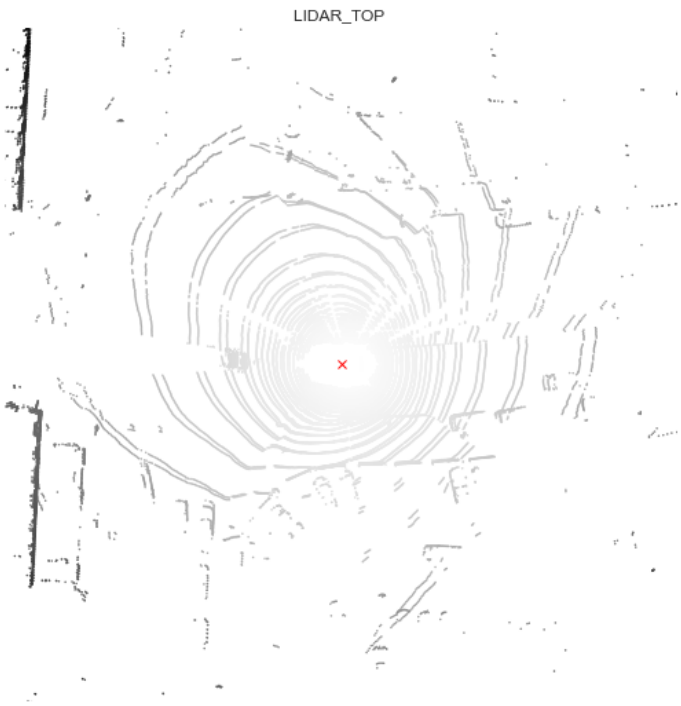

Figure A - Lidar sample

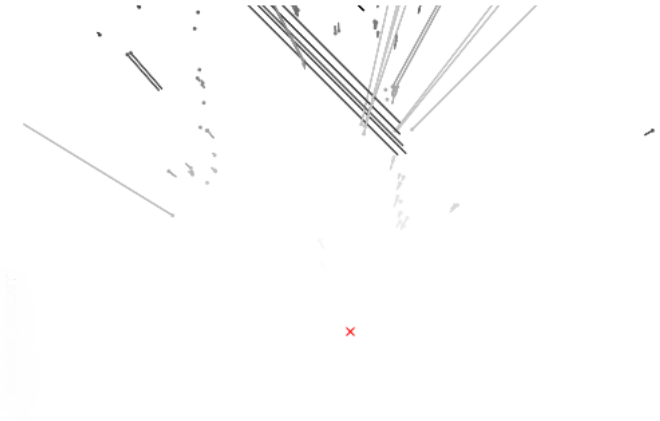

Figure B - Radar sample

Figure 1.1: Lidar and radar resolution comparison. Fig.A depicts lidar and Fig.B shows radar points

as they block optical sensors. A new sensor pair and more complex deep learning based approach are required to process complex inputs, implement more layers, and address the gradient vanishing problem.

4. Deep learning methods are data-driven methods that need a large dataset to learn more complex tasks such as road region detection. To the best of our knowledge, there is no public dataset that provides camera, radar, and lidar based collective sensor data recorded in various bad weather conditions.

1.4 Thesis Contribution

Our main contributions in this thesis are as follows::

1. We developed an integrated solution for accurate road region boundary detection in all weather conditions using camera and automotive radar. We use You Only Look Once (YOLO) based object detection to detect road boundaries in bad weather conditions. We demonstrate how our approach can work well in a darker environment and other harsh weather conditions.

2. We developed a novel deep learning approach using a camera and a mmwave radar to detect road drivable regions. The proposed deep learning method utilizes multi-modal inputs and an encoder-decoder based model to provide robust road region detection even in challenging weather conditions.

3. We developed a Kalman filter-based approach for continuous radar point tracking to improve the resolution of the automotive radar sensor.

4. We provide a small camera and radar sensors dataset collected by running a test vehicle in different light conditions and augmented nuScenes [10] dataset, which is ingested with rain and fog to train a deep learning model.

5. An open source Robot Operating System (ROS) package to filter radar and lidar pointcloud points based on position, velocity, and by object type. These utilities can help set up automotive radar easily in a perception sensor fleet of self-driving vehicles.

1.5 Thesis Organization

This thesis is organized as follows. Chapter one introduces road detection, the research gaps and open challenges in current research, and highlights our contributes. Chapter two provides background related to the proposed work and reviews the literature for related work. Chapter two includes an introduction to autonomous vehicles, sensor fleet, and the use of deep learning in autonomous vehicles. The related research in detecting road boundaries and road regions as also described. We present our end-to-end framework for road detection in Chapter three and showcase our approach to road boundary and road region detection. We begin by demonstrating the performance degradation of the state-of-the-art in inclement weather conditions and then introduce our proposed approach to improve the detection. Chapter four provides the performance evaluation of our work. Lastly, we conclude with Chapter five, providing some insight into our methodology, how it is successful, its potential limitations, and where our work can be extended in the future.

Chapter 2

Background and Related Work

In this chapter, We present the literature review for road boundary and region detection techniques. We use classical computer vision and YOLO based object detection to perform road boundary detection and a deep learning based approach for road region detection. This chapter acts as a review section, providing context for the proposed research. First, we introduce the autonomous system, then we provide the basics of deep learning used in the current state-of-the-art and our research. Finally, we discuss the related work in road detection.

2.1 Autonomous vehicles

Defense Advanced Research Project Agency (DARPA) grand Challenge for Autonomous Vehicles sparked interest in the technology, which has grown in popularity ever since [1]. Previously, DARPA coordinated three competitions. The first DARPA Grand Challenge took place in 2004 in the United States' Mojave Desert and required self-driving cars to cover 142 miles across desert trails in ten hours. However, no participant vehicles were able to complete the race. In 2005, the second DARPA

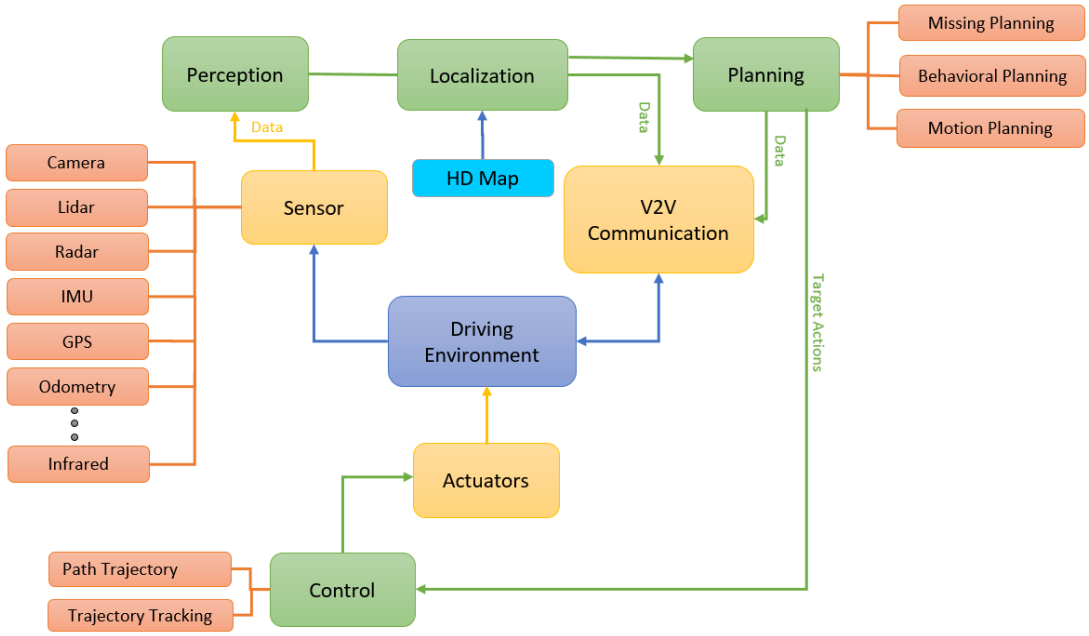

Figure 2.1: Architecture of autonomous vehicle system. Green color blocks are primary blocks and yellow color stacks indicate intermediate steps. The optional blocks are depicted in blue color, while supporting blocks are highlighted in red

Grand Challenge required autonomous vehicles to travel 132 miles across flat terrain, mountain passes, and more than 100 sharp left and right turns. There were 23 finalists in this competition, and four cars completed the route within the time limit. Several well-known universities in the United States of America competed with their cars in this challenge, with Stanford University's car named 'Stanely' [2] taking first place. This competition resulted in significant advancements in this field.

Autonomous vehicles perceive the world through a fleet of sensors and intelligent software. An autonomous vehicle can be considered a real-time safety-critical system. There are two approaches to design autonomous vehicle software, mediated perception and behavior reflex [12]. Mediated perception is a modular approach that independently parses each component of the system. Tasks such as traffic light detection, lane detection, and road region detection are considered separate tasks, and

their outputs are combined to decide how to control the vehicle. Where the behavior reflex approach connects sensor inputs directly to driving actions. This approach can achieve high throughput using huge data sets. But, because it provides driving instructions directly, it is harder to debug than mediated perception. The approach of mediated perception is used widely in industry and academic research. The mediated approach is the focus of our thesis research. Figure 2.1 provides the cognitive cycle of an autonomous vehicle. The figure shows the functionality of autonomous vehicles primarily in four categories: perception, localization, planning, and control. The **perception** module processes data from various sensors but is not limited to camera, lidar, radar, GPS, and an IMU. The perception module is in charge of processing inputs from these sensors and passing the detection results to subsequent modules. The perception module can perform a variety of complex tasks, including lane detection, traffic sign classification, light detection, pedestrian detection, vehicle detection, road boundary detection, and free drivable space detection. No other sensor is better at detecting the text on traffic signs or other roadside warning and information signs than the camera. As a result, the camera is a key sensor for a self-driving vehicle. However, as previously noted, they suffer under low light circumstances. As a result, an additional supplementary sensor is necessary to detect the surroundings in all weather conditions.

Environment perception is still an active research area, and up till now, automotive manufacturers have had a preference to use one sensor over another. For example, Tesla prefers radar to lidar, while Waymo favors lidar for its perception stack. Proposed work utilizes automotive radar along with a camera for the perception task. Perception output is also fed into the V2V(Vehicle to vehicle) communication stack to share the perception of one vehicle to other vehicles to reduce computation and generate insights ahead of the vehicle. The **localization** module precisely locates the

vehicle in the world by utilizing a high-definition map and inputs from the perception stack. It uses different localization approaches such as Simultaneous localization and mapping (SLAM), Adaptive Monte-Carlo localization, Markov localization, and Kalman filter localization. If a mismatch in position while mapping is detected, it notifies the mapping team to update the map for future use. The **planning** stack determines the future trajectory of the vehicle based on inputs from the perception and localization stack. It uses the data such as the positions of stationary and moving objects, lane and road detection, and vehicle speed to forecast the future trajectory of the vehicle and devise a plan to avoid colliding with other vehicles. The **control** module directs the vehicle to follow a projected trajectory determined by the planning stack. It is in charge of the vehicle's throttle, brake, steering, gear selection, horn, and lamp. It also takes into account the smooth driving experience when making a decision.

2.1.1 Sensors

We, humans, use our eyes to perceive the surroundings, and our brains use these visuals to generate awareness. Likewise, autonomous vehicles perceive their surroundings using a variety of sensors. It makes use of different sensors to assess the environment and locate the vehicle within it.

Camera depicts an environment visually. There are numerous types of cameras, including monocular cameras and infrared cameras. They enable a more nuanced understanding of a scene and have an excellent range and resolution. This sensor measures the amount of light reflected from an object. Variation in light, such as illumination, can affect the camera's performance. Under ideal lighting conditions, no sensor can supply more information than a camera, and it is the only sensor

capable of parsing text that makes it the default sensor for automotive vehicles.

Lidar provides high-resolution depth estimation. The term "Lidar" refers to light detection and ranging. It measures the depth and range of objects in the frame by using light as a pulsed laser beam. To calculate distance, lidar uses light pulses to send towards an object and measures the time it takes for the pulse to return. It sends these signals to all angles and generates a 2D/3D map of the environment based on the angular resolution of the lidar sensor. The generated map is a high-resolution map that is extremely useful for perceiving the environment and is also capable of working in low-light conditions. It is widely used in drone-based applications to prepare 3D maps of dangerous or unmapped areas, as well as in the automotive industry. Lidar prices are determined by the number of scanning beams present in the sensor. Unlike the visual presentation provided by a camera, lidar provides a pointcloud (An array of detected points in which each element represents a characteristic of a point such as distance, intensity, or other properties). It is more difficult to process a pointcloud than to analyze visual features.

Automotive radar also provides a pointcloud representation of an environment. It provides good results for mid and long-range detection. It is much cheaper compared to lidar. Unlike lidar, radar uses radio frequencies to detect objects. There are different types of radar, but most automotive vehicles use Frequency Modulated Continuous Wave (FMCW) radar. Instead of sending simple signals and measuring reflection time, FMCW radar sends chirps. A chirp is a sinusoidal pulse that grows in frequency linearly with time, and it measures the difference in frequency between transmitted and reflected pulse. Automotive radar consists of the transmitter and receiver antennas, signal processor, power section, and a digital interface. Figure 2.2 depicts the Texas Instruments AWR1843BOOST radar sensor used in our research.

It is a single chirp FMCW radar that operates between 76 and 81 GHz that features four receive and three transmit channels. That means it can perform 3D positional and 1D Doppler measurements. It generates a 3D pointcloud with distance, angle, and intensity values for each point.

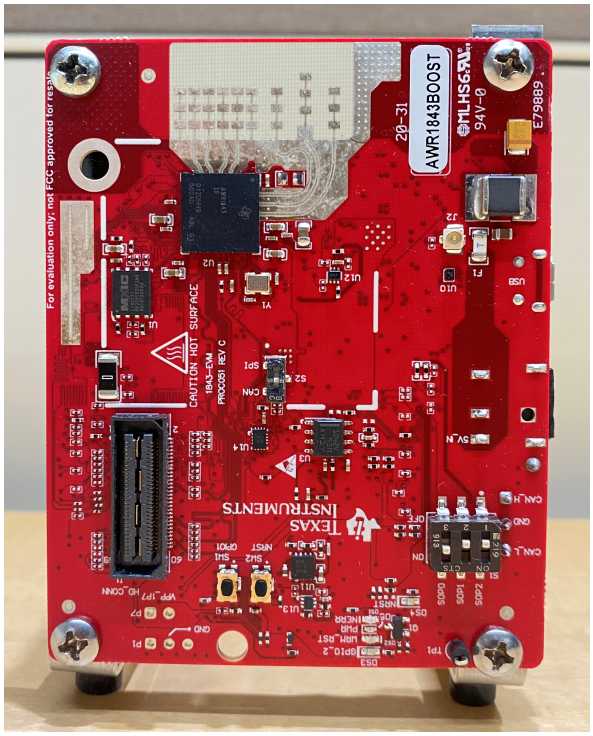

Figure 2.2: AWR1843BOOST radar evaluation board from Texas Instruments

Camera, lidar, and radar sensors are capable of mapping the surrounding environment in great detail. However, additional sensors are combined with the aforementioned sensors to reduce the system's complexity and provide more robust localization. Some of these sensors are as discussed below:

Inertial measurement unit(IMU) provides vehicular motion information. It internally uses an accelerometer, gyroscope, and magnetometer to provide acceleration, angular velocity, and magnetic orientation respectively. It assists in determining the position of a vehicle, but it introduces accumulated localization errors over time. The

introduced error can be corrected using the wheel encoder sensor.

Wheel encoder determines the linear distance traveled by a vehicle over some time between observations. It is a rotation optical encoder attached to the wheels of vehicles to calculate the linear displacement of the vehicle based on the diameter of the vehicle. It is used in conjunction with the IMU, where the IMU assists in determining angular displacement, whereas the encoders in the vehicle provide linear displacement. Together they help eliminate displacement error.

Global positioning system(GPS) helps locate a vehicle with meter level precision. The GPS's accuracy is contingent upon its connection to a satellite at any given time. Autonomous vehicles require precision down to the centimetre level, and due to its low accuracy, it is used in conjunction with other sensors to determine the vehicle's location.

2.2 Deep learning for automotive system

Deep learning is a subfield of AI and ML, where it learns the underlying features of data by using deep neural networks. The use of AI in autonomous vehicle perception consists of but is not limited to detecting traffic lights, traffic sign classification, lane prediction, road boundary detection, and road region detection. Fully autonomous vehicles need to be as intelligent as humans. To reach this intelligent level, AI provides the required intelligence in the system. We can not derive or set driving rules for all kinds of driving conditions in a dynamic environment. Deep learning helps learn rules by deriving them from a large dataset. It also adapts over time from the human inputs on various decisions and also from an additional dataset. This section provides high-level detail of CNN and auto-encoder which forms a base of our proposed deep

learning technique.

2.2.1 Convolution neural network

Convolutional Neural Networks (CNNs) are a subset of Artificial Neural Networks (ANNs) designed specifically for image processing. CNNs analyze and classify images according to their similarity clusters and recognize objects within a frame. Figure 2.3 shows the different layers and their interconnection in CNN, where they consist of different layers: input, convolutional, pooling, and fully connected.

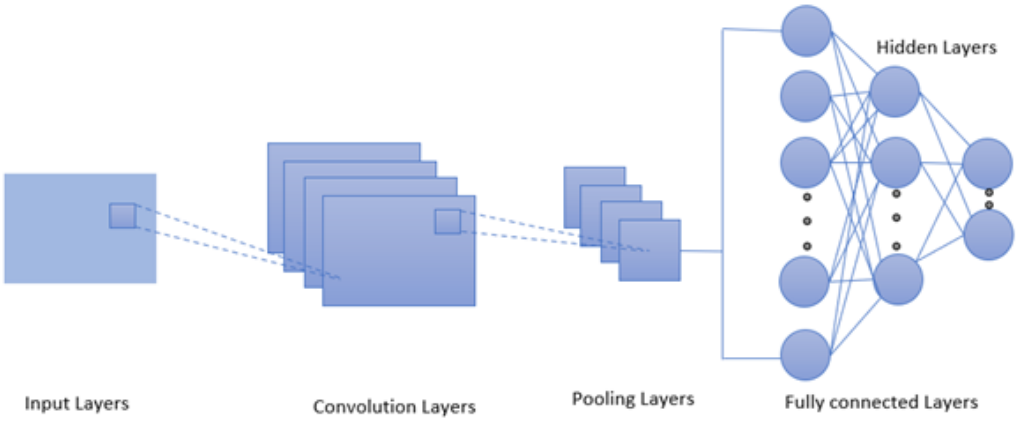

Figure 2.3: Convolutional neural network architecture

The input layer stores the image's pixel values. When an image is used as input, it will have m, $n \times n$ pixel matrices of an image, where m represents several channels and n represents a dimension of an image. The convolution layer detects image features such as edges using filters (Kernels, feature detectors). The filter iteratively scans the image pixel to determine the presence of the features it is designed to detect. The filter performs a convolution operation to generate a value indicating the degree of confidence if a particular feature is presented. Convolution computes a scalar product of the input weight and the region connected to the input volume. The output matrix is referred to as a feature map. Figure 2.4 shows the flow of feature maps generated

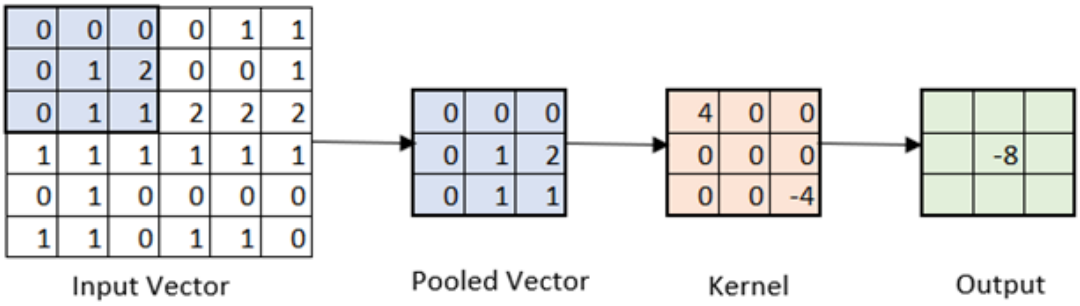

Figure 2.4: CNN feature maps generation flow diagram. Kernel operation is shown in red color

from the input vector.

By sliding a two-dimensional filter over each channel of a feature map, the pooling layer summarizes the feature laying in feature maps. It downsamples the feature map while simultaneously reducing the size of the image and preventing model overfitting. The max-pooling operation selects the largest element within the filter's feature map region. The output of Max pooling will contain the most notable features of the feature maps. The figure 2.5 shows the max-pooling layer operation. It shows how a filter of 2x2 and stride of 2x2 generates the output.

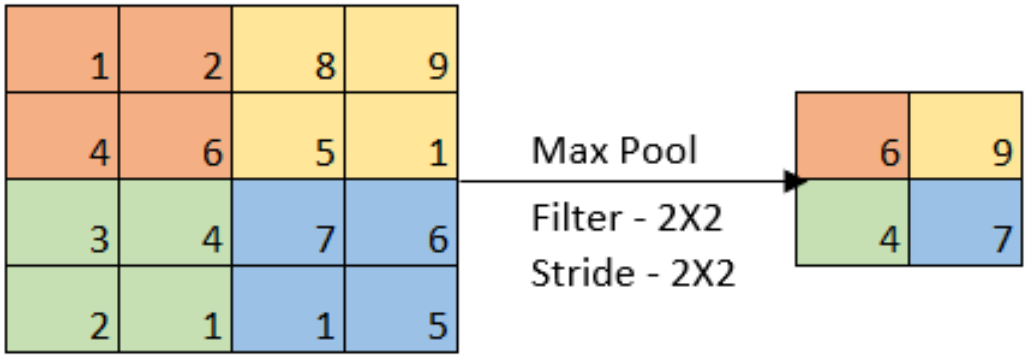

Figure 2.5: Max pooling operation in CNN. The filter and stride 2x2 operations have been utilized to reduce the 4x4 pixel matrix to a 2x2 matrix

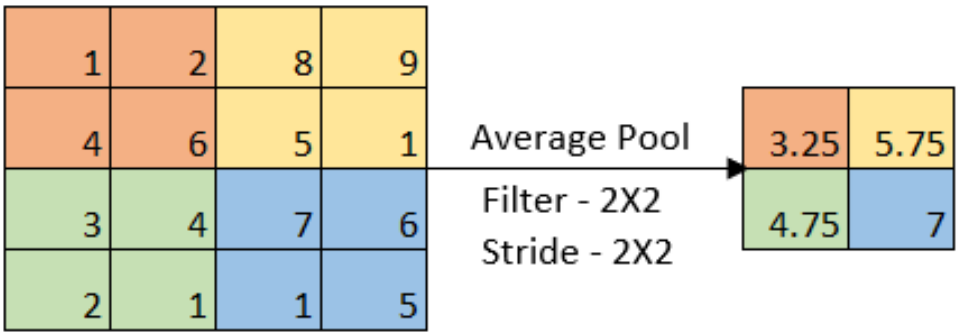

Figure 2.6: Average pooling operation in CNN. The average of each 2x2 matrix has been extracted into a single cell using the filter and stride of the size 2x2

Average pooling calculates the average of the elements in the filter's feature map region. Using the average pooling method will cause the image to be smoothed out, causing the sharp features to be lost. The figure 2.6 shows the average pooling operation.The mathematical formula for the output dimensions obtained after a pooling layer is given in the equation 2.1.

$$Dimensions\ of\ output = ((H - F + 1) \div S) \times ((W - F + 1) \div S) \times C \qquad (2.1)$$

where: H = Height of feature map

W = Width of feature map

C = Number of channels in the feature map

F = Filter size

S = Stride length (If the kernel moves over the matrix 1 pixel at a time than stride length will be 1)

The Fully Connected layer is completely interconnected with the preceding layers. It is similar to the hidden layers in a neural network, except it is completely connected. The data is transmitted over the network, and the prediction error is calculated. The forecast is then improved by back propagating the error through the system. To predict the input image, the output must be reduced to a value between 0 and 1. This is accomplished using the softmax activation function, which converts all the final outputs to a vector and then sums them into a single element. Softmax activation function can be derived as equation 2.2. where x denotes each element in the outputs of the final layer.

$$\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}} \tag{2.2}$$

2.2.2 Autoencoder

Autoencoder is a term that refers to an unsupervised neural network. It is a subtype of a feedforward neural network that produces the same output as its input with the least amount of distortion possible. In a nutshell, an autoencoder learns a low-level representation for a high-dimensional data set to comprehend and visualize complex correlations and relationships between data. To segment images, reduce their dimensions, denoise them, and extract their features, we used Autoencoder.

The architecture of Autoencoder is illustrated in Figure 2.7. Autoencoders are composed of an encoder, a bottleneck, a decoder, and a loss reconstruction component. Both the encoder and decoder are composed of fully connected feedforward neural networks. The Bottleneck is composed of a single-layer ANN (Artificial neural network). Before training, the number of nodes in a bottleneck must be determined. In the beginning, the input x is passed through the encoder, also denoted by $G\varphi$. Compression and reduction of input dimensions are performed by the encoder to pro-

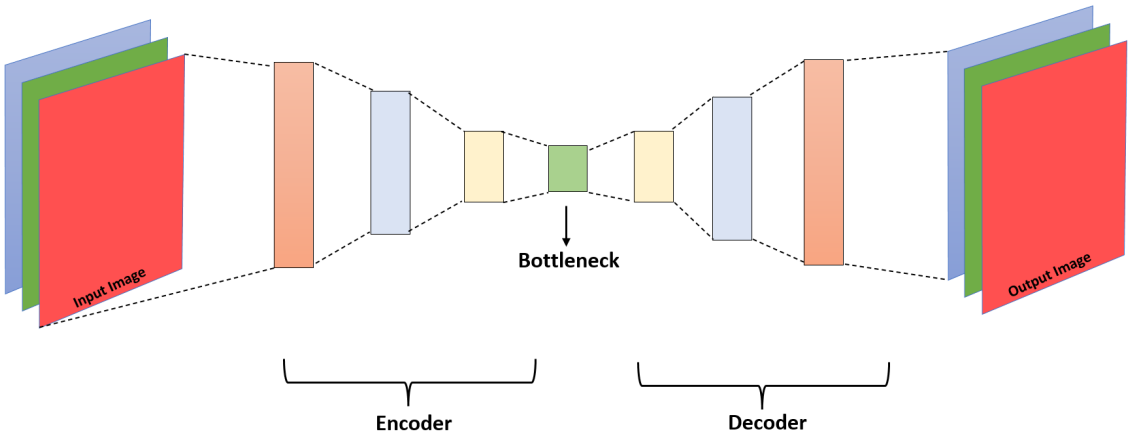

Figure 2.7: The architecture of the auto encoder

duce a lower dimensional encoding. The output encoder block is routed through the Bottleneck, denoted by Z. It is a lower dimensional hidden layer with fewer nodes. The output of Bottleneck is given to decoder $F\theta$, which reconstructs the input x and provides x' in such a way that $x \approx x'$. Once an image is reconstructed, it is critical to compare it to the original and determine the loss, which is calculated using the following equation 2.3. The overall aim is to minimize the Loss equation.

$$L(\theta, \varphi) = \frac{1}{n} \sum_{i=1} (x_i - f\theta(g\varphi(x_i)))^2 \tag{2.3}$$

It is obvious from the preceding formula that the loss function is dependent on θ and φ. The difference between the original image, x, and the regenerated image, $f\theta(g\varphi(x))$ is summed in the equation.

2.2.3 Pointcloud processing

ROS [47] provides packages and messages that enable the integration and fusion of several sensors. The majority of vendors provide ROS packages with their sensors as well. ROS handles lidar and radar sensors data in form of pointcloud. In a

scene, a pointcloud represents a collection of detected points and their associated detection information. ROS provides a standard datatype to process point clouds. The data type structure is available on its website, [47]. The figure 2.8 shows sample data stored in a pointcloud ROS message for automotive radar used in our research. Figure 2.8 shows the ROS message header, which stores the sequence, frame id, and frame dimension. Before writing row data, it includes a detailed description of each parameter and then displays the number of points and total row data count. As we can see, each pointcloud array contains encoded row data, and even after pre-processing, training a deep learning model in a pointcloud becomes complex.

Several deep learning networks can process complex pointcloud data in a variety of data formations. Zhang et al. [61] provides an in-depth examination of these techniques. These deep learning models convert the raw data to voxel grids, meshes, or multi-view formats and then pass it on to the deep learning model as input [14, 18, 46, 48, 65]. Multiview format converts three-dimensional pointcloud data to two dimensions and sends it to a deep learning model. Su et al. [48] propose an Multi-view convolutional neural network (MVCNN) model for classifying and segmenting input from a pointcloud. It converted a three-dimensional point cloud into multiple two-dimensional images taken from various angles and then extracted features from each perspective. The extracted features were then combined using the view pooling layer and fed into CNN for image classification and segmentation. Additionally, pointcloud multi-view can be combined with camera images to improve performance. Chen et al. [14] propose a fusion approach for object detection using multi-view lidar pointcloud views and a camera. Another possible input variation is to first create a voxel grid and then use it for deep learning. A voxel grid is a three-dimensional pixel representation of an unstructured pointcloud that is grid-based. Voxelnet [65] proposed the voxel-based method and after that several variants of voxelnet proposed

Figure 2.8: A sample ROS pointcloud data of a scene. Data field shown in the right side is a continuation after row_step field. Each pointcloud point has four properties: x, y, and z, as well as an intensity value

that focus on the variety of ways voxel grids can be generated [18, 46].

Instead of pre-processing the pointcloud, it can also be used as raw input to a deep learning model. Pointnet [41] propose a model capable of consuming raw pointcloud data, which represented a giant leap forward in the use of pointcloud data in deep learning. Pointnet [41] consumes 3D pointclouds directly, without pre-processing. It addresses the primary issues associated with using raw pointcloud as input. These issues include sparsity, point-to-point connection to form a shape and shape change when rotated at a different angle. The authors propose a multi-layer perceptron for extracting features from individual points and then aggregating this data to obtain high-level features via the max-pooling layer. However, pointnet struggles to maintain neighborhood information, and to address this, researchers have proposed various Pointnet variants [29, 41]. We convert the pointcloud points in an image format and project each point on the blank image on its respective transformed position.

2.3 Related Work

This section provides related research efforts on road boundary and region detection. We divide the related work in this section into two parts: work on road boundary detection and work on road region detection as they both require a different methodology for detection.

2.3.1 Boundary detection

Road boundary detection can be performed by utilizing a variety of sensors, including cameras, lidar, and mmwave radars [27, 39, 63]. Among these, camera-based methods have been the most widely adopted and utilized for the application. Classical computer vision-based methods utilize primitive properties of lanes such as gradients

and colors to recognize road and lane boundaries. The majority of these techniques employ a two-step pipeline. The first stage is responsible for edge detection while the second stage is responsible for line fitting. These methods are further classified based on the method of performing lane detection, such as in bird-eye view or in a regular front-facing image, depending on where the camera is mounted on the vehicle. Canny edge detection, Hough transform based lane detection, and sliding window search provide good results in lane and road boundary detection [34, 37, 59]. For continuous boundary detection, Kalman and particle filter works well in locating lane lines and estimating curvature of the roads [8, 33, 49]. However, traditional computer vision techniques fail to perform well in dynamic environments. Deep learning-based models are a viable alternative in such a case. Since the last few years, research has shifted toward developing models based on deep learning to detect road and lane boundaries.

Deep learning models can use a variety of different types of data to accomplish this task. Until now, various types of inputs have been used to train deep learning models for lane and road detection, including color images, waveforms, and lidar data [20, 24, 39, 63]. Kim et al. [25] propose a CNN based approach to first enhance the image using CNN which removes noise and non-required obstacles. The author used an Extreme Learning Machine (ELM) to train the model with the dataset. This learning algorithm reduces the training time dramatically so that the network can be better trained on large size datasets. Wang et al. [52] propose LaneNet, a deep neural network-based method that employs a pixel-by-pixel classification of lane edges. Ma et al. [32] propose a deep neural network consisting of a YOLO network for detecting and removing vehicles from images and a Convolutional Patch Network (CPN) for detecting and removing objects that are not on the road surface.

Multiple sensors can be used to leverage each sensor feature to make road bound-

ary detection robust. Ye et al. [58] project image as a pixel based waveform where the presence and continuation of the object lead to a spike in the waveform. They use CNN to detect and eliminate non-lane waveforms. Waveform threshold values are required to remove non-lane objects. If the threshold is set too high or too low, it will fail to detect lanes correctly, and the threshold will change in response to changing environmental conditions. This threshold was determined using CNN for different environmental conditions. Wulff et al. [53] propose a modified U-net-based FCN model that accepts data fusion from lidar and camera sensors. None of the above mentioned method demonstrate how their approach stands in a bad weather conditions and Xu et al. [57] demonstrates how lidar, RGB and infrared cameras fails in a bad weather conditions. Xu et al. [57] demonstrates how radar outperforms lidar in adverse weather conditions such as rain, smoke, and snow. Even though automotive radar is very effective in bad weather conditions, very little research has been conducted on radar-based localization due to the sparse resolution of radar sensors. They operate at a narrow band which helps them providing long range detection but the same characteristics introduces sparse resolution as well. This work primarily uses automotive radar to detect lane and road boundaries while mitigating the sparse resolution of automotive radars to make efficient detection in unfavorable weather conditions. We perform canny edge detection on a camera input and add the features from automotive radar to improve the resolution in all weather conditions, both sensors complement each other and provide a robust detection in all weather conditions.

2.3.2 Road region detection

Road region detection is primarily performed using semantic segmentation on camera images. Feng et al. [19] conducted a comprehensive survey of various deep learning methods for semantic segmentation. Long et al. [31] propose an image segmentation

technique based on a fully convolutional network. This was the first time FCN was used to perform end-to-end semantic segmentation. SegNet [5] is a deep fully convolutional neural network for semantic segmentation. It is made up of an encoder and decoder network, followed by a layer of pixel-by-pixel classification. The encoder assists in learning the low-level details of an image, while the decoder assists in mapping the low-level encoded details to the input resolution. ResNet (Residual Network) [21] is a segmentation model developed by a Microsoft research team as part of the ImageNet 2015 competition. It is made up of leftover blocks and skip connections. Ronneberger et al. [42] propose a U-Net architecture for segmenting biomedical images. It won the 2015 ISBI cell tracking challenge [35]. Even with fewer samples, the proposed architecture produces more precise results. Researchers prefer U-net when less data is available to train a neural network. The model substituted upsampling layers for pooling operators and provided feedback to each successive layer. These models discussed above are commonly used in segmentation tasks. However, pure vision-based approaches are insufficient for detecting road regions in all weather circumstances, as snow and rain obscure the sensor's view. Hence feedback from other sensors such as lidar and infrared are required to improve detection in bad weather conditions.

The performance of deep learning models can be improved by improving deep learning model structure, trained on the larger dataset, or by applying additional inputs. Models such as Segnet, U-net, MobileNet, FCN, and ResNet can be used interchangeably as encoder and decoder network pairs to improve system performance. Zhang et al. [62] propose a pyramid dilated convolution structure based on ResNet and U-net to segment medical images. To segment satellite images, [22] used U-net architecture with ResNet18 as an encoder. Abdollahi et al. [4] propose a model based on Segnet and U-net to segment high resolution aerial images.

Higher precision tasks, such as semantic and instance segmentation, which demands pixel-level classification, necessitate the use of complex deep learning models. Caltagirone et al. [11] used a camera and lidar fusion input with an encoder and decoder based FCN model to detect road regions. Lee et al. [26] propose a camera and lidar-based model to detect road regions using a spherical coordinate system to utilize three channels of camera image and height from lidar data as input to train an encoder-decoder based DL model. However, these state-of-the-art fusion techniques have been evaluated mainly in favorable weather conditions. Additionally, the use of lidar limits detection performance in adverse weather conditions (e.g., rain and snow) due to noise interference with laser rays. Thus, automotive radar is a better alternative to lidar for detecting conditions.

Automotive radars are relatively new in the market and there has been comparatively less research in using automotive radar for the perception stack of self-driving vehicles. Zhong et al. [64] demonstrated a technique for detecting pedestrians using a camera and mm-wave radar fusion. Lekic et al. [28] demonstrated a camera and automotive radar fusion technique based on a Generative adversarial Network (GAN). We take inspiration from the current research work and propose a deep learning model which utilizes the camera and automotive radar sensors as input, ResNet as encoder and U-net as a decoder for road region detection.

2.4 Summary

The autonomous vehicle system can be divided into two parts: mediated perception and end-to-end learning. Mediated perception treats each part of the system independently and end-to-end learning directly generates control signals from sensor inputs. Mediated perception uses modularity concepts that provide flexibility and

are easy to debug. This chapter provides an introduction of the modularity concept and a detailed description of each stack of the autonomous vehicle system and how they interact with each other. We also cover the basics of CNN and auto-encoders which lay the foundation of the proposed road detection approach in this thesis. A comprehensive review is also given on the current research state of road detection and discusses the background of autonomous vehicles. The next chapter provides a detailed methodology of the proposed work.

Chapter 3

Road Detection

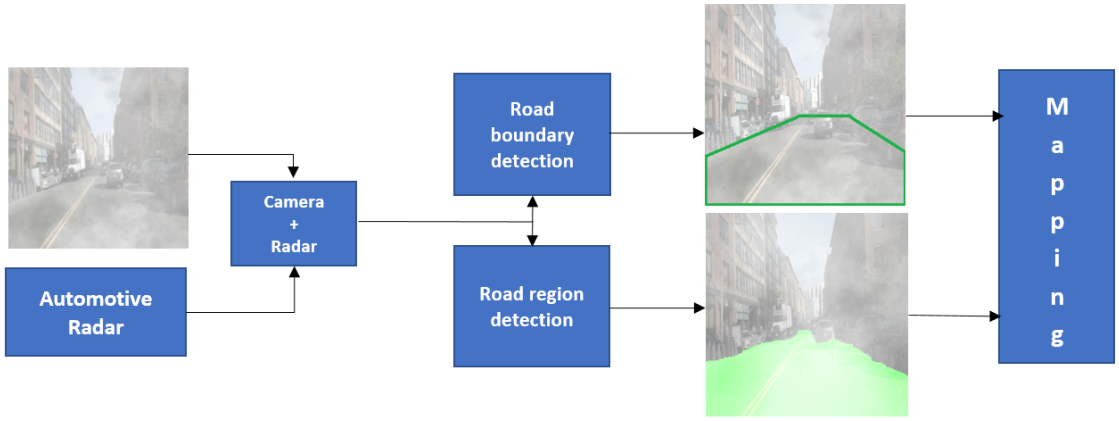

Figure 3.1: Road boundary and region detection framework

Figure 3.1 shows the overall architecture of our proposed work. We take input from the camera and automotive radar and use it to detect road boundaries and road regions. These detections can be used by the localization and mapping stack of autonomous vehicles to predict the future trajectory of the vehicle. This chapter discusses the proposed road boundary and region detection approaches in detail.

3.1 Road boundary detection

3.1.1 Framework overview

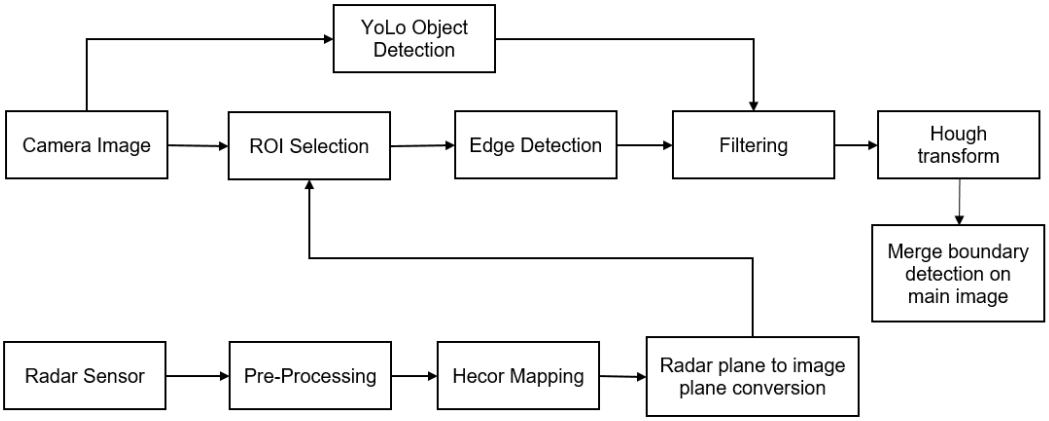

Figure 3.2: System architecture of road detection pipeline

Figure 3.2 shows the system architecture to detect road boundaries. First, canny edge detection detects the edges from color image then radar sensor detection gets pre-processed and projected to the edge image plane. YOLOv3 based object detection system removes the points that belong to vehicles. Hough transform scans for the connection points to detect the road boundaries. We project the detected road boundaries on a color image to showcase the detection. We validate our approach with a pure vision-based system that utilizes canny edge detection and hough transform to detect road boundaries [9].

The system receives two inputs: one from the camera and the other from the automotive radar. Each one of these inputs goes through a different processing pipeline. The camera pipeline includes performing canny edge detection, object detection to remove vehicles from the scene, and hough transform. The radar data processing includes pre-processing of pointcloud to remove noise, non-required radar detection,

and radar point projection on a camera image. The automotive radar provides a 3D pointcloud of an environment, which is an array with a group of 3D points detected by automotive radar in a frame. A camera provides three channel RGB images per frame, and they get converted to a single channel by performing edge detection on it. Canny edge provides edge spatial features of an image in one channel. We perform extrinsic calibration to map radar 3d pointcloud to a camera image frame. Extrinsic calibration maps each R(x,y,z) radar points to p(u,v) camera image pixel. We then use YOLOv3 to perform object detection on a camera image and to get the location of non-required objects such as vehicles and other obstacles. We remove the points that belong to them and apply hough transform on filtered points to get the road boundary. This section discusses each step in detail, and it covers sensor calibration, radar pre-processing, pointcloud mapping, radar to camera projection, camera edge detection, filters on pointcloud, and hough transform. We further discuss each step in detail.

Radar pre-processing

We use Texas Instruments AWR1843 radar which generates a 3D pointcloud with distance, angle, and intensity values for each point. These points must be pre-processed to perform the reliable mapping. For that, it includes configuring the radar, removing points from moving objects, and mapping pointcloud points. Non-static objects must be removed for accurate road boundary detection. The reason is that these moving objects (e.g., a car driving on the road) could be treated as a road boundary, resulting in inaccurate road mapping. We use an IMU to determine the test vehicle's velocity to remove moving objects. Automotive radar suffers due to sparse resolution. Radar points vanish in subsequent frames, even if the vehicle associated with the point is still present in the frame. We track pointcloud points across frames to track their

position and add them in a future position to increase the resolution of the radar sensor. We use odometry data to keep track of vehicle position and add the position displacement to each radar point to detect its future location.

Sensor calibration

Sensor fusion requires both intrinsic and extrinsic calibration. Intrinsic calibration is concerned with the internal calibration of a sensor. Intrinsic calibration generates the camera, distortion, rectification, and projection matrix in the case of a camera. When using multiple sensors, it is necessary to perform extrinsic calibration. Each sensor has its own axis, and they must transform on each other's axes. This can be accomplished through extrinsic calibration.

$$\begin{aligned} x_{corrected} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{corrected} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \tag{3.1}$$

$$\begin{aligned} x_{corrected} &= x + [2p_1 xy + p_2(r^2 + 2x^2))] \\ y_{corrected} &= y + [p_1(r^2 + 2y^2) + 2p_2 xy] \end{aligned} \tag{3.2}$$

Radial and tangential distortions are introduced by camera modules. Straight images get bent due to radial distortion, while Tangential distortion is primarily caused by an alignment error between the camera lens and the imaging plane. This creates the illusion that certain points are closer together than in the original image. Radial and tangential distortions are mathematically represented by equations 3.1 and 3.2, respectively. The equations contain five unknown parameters k1, k2, k3, p1, and p2 referred to as distortion coefficients. Additionally, a camera matrix must be computed because it is module dependent, and it only needs to be calculated once per module. We use a ROS based opencv solution to calibrate the distortion

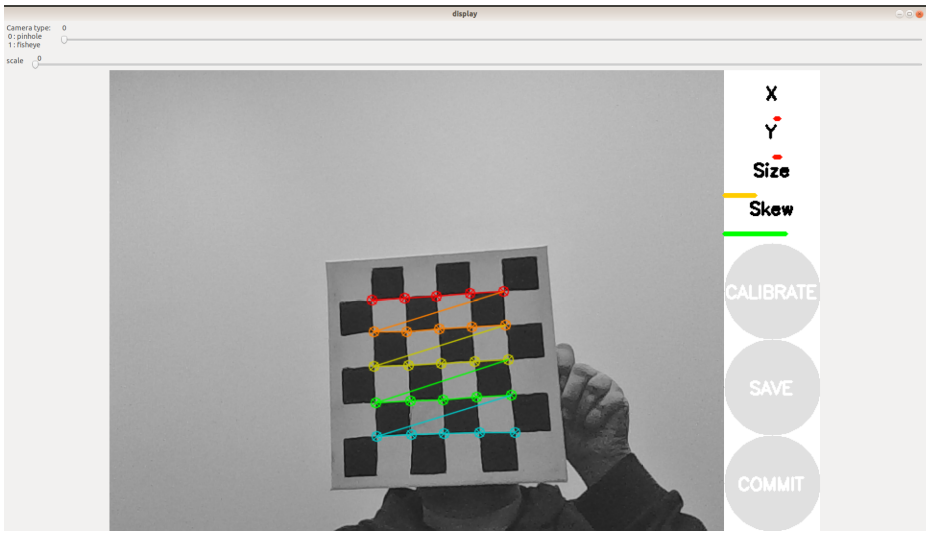

Figure 3.3: Chessboard based camera calibration

coefficients using a chessboard. We rotate the chessboard in all possible directions to record the different poses in x, y, z and skew directions as shown in Figure 3.3. Calibration contains the camera frame width and height, the camera matrix, the distortion coefficients, the rectification matrix, and the projection matrix.

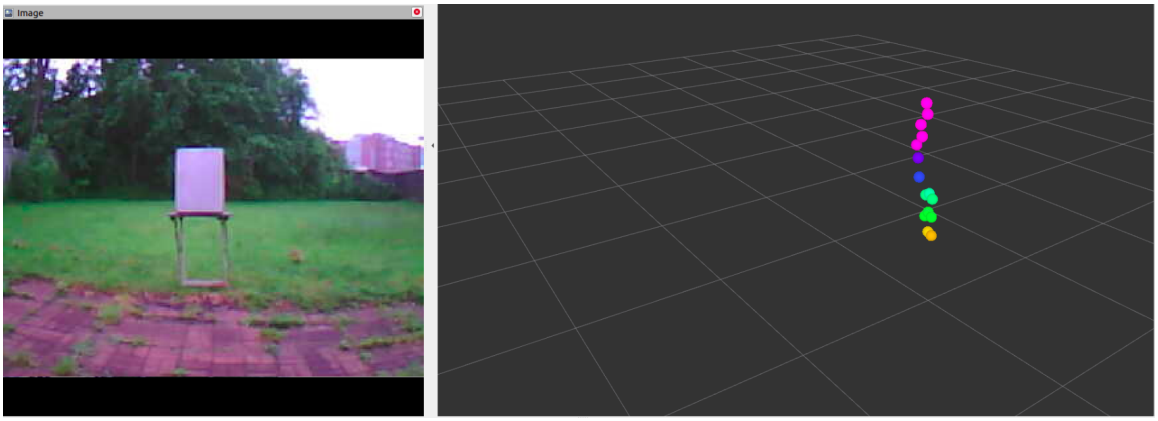

Figure 3.4: Camera and radar extrinsic calibration

$$\begin{bmatrix} u & v & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} R \\ T \end{bmatrix} K \tag{3.3}$$

$$\begin{aligned} \text{where:} \quad u, v &= \text{camera coordinates} \\ x, y, z &= \text{Radar point coordinates} \\ R &= \text{Rotation matrix} \\ T &= \text{Translation matrix} \\ K &= \text{Camera intrinsic matrix} \end{aligned}$$

The radar sensor generates points relative to real-world coordinates. As a result, they need to transform and map to the camera image. If (u,v) denotes camera pixels and (x,y,z) denotes radar points, projection mapping is required to convert (x,y,z) points in the radar pointcloud to (u,v) points. The main difficulty in performing extrinsic calibration is that the radar has a lower resolution than a lidar and provides a limited number of points for an object. However, on reflective materials such as metals, radar provides more points. Domhof et al. [17] published a comprehensive survey of open-source tools available for extrinsic calibration of various sensors. There has been very little research done on the fusion of camera and radar sensors, and no open-source tools are available to perform extrinsic calibration between the two. We provide a setup and method required to perform camera and radar sensor fusion. When the radar signal is reflected off metals, it generates strong signals. This characteristic generates additional calibration points and uses a metal rectangular shape after being inspired by various shapes created by [36, 40, 51]. Figure 3.4 illustrates the metal plate used in our experiments and the RVIZ tool's equivalent radar point detection. We record the radar point locations and equivalent camera pixels, and from the various observations taken from different locations, we calculate the translation and rotation vectors. The translation vector provides a linear transformation, which

converts the positions of objects to the camera's center. The rotation vector defines the angle at which the object rotates relative to the camera's position. Equation 3.3 provides a 3D radar to 2D camera image plane relation. An equivalent camera plane is obtained by matrix multiplication between radar points, camera intrinsic matrix, rotation matrix, and translation matrix.

Radar point tracking

We use the ROS environment to localize the vehicle and for that, we utilize hector mapping to localize the vehicle position using the odometry data. We use IMU, encoder, and radar sensor to localize the vehicle and utilize the octomap [23] to prepare a map of an environment. We use a prepared map, as shown in Figure 3.5 and convert it into pointcloud to get all the radar points present in the current frame. We mitigate the radar resolution problem and achieve high resolution per radar frame.

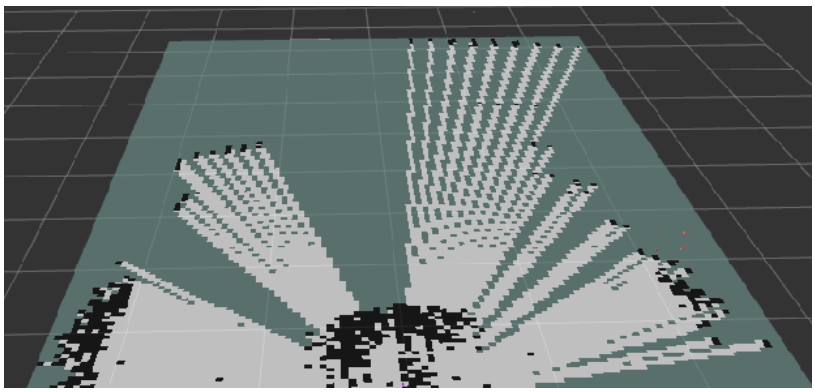

Figure 3.5: Radar pointcloud mapping

Edge detection

Edge information is one of the most important aspects of an image. It is used to observe the outline of an object, the relative positions of various objects, and other significant features. In this work, we use edge spatial features to detect road bound-

aries. The longest connected line is detected and extracted pixel points from detected road boundaries using canny edge detection and Hough transformation. Canny edge detection determines the gradient's intensity using a Gaussian derivative. It employs Gaussian filtering to smooth the image and minimize the noise effect in the image. Following that, it converts detected thick edges into thin edges by removing non-maximum gradient magnitude pixels. It eliminates non-candidate edges using a high and low threshold. Canny edge detection requires five inputs: an image, sigma, the width of the Gaussian filter, and low and high hysteresis threshold values. The sharper the image, the more Gaussian filtering is required to smooth it out. To accurately detect road boundary edges, we fine-tuned the canny edge detection parameters using camera samples. We filtered the image to investigate only the region of interest before applying canny edge detection. We created a polygon mask to denote the Region of Interest (RoI), which assists in filtering images that fall in that region.

Filters on camera and radar inputs

We propose several filters for radar sensors that are applicable to a wide variety of applications. While detecting road boundaries, it is possible to detect non-required objects such as cars. We use the following method to remove non required radar points.

Automotive radar detects points in the Y and Z axes with a range of 180 degrees (-90 to +90). It detects some points nearby that should not be detected, such as its mounting points and the vehicle's body. Figure 3.5 shows the radar detecting closer objects of sensor mount and vehicle body which falls within radar scanning. These points should be removed before converting them into image pixels as they may lead to incorrect road boundary detection. We create a position-based filter that removes all these non-required points based on their orientation. Figure 3.6 illustrates radar

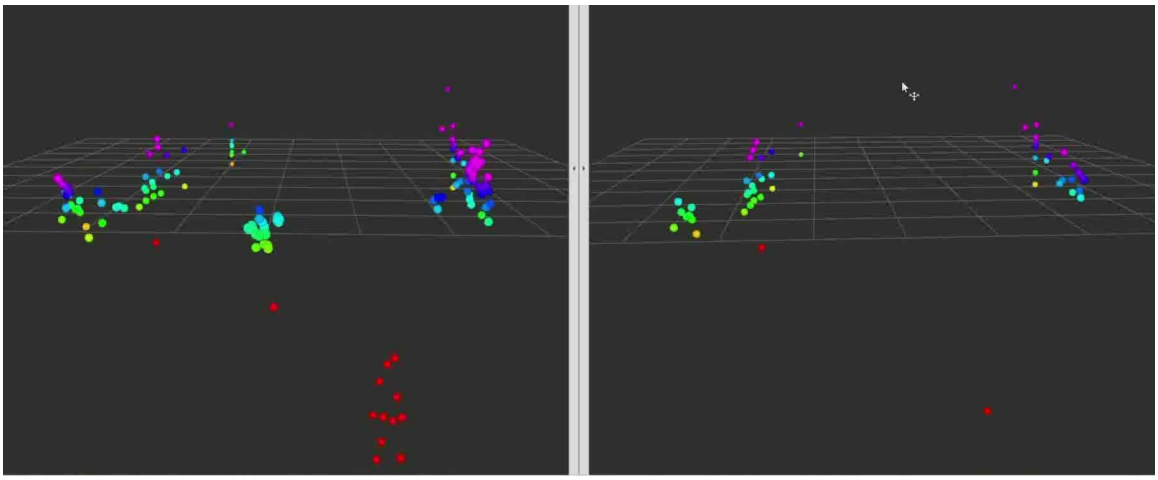

Figure 3.6: Position based filtering

points in a three-dimensional plane. As seen in the first image, radar points detect a portion of a vehicle and its mounting brackets, and this portion remains visible in all frames. In the second image, it has been omitted. This is how unnecessary radar points are eliminated by utilizing their absolute position to improve accuracy. Static clutter can be filtered out using TI mmwave radar's built-in functionality, but we want to keep the static points while removing moving ones, such as vehicles. To implement velocity-based filtering, we used the odometry data to get the velocity of the vehicle and use it to find out the points moving at the same velocity. These points are considered stationary, and all other points are discarded. This step provides the candidate points for road boundary detection.

However, there is a chance to miss some points for moving objects in the previous step due to the difference in velocity at turning points when the test vehicle slows down. In such a case, some moving vehicles on the road may be misclassified as static objects and result in incorrect road boundary detection. To tackle this issue, we use deep learning-based object detection to determine the position of other vehicles and remove them from the pointcloud matrix. For that, we utilize YOLO v3 with a darknet ROS package implemented by [7]. YOLO v3 works well in our case, but

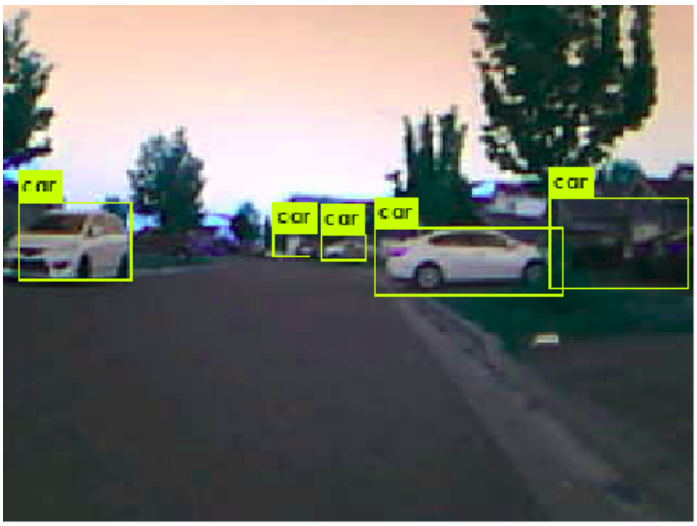

Figure 3.7: ROS based YOLOv3 Darknet object detection

a higher YOLO version or other deep learning methods can be utilized to improve object detection performance in general. YOLOv3 uses the COCO dataset for training which includes about 80 different classes. The classes we are interested in are: person, bicycle, car, motorcycle, bus, train, truck, traffic light, fire hydrant, stop sign, parking meter, and bench. We extract the pixel locations of objects belonging to these classes and delete the radar and camera points that fall within these regions. Figure 3.7 shows the object detection labels. YOLO is a probabilistic object detection model, and we use a higher probability score to filter out camera and radar points to avoid false detection. We use filtered points and project them on a black background with white color for further processing.

Boundary detection

The Hough transform detects road boundary edges that take the shape of a connected line (straight or polynomial). We tuned the Hough transform on a set of images to efficiently detect road boundaries. The Hough transform returns coordinates that are used to create a polygon that represents the road boundaries on an image. The

polygon coordinates are plotted on an image to show the road boundaries. The final output is depicted in Figure 4.6, which shows the road boundaries highlighted in green color.

3.2 Road region detection framework

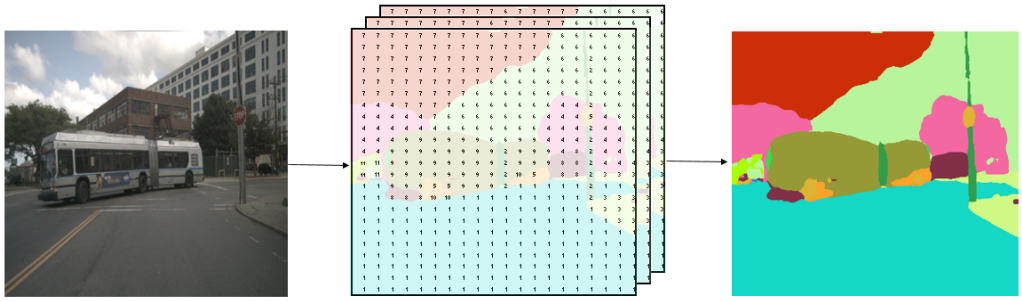

Figure 3.8: semantic segmentation of color image

Road region detection requires classifying each pixel of a road image to identify the boundaries, also referred to as image segmentation as shown in Figure 3.8. Image segmentation first identifies the boundaries of each object using gradients and marks them with different colors. This task requires pixel-level classification, which makes it harder compared to object detection. Figure 3.9 illustrates the proposed multi-modal deep learning model to detect the road regions.

3.2.1 Framework overview

The deep learning model uses the camera and automotive radar as input and uses an encoder and decoder based deep learning method. The encoder learns low-level features from the input using a series of convolutions, normalization, and activation layers. The decoder helps by learning high-level features. We use a U-net based decoder and ResNet50 based encoder for our work because U-net employs skip connec-

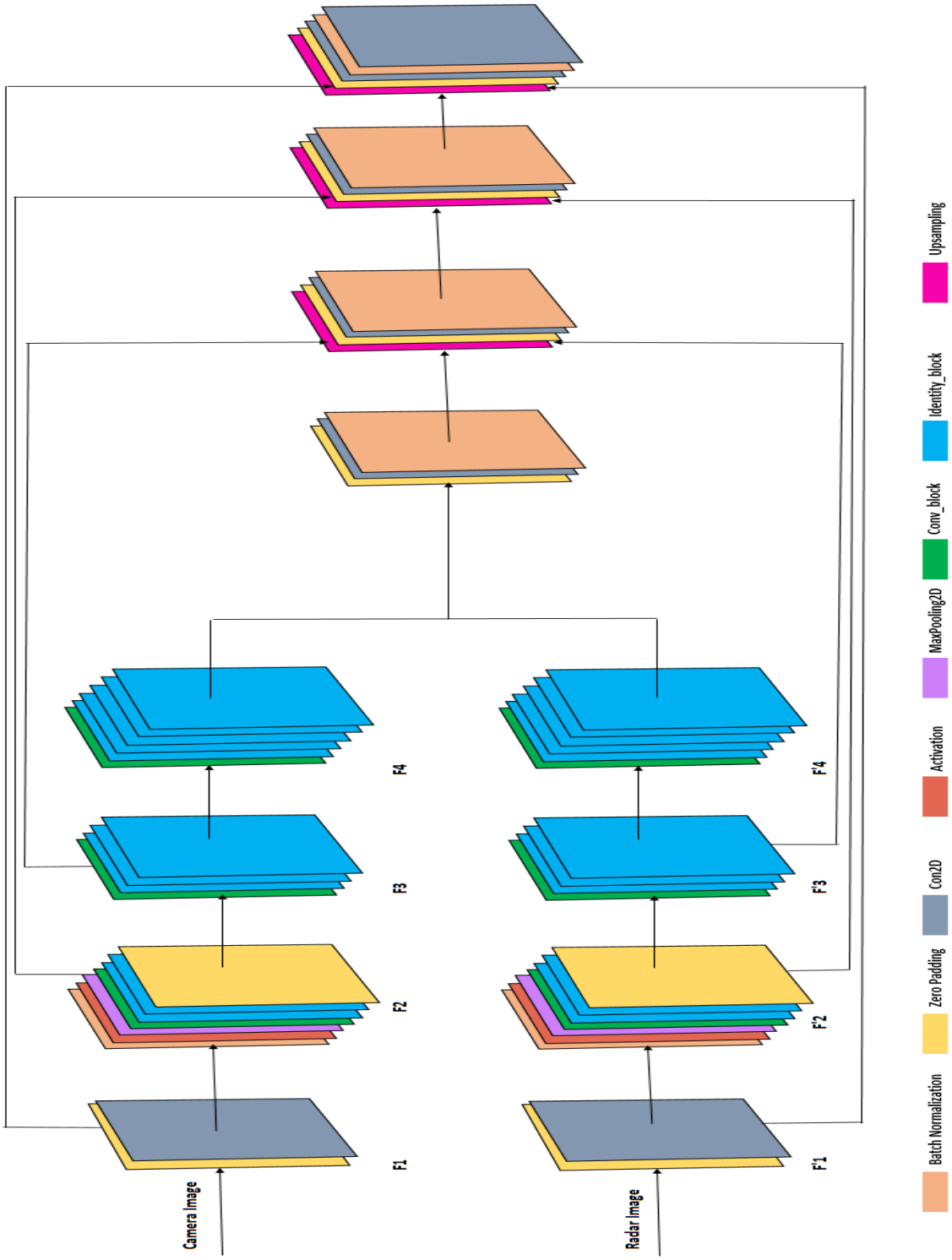

Figure 3.9: Camera and radar based deep learning segmentation model

tions between identical encoder and decoder layers. Therefore, it produces excellent results even when trained on a small dataset. While ResNet is made up of residual blocks connected via skip connections.

It has dual encoder channels, one for the camera and the other for the automotive radar sensor. These channels are combined using a concatenation layer. Both channels of the encoder layer use a ResNet encoder, while the decoder layer uses a U-net architecture. We use a total of four encoder levels and four decoder layers in the proposed architecture. Each level consists of convolutional and normalization layers. The encoder uses residual layers from a ResNet network. The residual and convolutional layers use a skip connection to consider the features learned at the input tensor. Each encoder layer is responsible for extracting features for its associated decoder layer. We use RELU as an activation layer.

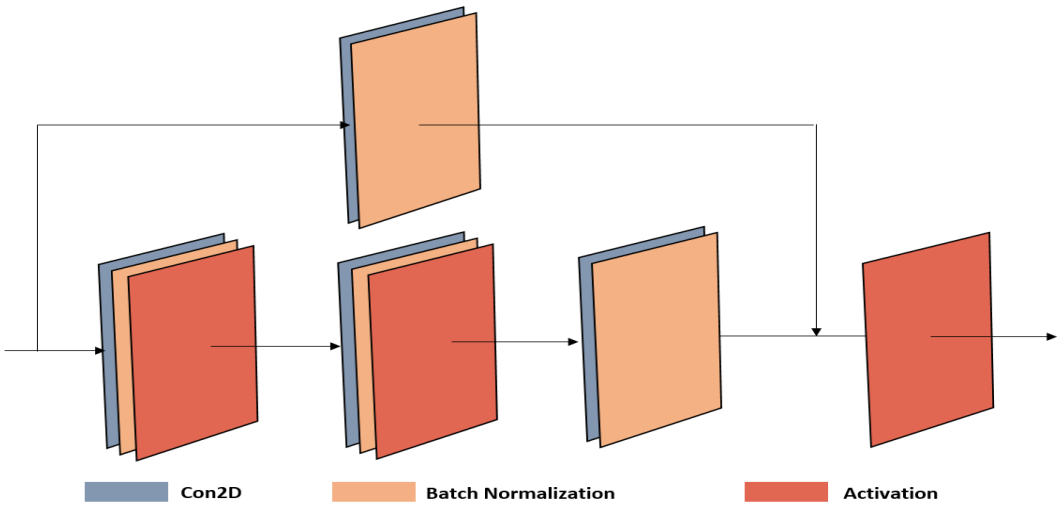

Figure 3.10: ResNet convolution block

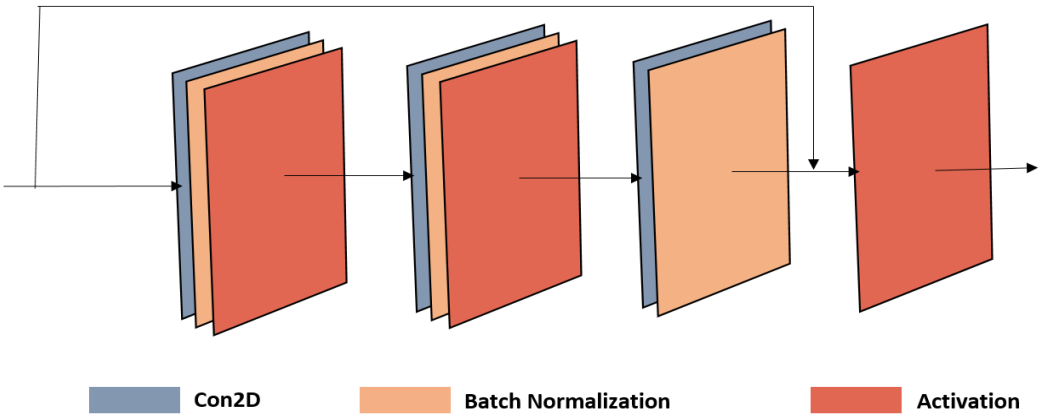

Figure 3.11: ResNet Residual or Identity block. Residual block composed of multiple layers of convolution and batch normalization

With the increase in layers in deep learning architectures, they suffer from vanishing gradient problems. It is a situation where a deep network is unable to propagate gradient information from the output end to previous layers near the input end. ResNet resolves the vanishing gradient problem by employing residual blocks. It fast-forwards the activation layer to a deeper layer in the network. Figure 3.11 illustrates the residual block referred to as an identity block. It skips connections to skip the layers, allowing information to be transmitted without attenuation and uses a 1x1 filter and RELU activation. The input tensor is denoted by x, and layer output is denoted by F(x). The residual block performs F(x) + x and the output of this is passed to the activation function. ResNet uses a convolution block that is similar to a residual block. However, instead of directly passing input at the end, it applies convolution to input to change the input dimension for matching it with the output dimension. Figure 3.10 illustrates a convolution block. Compared to the identity block, the convolution block contains the convolutional layer as the shortcut. A network consists of four layers containing a sequence of convolution and identity blocks. It accepts input sizes as multiples of 32 pixels (e.g., 32. 64, 128, and so on). Encoders com-

municate with decoders via skip connections. Concatenation layers join the feature maps together. We use pre-trained ResNet weights obtained from the ImageNet [15] dataset. The model provides a single channel, two-dimensional matrix containing the class value for each pixel location. Since we are using road and non-road regions as classes, each pixel in the 2D matrix will have a value of either 0 or 1. We generate a mask from the output matrix and apply it to the image to extract the road region.

Road region detection requires a high resolution sensor, while automotive radar has a comparatively lower resolution than lidar. One way to resolve this issue is to track the position of radar points across frames and plot them in subsequent frames by determining their future position. The vehicle's odometry determines the future location of radar points and incorporates them into the following radar frame. We use the frame's farthest detected point as a reference and then scan the preceding n frames to determine whether any point remains undetected within the range of the maximum detected point and vehicle position. If the point is not present in the current frame, we add it to the current frame to its new position. The number of combined frames is directly proportional to the vehicle's speed. At high vehicle speed, objects vanish from the frame faster compared to a slower speed. At slow speed, they may remain in relatively more frames. For our work, we have used n=3 frames, which means we track the points across three frames for better detection.

We use VLDNet as a baseline to compare the performance of our solution [16]. VLDNet outperforms the other state-of-the-art techniques [11, 13, 44, 45, 54–56] in different road situations such as urban, urban unmarked, urban marked, and multiple marked lanes. They use ResNet50 and U-net based encoder-decoder deep learning models. We reproduced the work and trained their model on our dataset to have a fair performance comparison. We have used the nuScens dataset to first compare the performance of our method compared to VLDNet on a normal weather condition

dataset. Then, we synthesize a new augmented dataset from nuScens to add bad weather conditions including, rain, snowfall, low visibility, to assess the performance of both models on unfavorable weather conditions.

3.3 Summary

This chapter introduced our proposed work of road region detection. We presented our framework of road detection and divided the problem into two sub-divisions: road boundary and road region detection. Each of these detections was carried out utilizing a distinct process. Road boundary detection uses edge spatial information for the detection whereas road region detection requires a much more detailed understanding of a scene and for that, we used a deep learning based approach. Both approaches employ an automotive radar along with the camera and other sensor stacks to provide all weather detection. We demonstrate how our point tracking approach provides a high-resolution radar input compared to the current methods. The next chapter discusses the evaluation strategy of the proposed work. It provides the experimental setup needed to evaluate the approach.

Chapter 4

Performance Evaluation and Discussions

This chapter provides the dataset collection process, the experimental setup for the evaluation, and the evaluation strategy to validate the proposed work. We use precision and recall as evaluation criteria for the road boundary detection whereas IoU for the road region detection. This chapter also describes the motivation behind selecting these criteria for performance evaluation and compares the results of our work with the state-of-the-art methods.

4.1 Experimental setup

4.1.1 Road boundary detection setup

We use 9Dof IMU M0 inertia sensors module, wheel encoder, openmv H7 monocular camera, TI mmwave 1843BOOST evaluation board, and Nvidia Jetson TX2 for the experiment of our proposed work. We mount the sensor package on a test vehicle to gather and process the inputs and use ROS noetic to handle the sensor data flow. We

Parameter	**value**
Frequency	77 GHz
Azimuth Resolution(deg)	15 + Elevation
Range Resolution(m)	0.044
Maximum unambiguous Range(m)	9.02
Maximum Radial Velocity(m/s)	1
Radial velocity resolution(m/s)	0.13
Frame Duration(msec)	100
Group peaking	Disable
Static clutter removal	Disable
Desired Radar Cross Section (sq. m)	0.5

Table 4.1: TI mmwave AWR1843BOOST radar evaluation board configuration parameters

process the inputs from different sensors using our own ROS package. The Nvidia Jetson TX2 is a small computer equipped with an Nvidia Pascal graphics processor with 8 GB of RAM. We perform camera and radar intrinsic and extrinsic calibration using the methods discussed in section 3.1. Openmv H7 is a small, low-power camera module. It has an onboard microcontroller and an SD card slot. The Openmv camera can capture 8-bit grayscale images or 16-bit RGB565 images at a frame rate of 75 frames per second when the resolution exceeds 320x240 and 150 frames per second when the resolution is lower than that.

Figure 4.1: Radar and camera sensor placement on test vehicle

Table 4.1 contains the tuned radar configuration parameters that enable the acquisition of radar points with less noise for the proposed work. It is necessary to efficiently tune the radar parameters to adjust object detection and the precision level. The maximum unambiguous range and velocity are used in this experiment. Generally, the radar detection precision decreases when the vehicle speed increases. We use the online visualizer platform provided by Texas Instruments to generate a radar configuration profile. On-board DSP processing powers up the radar sensor that enables features such as group peaking and static clutter removal. Group peaking identifies the range and Doppler direction points of a single object. Then, it merges or peaks them to reduce the number of radar points per frame. Both these features are not required in this work and thus have been disabled. Figure 4.1 shows the camera and radar sensor placement on the test vehicle. The sensors for the odometry were placed behind mmwave radar and the plate. An encoder was mounted on wheels to record the linear displacement, and we achieved angular displacement using the IMU sensor. In the next section, we discuss the performance evaluation and comparison with the pure vision-based method. The pure vision-based method [9] uses canny edge detection on camera frames and then detects the road boundaries using Hough transform. This approach works very well in daylight conditions, but we test this approach on a mixed environment comprising daytime and nighttime conditions.

4.1.2 Road region detection setup

This section discusses the steps involved in assessing the proposed method which includes preparing the dataset and training the model.

There are numerous publicly available datasets for camera and lidar, but very few for automotive radar. Table 4.2 contains a list of publicly available automotive radar datasets. There are a small number of datasets available that include a camera,

(a) Day time (b) Night time (c) Rainy weather (d) Temporary obstacles on road

Figure 4.2: Nuscenes dataset camera images samples of different weather conditions

Dataset	**Publisher**	**Features**
Nuscenes [10]	Motional	1000 driving scenes, 1.4 million radar sweeps, 6 camera, 5 radar, 1 lidar, GPS and IMU
Radar scenes [43]	Mercedes Benz	11 object categories
Radical [30]	University of Illinois	Camera, FMCW radar and IMU sensors
CARRADA [38]	Institut Polytechnique de Paris	camera and radar sensor with range-angle doppler annotations
RADDet [60]	University of Ottawa	Range-azimuth-doppler radar
RobotCar [6]	University of Oxford	2,40,000 radar scans, 2 lidars and 6 cameras

Table 4.2: publicly available Radar dataset

Figure 4.3: Labels generation of nuscenes dataset

radar, and lidar sensors as part of the sensor suite. Radical [30], CARRADA [38], and RADDet [60] are all datasets that contain range-azimuth-doppler radar data. The proposed work requires pointcloud data, more precisely from the radar sensor mounted in front of a vehicle. The front-facing radar sensor collects additional pointcloud data for the road and other objects in front of the testing vehicle.

The nuScenes [10] dataset is a large collection of camera, lidar, and radar images. The dataset contains more than 1000 driving scenes and 1.4 million radar sweeps. It provides data from a comprehensive sensor suite that includes six cameras, five radars, and one lidar. Figure 4.2 illustrates representative camera images from the nuScens dataset. Figure 4.2 illustrates the various weather conditions covered by the nuScenes dataset. It provides the day and nighttime conditions but does not provide inclement weather conditions (e.g. fog, rain, etc.). The Nuscenes dataset includes annotations for radar points but not road boundary objects. We manually labeled a subset of the dataset and the process of labeling is discussed in subsequent sections. Although we demonstrate our work using the nuScenes dataset, it can be applied to any automotive radar pointcloud data by resizing the camera and radar input to the input size of the deep learning model. We used 19 scenes from the nuScenes dataset that included a variety of weather and road conditions and combined them into a total of 583 color images for our dataset. We created a segmentation ground truth manually using a labelme [50]. Figure 4.3 illustrates the segmentation and labeling of an image. Following that, we created a one-channel ground truth image with the pixel value replaced with the class value. Because we only have two classes, the pixel values were limited to 0 or 1.

To assess the impact of bad weather conditions, we need images of the same locations in various weather conditions. The nuScenes dataset does not provide scenes that cover rain, snow, fog, or other bad weather conditions. At present, no automo-

Figure 4.4: Fog and rain augmentation in nuscenes dataset

tive radar related public dataset exists which contains bad weather conditions data. Further, the cost of collecting new datasets with these real-life conditions naturally embedded is beyond our budget and time constraints. To address this problem, we developed an augmentation technique to ingest rain and fog on the color images of the nuScenes dataset. The data augmentation expands the dataset artificially by generating more data from the existing dataset. Snow ingestion is performed using a classical computer vision based approach. We have used HLS (High-level synthesis) based lightness technique to whiten darker landscape areas such as roads and trees. We have used slope based lines to ingest rain in an image, reduced brightness and made it blurry to pretend rainy weather conditions.

Figure 4.4 illustrates the various levels of fog and rain present in images. The amount of snow or rain required in the dataset can be controlled. We have used medium severity weather conditions when augmenting the dataset. When introducing rain, we ignored the blurriness of the camera lenses as it was hard to achieve and out of scope for this work. However, in the real situation, we believe the performance of road detection will degrade even more for the state-of-art techniques. We prepare pointcloud image by plotting radar points on a mask image for a deep learning model

to use as an input. Figure 4.5 illustrates a color image and its corresponding radar frame. On a black mask image, radar pointcloud points are highlighted in red.

Figure 4.5: Color image and its equivalent radar frame

4.2 Evaluation

4.2.1 Road boundary detection evaluation

The performance of the proposed method is evaluated by examining how well the system detects road boundaries within the camera frames. A camera frame is visualized as a 2x2 matrix with each pixel containing either 1 or 0, where 1 is assigned to the road region and 0 to the non-road region. We use Precision and Recall as our evaluation metrics due to the imbalance between road segment pixels compared to the large background areas in the camera frames.

$$Accuracy = \frac{True\ negative + True\ positive}{Total\ Predictions} \tag{4.1}$$

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{4.2}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{4.3}$$

$$F = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4.4}$$

Precision is defined as the ratio between the true positives (i.e., successful detections) and a total number of positive detections (both true and false), while recall is the ratio between the number of True detections and the total number of detections. We examined each pixel of road regions and compared them to ground truth values to determine the system's precision and recall. To assess the proposed method, we collected data by driving a test vehicle in various conditions. There were a total of 20320 frames collected. Certain frames from various scenes evaluate the performance of the proposed method. We chose 240 frames from a total of 20320 captured frames. Following that, camera frames are manually labeled. Efficiency was determined by comparing the output to the ground truth. Our system's precision and recall were 82.19% and 88.23%, respectively. Vision based approach scores 72.19% precision and 66.23% recall. It is worth noticing that, in a dark environment vision-based approaches cannot detect road boundaries correctly causing false negative increases and providing less recall. However, precision depends on False positives, and vision based approaches have more False positives. We can say that the proposed method provides 22% more recall compared to pure vision-based approaches.

The road boundary detection is depicted in Figure 4.6. Green paint is used to denote the road's boundary. Fig. 4.7 illustrates the detection of roads in a dark environment. When the camera is unable to detect the road edges at night, radar

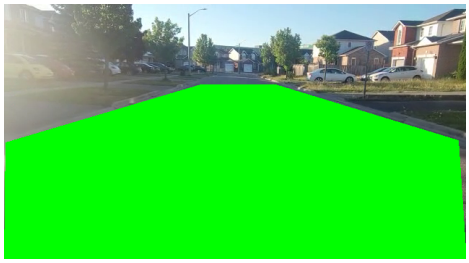

Figure 4.6: Camera and radar based road detection

Figure 4.7: Road boundary detection in dark environment

data assists in detecting the boundaries. While radar may not detect road boundaries accurately in curvature, it does assist drivers in estimating the road region in inclement weather conditions when no other sensor works.

It is obvious that in a dark environment, a camera cannot detect road boundaries alone. Thus the proposed system uses both a camera and radar sensors to detect road regions well in such adverse weather conditions. As a result, the proposed work contains several false positives. The primary reason for this is that in certain scenes, such as those with curved roads or those with a metal manhole cover near the road boundary, the metal manhole cover is detected by the radar and is treated as a road boundary. Such dynamic scenarios can be resolved in two ways. The first resolution is by inspecting their nature and incorporating them into the current project architecture. The second resolution, by detecting road regions using deep learning-based architectures.

4.2.2 Road region detection evaluation

$$Accuracy = \frac{Correct\ predictions}{All\ predictions} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

$$IoU = \frac{Are\ of\ overlap}{Area\ of\ union} \tag{4.2}$$

We use confusion matrix-based evaluation parameters to evaluate road region detection approaches. Equation 4.1 represents the accuracy metric. It is determined by the ratio of the correct predictions to total predictions. Road region detection is a binary class segmentation that is covered by non-road regions. The key disadvantage of using accuracy-based parameters is that even if one class outperforms others, the results still look good. If the background of an image covers 90% of it, it means a 90% accuracy of the system is achieved simply by classifying the background image only. To mitigate this issue, we are evaluating our system using the intersection over union (IoU) metric. It quantifies the interaction between predicted and ground truth segmentation. The IoU value is proportional to the performance of the object detection system. A perfect system will have identical interactions and unions between predicted and ground truth detection. It is derivable independently for each class. We use the mean and frequency weighted IoU in addition to class-level IoU. As the name suggests, mean IoU uses the mean of both classes to calculate the IoU. Frequency weighted IoU addresses the issue of class dominance further by utilizing weighted means and examining the frequency of class regions within the dataset.

Table 4.3 compares the performance of our method with VLDNet under various weather conditions. In favorable weather conditions, VLDNet provides good results and we achieve slight improvement over it. The performance of both deep learning models degrades when rain and fog are introduced in the nuScens dataset. VLDNet provides 91% and 76% respectively. Compared to VLDNet, our method performs

Condition	Model	Frequency Weighted IoU	Mean Iou	Class Wise IoU	
				Road Region	Non-road Region
Normal weather	VLDNet	0.9425	0.9352	0.9137	0.9560
	Proposed	0.9728	0.9627	0.9582	0.9793
Rainy	VLDNet	0.9168	0.9046	0.8952	0.9221
	Proposed	0.9485	0.9280	0.9273	0.9362
Fog	VLDNet	0.7646	0.7528	0.7384	0.7769
	Proposed	0.8873	0.8733	0.8320	0.9146

Table 4.3: Performance comparison of deep learning models on different conditions

better in inclement weather conditions and provides 94% and 88% respectively, which is a 12% improvement over VLDNet. It is worth noting that during rain, the models suffer less than they do with fog, as the augmented dataset does not include any fogging effect. In real life, rain adds fog to a camera lens and obstructs its view, which leads to further performance reduction of the VLDNet method. These results demonstrate that in adverse weather conditions, deep learning models can suffer. Even classical methods are incapable of detecting edges due to the blurring effect of fog and thus require additional input to comprehend the environment. The skip connection used in our model facilitates the transfer of encoder-learned features to decoders.

Figure 4.8 displays a number of images in three columns. The first column displays the original image, the second column shows the performance degradation in fog weather when using VLDNet, and the third column depicts the performance improvement using the proposed camera and automotive radar based model. We can see that when fog is introduced into images, it becomes difficult to detect road regions because the gradient on the edges smoothens and becomes difficult to differentiate. We can also see that detection on vehicles overlaps and tends to go outside of the road region. However, because of the additional input provided by radar frames, our pro-

posed approach detects the road region much more correctly. It is clear that when we use the proposed camera and radar-based deep learning model in inclement weather, the segmentation performance improves significantly compared to the state-of-the-art techniques.

4.3 Summary

This chapter illustrates the experimental setup and evaluation strategy for the proposed road detection approach. Currently, available public datasets for autonomous vehicles rarely include automobile radar sensors, and when they do, they typically only comprise scenarios in favorable weather circumstances. We collected a custom dataset for our road boundary detection approach by driving a test vehicle equipped with the experimental setup on a road. We also augmented the nuScens dataset with unfavorable weather conditions like fog and rain to evaluate our road region detection approach. We discussed the experimental setup required for this task in this chapter. We use precision and recall to evaluate the road boundary detection and IoU for road region detection evaluation. We did not employ the same evaluation technique for the second task, as it needs pixel-level classification, for which precision and recall are insufficient. We compared our approaches to state-of-the-art methodologies and demonstrated an improvement in performance under adverse weather circumstances.

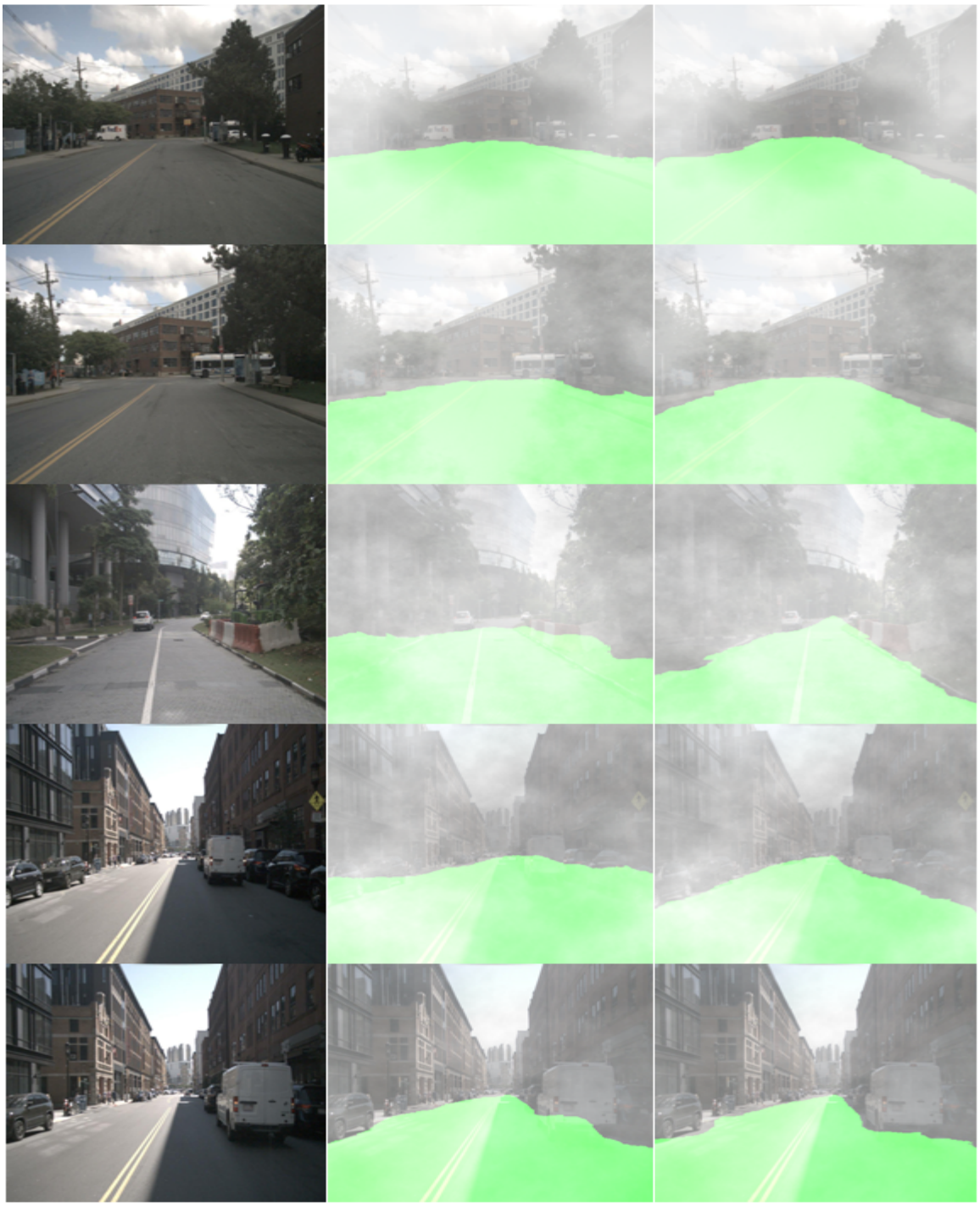

Figure 4.8: Road region detection using deep learning. Images from left to right shows original image, deep learning model performance on fog, proposed model performance

Chapter 5

Conclusion and future work

This thesis proposes a novel method for detecting road regions and boundaries. Road region detection was accomplished by incorporating a camera and radar-based multimodal encoder-decoder architecture into the image segmentation process. This architecture was then utilized to recognize road regions. With little modifications, this method may be used to identify any item in the image. To determine road boundaries, we employed a conventional computer vision technique. The input from YOLO object identification was used to eliminate vehicles from the image. In severe weather circumstances, we highlighted the importance of automobile radar above other sensors. We illustrated how other sensors fail to operate properly in adverse weather situations such as fog and rain. Additionally, this research proved the automotive radar's limited resolution and its inability to identify static objects. We offered a unique strategy for circumventing the constraints of mmwave radar and introduced novel techniques for improving perception in inclement weather. Deep learning-based road region detection outperforms state-of-the-art deep learning-based road region detection approaches significantly. Our research applies not only to autonomous vehicles, but manual drivers traveling in adverse weather situations such as rain, fog,

storm, or snow can benefit from it as well. Snowplow vehicles may use our suggested road border detection approach to accurately clean the road, resulting in a more comprehensive drivable road throughout the winter. While our techniques performed well in all weather situations, they may still fail in some dynamic circumstances such as road curvature roads covered with dense fog or rain. This may be mitigated by training a deep learning model on unsuccessful instances.

5.1 Conclusion

Road boundary detection is critical for vehicle perception because it defines the drivable road region that the mapping algorithm uses to calculate future trajectories. If road boundary detection fails in inclement weather conditions, trajectories cannot be generated, resulting in the failure of vehicle autonomy. This thesis proposes a novel approach that combines both radar and camera sensors to precisely detect road boundaries in unfavorable weather conditions such as snow, rain, and low visibility when other sensors render ineffective. We demonstrate that the proposed method outperforms vision-based methods by at least 20%. Further, the proposed approach can detect road boundaries in dark and low visibility conditions when all other approaches completely fail. In the future, we plan to implement more sophisticated computer vision algorithms to detect road boundaries, use YOLO object detection in conjunction with radar data to precisely detect roadside objects for more detailed mapping, and develop techniques to augment road boundary detection to visually assist. We demonstrated the effect of inclement weather on road region detection using existing techniques and how they can fail in such conditions. This Thesis presents an innovative method for detecting road regions based on deep learning and demonstrates its effectiveness on the famous nuScenes dataset. The proposed model employs an

automotive radar in conjunction with a camera to detect road segments. This sensor fusion aids in improving the overall performance in inclement weather. Continuous performance improvement of the camera and radar-based fusion techniques enables the removal of more expensive lidar from a sensor suite, thereby lowering the cost of manufacturing an autonomous vehicle. Performance evaluation shows that our approach outperforms the state-of-the-art techniques by 12% on the overall detection accuracy.

5.2 Future work

In the future, we intend to allow mmwave radar to recognize road lanes. To achieve this, the automobile radar resolution needs to be improved further, which is a difficult challenge at the moment. We are investigating the possibility of developing a cat-eye-equivalent pavement marker that can be placed inside the road and aid automotive radar in detecting lane lines. Additionally, we intend to use roadside units to collect additional feedback to improve a vehicle's localization. Additionally, we are investigating the use of holograms and augmented reality to supplement our road region and boundary detection output onto the road or car windshield to advise drivers without distracting them while driving.

Bibliography

[1] Defense Advanced Research Project Agency(DARPA). https://www.darpa.mil/about-us/timeline/-grand-challenge-for-autonomous-vehicles, 2021. Online; accessed 19 October 2021.

[2] Robot car "stanley" designed by Stanford Racing Team. https://cs.stanford.edu/group/roadrunner/stanley.html, 2021. Online; accessed 19 October 2021.

[3] Envision the Future — Velodyne Lidar. https://velodynelidar.com/, 2022. Online; accessed 07 February 2022.

[4] Abdollahi, A., Pradhan, B., and Alamri, A. M. An ensemble architecture of deep convolutional segnet and unet networks for building semantic segmentation from high-resolution aerial images. *Geocarto International* (2020), 1–16.

[5] Badrinarayanan, V., Kendall, A., and Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence 39*, 12 (2017), 2481–2495.

[6] Barnes, D., Gadd, M., Murcutt, P., Newman, P., and Posner, I. The oxford radar robotcar dataset: A radar extension to the oxford robotcar

dataset. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Paris, 2020).

[7] Bjelonic, M. YOLO ROS: Real-time object detection for ROS. `https://github.com/leggedrobotics/darknet_ros`, 2016–2018.

[8] Borkar, A., Hayes, M., and Smith, M. T. Robust lane detection and tracking with ransac and kalman filter. In *2009 16th IEEE International Conference on Image Processing (ICIP)* (2009), IEEE, pp. 3261–3264.

[9] Bounini, F., Gingras, D., Lapointe, V., and Pollart, H. Autonomous vehicle and real time road lanes detection and tracking. In *2015 IEEE Vehicle Power and Propulsion Conference (VPPC)* (2015), IEEE, pp. 1–6.

[10] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027* (2019).

[11] Caltagirone, L., Bellone, M., Svensson, L., and Wahde, M. Lidar–camera fusion for road detection using fully convolutional neural networks. *Robotics and Autonomous Systems 111* (2019), 125–131.

[12] Chen, C., Seff, A., Kornhauser, A., and Xiao, J. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE international conference on computer vision* (2015), pp. 2722–2730.

[13] Chen, L., Yang, J., and Kong, H. Lidar-histogram for fast road and obstacle detection. In *2017 IEEE international conference on robotics and automation (ICRA)* (2017), IEEE, pp. 1343–1348.

[14] Chen, X., Ma, H., Wan, J., Li, B., and Xia, T. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (2017), pp. 1907–1915.

[15] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (2009), Ieee, pp. 248–255.

[16] Dewangan, D. K., Sahu, S. P., Sairam, B., and Agrawal, A. Vldnet: Vision-based lane region detection network for intelligent vehicle system using semantic segmentation. *Computing* (2021), 1–26.

[17] Domhof, J., Kooij, J. F., and Gavrila, D. M. A joint extrinsic calibration tool for radar, camera and lidar. *IEEE Transactions on Intelligent Vehicles* (2021).

[18] Dou, J., Xue, J., and Fang, J. Seg-voxelnet for 3d vehicle detection from rgb and lidar data. In *2019 International Conference on Robotics and Automation (ICRA)* (2019), IEEE, pp. 4362–4368.

[19] Feng, D., Haase-Schütz, C., Rosenbaum, L., Hertlein, H., Glaeser, C., Timm, F., Wiesbeck, W., and Dietmayer, K. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems 22*, 3 (2020), 1341–1360.

[20] Feng, Z., Zhang, S., Kunert, M., and Wiesbeck, W. Applying neural networks with a high-resolution automotive radar for lane detection. In *AmE 2019-Automotive meets Electronics; 10th GMM-Symposium* (2019), VDE, pp. 1–6.

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

[22] Hordiiuk, D., Oliinyk, I., Hnatushenko, V., and Maksymov, K. Semantic segmentation for ships detection from satellite imagery. In *2019 IEEE 39th International Conference on Electronics and Nanotechnology (ELNANO)* (2019), IEEE, pp. 454–457.

[23] Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., and Burgard, W. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots* (2013). Software available at `http://octomap.github.com`.

[24] Hou, Y., Ma, Z., Liu, C., and Loy, C. C. Learning lightweight lane detection cnns by self attention distillation. In *Proceedings of the IEEE International Conference on Computer Vision* (2019), pp. 1013–1021.

[25] Kim, J., Kim, J., Jang, G.-J., and Lee, M. Fast learning method for convolutional neural networks using extreme learning machine and its application to lane detection. *Neural Networks 87* (2017), 109–121.

[26] Lee, J.-S., and Park, T.-H. Fast road detection by cnn-based camera-lidar fusion and spherical coordinate transformation. *IEEE Transactions on Intelligent Transportation Systems* (2020).

[27] Lee, S., Xie, K., Ngoduy, D., and Keyvan-Ekbatani, M. An advanced deep learning approach to real-time estimation of lane-based queue lengths at a signalized junction. *Transportation research part C: emerging technologies 109* (2019), 117–136.

[28] Lekic, V., and Babic, Z. Automotive radar and camera fusion using generative adversarial networks. *Computer Vision and Image Understanding 184* (2019), 1–8.

[29] Lian, Y., Feng, T., and Zhou, J. A dense pointnet++ architecture for 3d point cloud semantic segmentation. In *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium* (2019), IEEE, pp. 5061–5064.

[30] Lim, T.-Y., Markowitz, S. A., and Do, M. N. Radical: A synchronized fmcw radar, depth, imu and rgb camera data dataset with low-level fmcw radar signals. *IEEE Journal of Selected Topics in Signal Processing 15*, 4 (2021), 941–953.

[31] Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 3431–3440.

[32] Ma, Y., Havyarimana, V., Bai, J., and Xiao, Z. Vision-based lane detection and lane-marking model inference: A three-step deep learning approach. In *2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)* (2018), IEEE, pp. 183–190.

[33] Mammeri, A., Boukerche, A., and Lu, G. Lane detection and tracking system based on the mser algorithm, hough transform and kalman filter. In *Proceedings of the 17th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems* (2014), pp. 259–266.

[34] Mammeri, A., Boukerche, A., and Tang, Z. A real-time lane marking localization, tracking and communication system. *Computer Communications 73* (2016), 132–143.

[35] Maška, M., Ulman, V., Svoboda, D., Matula, P., Matula, P., Ederra, C., Urbiola, A., España, T., Venkatesan, S., Balak, D. M., et al. A benchmark for comparison of cell tracking algorithms. *Bioinformatics 30*, 11 (2014), 1609–1617.

[36] Natour, G. E., Ait-Aider, O., Rouveure, R., Berry, F., and Faure, P. Toward 3d reconstruction of outdoor scenes using an mmw radar and a monocular vision sensor. *Sensors 15*, 10 (2015), 25937–25967.

[37] Niu, J., Lu, J., Xu, M., Lv, P., and Zhao, X. Robust lane detection using two-stage feature extraction with curve fitting. *Pattern Recognition 59* (2016), 225–233.

[38] Ouaknine, A., Newson, A., Rebut, J., Tupin, F., and Pérez, P. Carrada dataset: camera and automotive radar with range-angle-doppler annotations. In *2020 25th International Conference on Pattern Recognition (ICPR)* (2021), IEEE, pp. 5068–5075.

[39] Perng, J. W., Hsu, Y. W., Yang, Y. Z., Chen, C. Y., and Yin, T. K. Development of an embedded road boundary detection system based on deep learning. *Image and Vision Computing* (2020), 103935.

[40] Peršić, J., Marković, I., and Petrović, I. Extrinsic 6dof calibration of a radar–lidar–camera system enhanced by radar cross section estimates evaluation. *Robotics and Autonomous Systems 114* (2019), 217–230.

[41] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 652–660.

[42] Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (2015), Springer, pp. 234–241.

[43] Schumann, O., Hahn, M., Scheiner, N., Weishaupt, F., Tilly, J., Dickmann, J., and Wöhler, C. RadarScenes: A Real-World Radar Point Cloud Data Set for Automotive Applications, mar 2021.

[44] Shinzato, P. Y. Estimation of obstacles and road area with sparse 3d points. *Institute of Mathematics and Computer Science (ICMC)/University of Sao Paulo (USP)* (2015).

[45] Shinzato, P. Y., Wolf, D. F., and Stiller, C. Road terrain detection: Avoiding common obstacle detection assumptions using sensor fusion. In *2014 IEEE Intelligent Vehicles Symposium Proceedings* (2014), IEEE, pp. 687–692.

[46] Sindagi, V. A., Zhou, Y., and Tuzel, O. Mvx-net: Multimodal voxelnet for 3d object detection. In *2019 International Conference on Robotics and Automation (ICRA)* (2019), IEEE, pp. 7276–7282.

[47] Stanford Artificial Intelligence Laboratory et al. Robotic operating system. `https://www.ros.org`.

[48] Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision* (2015), pp. 945–953.

[49] Suttorp, T., and Bucher, T. Learning of kalman filter parameters for lane detection. In *2006 IEEE Intelligent Vehicles Symposium* (2006), IEEE, pp. 552–557.

[50] Wada, K. labelme: Image polygonal annotation with python. https://github.com/wkentaro/labelme, 2018.

[51] Wang, T., Zheng, N., Xin, J., and Ma, Z. Integrating millimeter wave radar with a monocular vision sensor for on-road obstacle detection applications. *Sensors 11*, 9 (2011), 8992–9008.

[52] Wang, Z., Ren, W., and Qiu, Q. Lanenet: Real-time lane detection networks for autonomous driving. *arXiv preprint arXiv:1807.01726* (2018).

[53] Wulff, F., Schäufele, B., Sawade, O., Becker, D., Henke, B., and Radusch, I. Early fusion of camera and lidar for robust road detection based on u-net fcn. In *2018 IEEE Intelligent Vehicles Symposium (IV)* (2018), IEEE, pp. 1426–1431.

[54] Xiao, L., Dai, B., Liu, D., Hu, T., and Wu, T. Crf based road detection with multi-sensor fusion. In *2015 IEEE Intelligent Vehicles Symposium (IV)* (2015), IEEE, pp. 192–198.

[55] Xiao, L., Wang, R., Dai, B., Fang, Y., Liu, D., and Wu, T. Hybrid conditional random field based camera-lidar fusion for road detection. *Information Sciences 432* (2018), 543–558.

[56] Xu, F., Chen, L., Lou, J., and Ren, M. A real-time road detection method based on reorganized lidar data. *PloS one 14*, 4 (2019), e0215159.

[57] Xu, F., Wang, H., Hu, B., and Ren, M. Road boundaries detection based on modified occupancy grid map using millimeter-wave radar. *Mobile Networks and Applications* (2019), 1–8.

[58] Ye, Y. Y., Hao, X. L., and Chen, H. J. Lane detection method based on lane structural analysis and cnns. *IET Intelligent Transport Systems 12*, 6 (2018), 513–520.

[59] Yi, S.-C., Chen, Y.-C., and Chang, C.-H. A lane detection approach based on intelligent vision. *Computers & Electrical Engineering 42* (2015), 23–29.

[60] Zhang, A., Nowruzi, F. E., and Laganiere, R. Raddet: Range-azimuth-doppler based radar object detection for dynamic road users. *arXiv preprint arXiv:2105.00363* (2021).

[61] Zhang, J., Zhao, X., Chen, Z., and Lu, Z. A review of deep learning-based semantic segmentation for point cloud. *IEEE Access 7* (2019), 179118–179133.

[62] Zhang, Q., Cui, Z., Niu, X., Geng, S., and Qiao, Y. Image segmentation with pyramid dilated convolution based on resnet and u-net. In *International conference on neural information processing* (2017), Springer, pp. 364–372.

[63] Zhang, S., Koubia, A. E., and Mohammed, K. A. K. Traffic lane detection using fcn. *arXiv preprint arXiv:2004.08977* (2020).

[64] Zhong, Z., Liu, S., Mathew, M., and Dubey, A. Camera radar fusion for increased reliability in adas applications. *Electronic Imaging 2018*, 17 (2018), 258–1.

[65] Zhou, Y., and Tuzel, O. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 4490–4499.