Operational Design Domain Monitoring and Augmentation for Autonomous Driving

by

Chen Sun

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2022

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner: **Huijing Zhao**
Professor, Dept. of Electronics Engineering & Computer Science,
Peking University

Supervisor(s): **Amir Khajepour**
Professor, Dept. of Mechanical & Mechatronics Engineering,
University of Waterloo

Internal Member: **Soo Jeon**
Professor, Dept. of Mechanical & Mechatronics Engineering,
University of Waterloo

Internal Member: **Jan Huissoon**
Professor, Dept. of Mechanical & Mechatronics Engineering,
University of Waterloo

Internal-External Member: **Jun Liu**
Professor, Dept. of Applied Math,
University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Chen Sun was the sole author of Chapters 1, 4, 5, and 6 which were written under the supervision of Prof. Amir Khajepour and were not published. This thesis consists of part of one manuscript written for publication.

Research presented in Chapters 2 and 3:
C. Sun, Z. Deng, W. Chu, S. Li and D. Cao, "Acclimatizing the Operational Design Domain for Autonomous Driving Systems," in IEEE Intelligent Transportation Systems Magazine, vol. 14, no. 2, pp. 10-24, March-April 2022, doi: 10.1109/MITS.2021.3070651.

As lead author of this chapter, I was responsible for contributing to conceptualizing study design, carrying out data collection and analysis, and drafting and submitting manuscripts. My coauthors provided guidance during each step of the research and provided feedback on draft manuscripts.

Abstract

Recent technological advances in Autonomous driving systems (ADS) show promise in increasing traffic safety. One of the critical challenges in developing ADS with higher levels of driving automation is to derive safety requirements for its components and monitor the system performance to ensure safe operation. The Operational Design Domain (ODD) for ADS confines ADS safety in the context of its function.

The ODD represents the operating environment within which an ADS operates and satisfies the safety requirements. To reach a state of "informed safety", the system's ODD must be explored and well-tested in the development phase. At the same time, the ADS must monitor the operating conditions and corresponding risks in real-time. Existing research and technologies do not directly express the ODD quantitatively, nor have a general monitoring strategy designed to handle the learning-based system, which is heavily used in the recent ADS technologies. The safety-critical nature of the ADS requires us to provide a thorough validation, continual improvement, and safety monitoring of these data-driven dependent modules. In this dissertation, the ODD extraction, augmentation, and real-time monitoring for the ADS with machine learning components are investigated.

There are three major components for the ODD of the ADS with machine learning components for general safety issues. In the first part, we propose a framework to systematically specify and extract the ODD, including the environment modeling and formal and quantitative safety specifications for models with machine learning parts. An empirical demonstration of the ODD extraction process based on predefined specifications is presented with the proposed environment model. In the second part, the ODD augmentation in the development phase is modeled as an iterative engineering problem solved by robust learning to handle the unseen future natural variations. The vision tasks in ADS are the major focus here and the effectiveness of model-based robustness training are demonstrated, which can improve model performance and the application of extracting the edge cases during the iterative process. Furthermore, the testing procedure provides us valuable priors on the probability of failures condition in the known testing environment, which can be further utilized in the real-time monitoring procedure. Finally, a solution for online ODD monitoring that utilizes the knowledge from the offline validation process as Bayesian graphical models to improve safety warning accuracy is provided. While the algorithms and techniques proposed in this dissertation can be applied to many safety-critical robotic systems with machine learning components, in this dissertation, the main focus lies on the implications for autonomous driving.

Acknowledgements

I would like to first and foremost thank my advisor Prof. Amir Khajepour. His patience and ability to see the positive in every bit of academia and elevate ideas to solve real engineering problems have motivated me throughout the last four years. I have also been fortunate to have Prof. Amir's invaluable guidance throughout my Ph.D. and the hands-on projects I can take. I learned about challenging real-life problems and how to approach them and solve them because of the extensive depth and breadth of his knowledge and vision of vehicle systems. The close collaboration with the industry provided by Prof. Amir also broadened my vision in the vehicle safety and risk monitoring area.

I also owe a great deal to Prof. Dongpu Cao, my co-supervisor, for this endeavor's first half. Over the short time that Prof. Dongpu has been at Waterloo, he has given me a unique viewpoint on research and life. Many thanks for his constant help, positive remarks, and guidance to help me step on the research in autonomous driving and safety monitoring area. I am also thankful to the members of my defense committee: Prof. Jan Huissoon, Prof. Soo Jeon, and Prof. Jun Liu, who have provided helpful guidance to shape my research direction from the comprehensive exam.

I want to thank my friends at the CogDrive Lab and Mechatronic Vehicle Systems (MVS) Lab. During my starting research experience at CogDrive Lab, I enjoyed socializing and studying with Dr. Huilong Yu, Teng Liu, Zejian Deng, JMU Vianney, Lang Su, Xingxin Chen, and Yaodong Cui. I cannot name all the friends I spent time with during lunchtime, teatimes, and dinnertime at the E-3 building, but thank you, Keqi Shu, Zemin Sun, Jinwei Zhang, and Wenbo Li. Special thanks to Neel, Ben, Pouya, and Ruihe for the great help when I first joined the MVS lab and the collaboration on the shuttle bus related projects. Also, thank you, Minghao Ning, for the careful instructions on the lane identification algorithms. Many thanks to Prof. Mohammad Pirani and Reza for their help in reviewing this dissertation.

The final thanks go to my family, that always supported and encouraged me to accomplish this goal. I am very thankful to my parents, Changui and Benqin, for all their sacrifices, love, and support.

Table of Contents

List of Figures xi

List of Tables xvii

1 Introduction 1

1.1 Background and Motivation 1

1.2 Relevance 3

1.2.1 Driving Automation 3

1.2.2 Machine Learning in ADS 4

1.2.3 Safety Issues in ADS 5

1.3 Challenges 7

1.3.1 Operational Design Domain Extraction of High-level ADS 7

1.3.2 Measurable Safety 8

1.3.3 Real-time ODD Monitoring 9

1.4 Research Scope and Contributions 10

1.5 Chapter Outlines 12

2 Literature Review 14

2.1 Recent ODD Related Standards 14

2.2 Scene Understanding and Situation Awareness 16

2.2.1 Background Representation . . . 16
2.2.2 Event Detection . . . 18
2.2.3 Intention Prediction . . . 19
2.2.4 Remarks . . . 19
2.3 Safety Validation and Related Metrics . . . 20
2.3.1 Validation and Falsification of Learning-based Components of ADS 20
2.3.2 Road Testing and Progressive Metrics . . . 21
2.3.3 Validation Benchmarks . . . 22
2.3.4 Motion Metrics . . . 24
2.3.5 Statistical Metrics . . . 26
2.3.6 Remarks . . . 26
2.4 Encoding and Improving the ODD . . . 27
2.4.1 ODD Encoding . . . 27
2.4.2 Robust Training . . . 28
2.5 Summary . . . 29

3 ODD Encoding and Extraction **30**
3.1 Framework for ODD Acclimatization . . . 30
3.1.1 Scenario Database . . . 31
3.1.2 Environment Modeling and States . . . 33
3.1.3 Formal and Quantitative Specifications . . . 35
3.2 Case studies . . . 40
3.2.1 ADS Policies . . . 41
3.2.2 Task Scenario & Environmental Conditions . . . 42
3.2.3 Evaluation . . . 43
3.3 Discussion and Future Work . . . 45

4 ODD Augmentation **48**
4.1 Problem Description and Preliminaries . . . 48
4.2 Methodology . . . 49
4.2.1 Transfer Learning . . . 49
4.2.2 Robust Learning to Natural Perturbation . . . 51
4.2.3 Perturbation Models . . . 55
4.2.4 Robust Training for Model-based Perturbation . . . 59
4.3 Experiment . . . 62
4.3.1 Datasets and Metrics . . . 62
4.3.2 Characteristics of Neighbour Accuracy and Robustness . . . 67
4.3.3 Sensitivity to Natural Perturbations . . . 73
4.3.4 Evaluation of Robust Training . . . 77
4.4 Discussion . . . 82

5 ODD Monitoring **84**
5.1 Related Works . . . 84
5.2 Problem Formulation and Preliminaries . . . 85
5.3 ODD Monitoring without Map Matching . . . 85
5.3.1 Probabilistic Model for Causal Relation . . . 86
5.3.2 Example . . . 87
5.3.3 Discussion . . . 91
5.4 ODD Monitoring with HD Map . . . 92
5.4.1 General Architecture . . . 92
5.4.2 Lane Marking Measurements . . . 94
5.4.3 Landmark Measurement . . . 96
5.4.4 Out of ODD Classification . . . 97
5.5 Experiment on Virtual Environment . . . 100

5.5.1 Simulation Setup . . . 100
5.5.2 Evaluation Metrics . . . 102
5.5.3 Qualitative Analysis . . . 106
5.5.4 Classifier Performance Analysis . . . 115
5.6 Experiment on Real Operation Data . . . 119
5.6.1 Data Roll-outs Configuration and Perturbation . . . 119
5.6.2 Qualitative Analysis . . . 121
5.6.3 Classifier Evaluation . . . 122
5.7 Discussion . . . 126

6 Summary and Outlook **127**
6.1 Summary and Contributions . . . 127
6.1.1 Contributions . . . 127
6.2 Outlook . . . 129

Letter of Copyright Permission **130**

References **131**

APPENDICES **161**

A **162**
A.1 Automotive standards on Lane Departure Warning . . . 162
A.2 Localization Requirements for Autonomous Vehicles . . . 164
A.3 Perception Requirements for Autonomous Vehicles . . . 165
A.4 Training Details for Vision Tasks . . . 166
A.5 Lane Estimation Software Setup . . . 167
A.6 Shuttle Bus Platform and Sensor Setup . . . 168

List of Figures

1.1 A holistic view of safety in modern ADAS system (figure adapted from [1]). The advanced ADAS monitors the condition of the traffic and driver and interacts with the driver by sending warnings and assisting with dynamic control and possible avoidance [2], including the crash mitigation and post-crash modules, to enhance safety. 6

1.2 An overview of research scope. 10

2.1 Recent popular synthetic datasets and open-sourced driving Simulator. Examples from SYNTHIA [3], CARLA [4], Virtual KITTI [5], and AirSim [6]. 24

3.1 Framework of the proposed approach. 31

3.2 Environmental modeling and the corresponding quantification, redundancy, and monitoring. SQL: Structured Query Language; APIs: application programming interfaces; NDS: Navigation Data Standard. 34

3.3 The flowchart of the coverage-driven verification. 36

3.4 Two network architectures for imitation learning based autopilot module tested in the experiment. The E2E CIL Agent (left) takes an RGB image as input and directly maps to action based on the high-level command. The CAL Agent (right) maps the image to explicit states, and the action is decided explicitly based on the perceived affordances. 41

3.5 The testing scenarios used in this study. Left: driving straight following the lane at single-lane tertiary. Right: one-turn road following task based on the directional command at T-junction. The ego-vehicle is initiated at the spawning point (SP), and the mission is to driving towards the corresponding ending point (EP). 42

3.6 The RGB image inputs under various environmental conditions. The left three columns correspond to the daytime (SunAltitude = 60), whereas the right two columns are sunset (SunAltitude = 0.5, SunAzimuth=180) and night (SunAltitude = -90). The cloudy, rainy, and foggy weather are demonstrated in rows with three discrete conditions, light, medium, and heavy. . 43

4.1 Models of natural perturbation. The model of natural perturbation creates a variation of the input data based on the semantic nuisance parameters, $x' = G(x, \delta)$. The perturbation model can be white-box with known rules g_p, black-box model g_q, or a composite of a series connection of the models $G = g_1(g_2(...g_m(\cdot)))$. In our paradigm, the model of natural perturbation directly uses the G notion and nuisance parameter $\delta \in R^c$ with c sets of perturbation direction. 52

4.2 The general architecture for generating natural perturbation using a learning model. It is assumed that the input data and generated perturbed data can be mapped into the same content space as [7]. 58

4.3 Reference traffic sign image from CURE-TSR [8] with various white-box based perturbations by increasing δ. 63

4.4 Example reference image and the perturbed ones provided in CURE-TSR [8] with challenge levels from 1 to 5. 64

4.5 Example images from CURE-TSD with challenge levels 1 to 5 (from left to right). The top row corresponds to the dark perturbation, the middle is associated with rain and the bottom row is the haze effect. 64

4.6 Example images CARLA [4] in foggy (top), sunset and night (middle), and the rainy (bottom) weather perturbation. 65

4.7 The histogram of neighbour accuracy for a subset of CURE-TSR [9] with respect to white-box based brightness perturbation among different benchmark models. 67

4.8 The histogram of neighbour accuracy for a subset of CURE-TSR [9] with respect to white-box based contrast perturbation among different benchmark models. 68

4.9 The histogram of neighbour accuracy for a subset of CURE-TSR [9] with respect to white-box based fog weather perturbation among different benchmark models. 69

4.10 The scatter plot for the test performance of the ResNet and GoogleNet models (trained on clean image) selected weak data from the VGG benchmark. 70

4.11 The scatter plot for the test performance of the ResNet and GoogleNet models (trained on clean image) selected strong data from VGG benchmark. 71

4.12 Comparing the drop of classification performance in various natural perturbations in terms of different challenge levels. The black line corresponds to the reference accuracy evaluated on the clean image data for the model (trained on clean image). There are different levels of performance drop in terms of various semantic perturbation directions and levels. (left: perturbed using white-box models, right: obtained from CURE-TSR dataset, considered as black-box) 74

4.13 Comparing the drop of detection performance in various natural perturbations in terms of different challenge levels in CURE-TSD dataset. The black line corresponds to the reference accuracy evaluated on the clean image data for the model (trained on clean images). There are different levels of performance drop in terms of various semantic perturbation directions and levels. (left: model precision, right: model recall) 75

4.14 Comparing the drop of detection performance in various natural perturbations in terms of different challenge levels in virtual data extracted from CARLA. The black line corresponds to the reference accuracy evaluated on the clean image data for the model (trained on clean images). There are different levels of performance drop in terms of various semantic perturbation directions and levels. (left: model precision, right: model recall) 76

4.15 Model robustness evaluation on unseen perturbation for various training procedures. The black solid lines denote the reference model performance on clean data and the dashed denotes the reference model performance on the unseen perturbation data. The plus, cross, and circle signs correspond to the different trained models evaluated on the unseen perturbation set. The gap between the dots to the black solid line reflects the robustness (the lower the better). 80

4.16 The performance evaluation for the proposed methods with different settings of exploration space. 81

5.1 The proposed framework of ODD monitoring strategy with and without map information. 86

5.2 Two events BN example of PM failure and the driving situation. 87

5.3 Formulation of the ODD monitoring based on the hard encoding for single perception module. 88

5.4 Formulation of the ODD monitoring based on the Bayesian Network with two perception modules in parallel. 89

5.5 The Gaussian kernel as the covariance function with different values of β_c. 90

5.6 Formulation of the ODD monitoring based on the map checking and stacked perception modules. 92

5.7 A demonstration of the landmark and lane marking observation and matching in the world frame and vehicle reference frame (based on ISO 8855 [10]). The measurements in the vehicle frame can be transformed to the world frame and vice versa based on the pose and calibration. 93

5.8 Sample front images from real (top row) and virtual (bottom row) driving environments. The left corresponds to the cases where the reference lane markings are unavailable, whereas the right two images correspond to the case with shorter lane information approaching the intersection. 95

5.9 The BN diagram of the ODD monitoring based on map checking. The green nodes are observable nodes that can be obtained in real-time as evidence. . 97

5.10 The scenario region setting in Town05 map using CARLA simulator, the selected regions are set to be out of ego vehicle's ODD. 100

5.11 The example conditional probability table for BN classifier. 103
5.12 The sample error measures with respect to lateral deviation, heading error as well as the choice of looking forward distance at straight lane. 104
5.13 The lane estimation parameters and ground truth values at the straight lane with minor weather changes (top: left lane parameters; middle: right lane parameters; bottom: projected error). 107
5.14 The lane estimation parameters and ground truth values at the road curve within ODD (top: left lane parameters; middle: right lane parameters; bottom: projected error). 108
5.15 The lane estimation parameters and ground truth values when ego-vehicle driving from road curve into S1. (top: left lane parameters; middle: right lane parameters; bottom: projected error). 109
5.16 The lane estimation parameters and ground truth values when ego-vehicle driving from S1 into S2. (top: left lane parameters; middle: right lane parameters; bottom: projected error compared with ground truth and with queried reference). 111
5.17 The lane estimation parameters and ground truth values when ego-vehicle driving in S3. (top: left lane parameters; middle: right lane parameters; bottom: projected error). 112
5.18 The landmark error when ego-vehicle exiting its ODD. (top: S2-noisy configuration; bottom: S3-map error). 113
5.19 The example Bayesian Network with fixed evidence. 115
5.20 The example mapping function from measured error to the probability of a correct pattern match, ($a = 0.3,\ b = 0.7,\ k = 20, i = 0.5$). 117
5.21 The ROC curve over different values of the threshold parameter $\lambda = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ in the CARLA testing scenario. (left: test set with more segments in S3, right: test set with more segments in S1-2) 118
5.22 The data processing pipeline for the shuttle bus driving data. 119
5.23 The example center camera image is classified as in/out of ODD based on geofencing. 120

5.24 The sample reference roll-out data and the corresponding augmentation and raw lane detection results (fog, contrast, and rain). The green text below reflects the last layer sigmoid function output from the neural network for lane classification for each image. 121

5.25 The sample snow augmentation and raw lane detection results. The green text below reflects the last layer sigmoid function output from the neural network for lane classification for each image. 122

5.26 The ROC curve over different values of the threshold parameter $\lambda = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ (from top right to left bottom). 124

A.1 The illustration of warning thresholds for lane departure referenced from [11]. The rate of lane departure V_d should smaller than 0.5 m/s. 162

A.2 The illustration of the bounding box required for localization by lateral and longitudinal and heading angle components. 164

A.3 The WATonoBus platform coordinates frame convention and transforms. . 168

A.4 The WATonoBus local map reference and vehicle heading notion. 169

A.5 The sample data collected from the WATonoBus experiment platform. . . . 169

List of Tables

1.1 Levels of driving automation according to SAE J3016 [12]. ACC: Adaptive Cruise Control ; LKA: Lane Keeping Assist; ICA: Integrated Cruise Assist; TJA: Traffic Jam Assist; TJP: Traffic Jam Pilot; HWP: Highway Pilot; FAV: Fully Autonomous Vehicle; CAV: Connected Autonomous Vehicle. Starting with level 3, the system is fully responsible for monitoring the system itself and the driving environment. In this research proposal, levels 1-2 are summarized as advanced driver assistance systems (ADAS) and levels 3-5 as higher levels of driving automation. 3

2.1 The recent standards proposed for assuring SOTIF aspects of ADS and the corresponding context related to ODD. (ISO: International Organization for Standardization; SAE: Society of Automotive Engineers; ANSI: American National Standards Institute; PAS: Publicly Available Specification; ASAM: Association for Standardisation of Automation and Measuring Systems) . 15

2.2 Popular Benchmarks in Autonomous Vehicle Research. The datasets for their sample location, size, raw-data construction, and annotation are compared. The sensor sources (RGB, LiDAR and IMU) and the corresponding annotations (Bounding-Boxes, Segmentation, Action, etc.) are available for Stereo, Reconstruction, Detection, and E2E field of research in autonomous driving. The recent Lyft L5 follows the same data format as nuScenes, and the current Waymo Open dataset share a comparable size to nuScenes, but at a 5x higher annotation frequency. (-) indicates that no information is provided. (+) indicate the dataset provided annotation or dev-kit for a customized extension. (NY: New York, SF: San Francisco. 23

2.3 The motion metrics in risk assessment for autonomous vehicles. 25

3.1 Infraction analysis across various operating conditions. The testing results of the two agents are grouped in two rows at each cell. The upper one corresponds to CIL and the CAL below. Each row's results are organized based on the qualitative order (light/medium/heavy). 46

4.1 Comparing the performance and robust training on clean data. 78

5.1 Example of Validated Likelihood of Failure (LoF) for selected situations. PM_1 and PM_2 can be different perception modules, for example, an object detection module, and a lane estimation module. 87

5.2 The sensor noise, weather configuration in simulation and post-noise adjustment setting in our experiment. 101

5.3 Example conditional probabilities of image-based perception module. . . . 123

5.4 Example conditional probabilities of lane marking match. 123

5.5 Precision and Recall on the subsets of roll-outs with specific perturbation. 125

A.1 Example performance requirement details for perception system 166

Chapter 1

Introduction

The move toward an Automated Driving System (ADS) is being driven by many potential benefits of the technology, such as increased safety, reduced traffic congestion, lowered emissions, and potentially increased mobility for those unable to drive. In order to realize these benefits, ADS technology must be introduced safely. The safety requirement is specified by the Operational Design Domain (ODD), where ADS requires a safe operation implemented by the functional modules in perception, decision-making, and control to guarantee certain output quality. This dissertation focuses on the extraction, augmentation, and monitoring of the ODD of the ADS, especially for the perception modules that rely heavily on machine learning technologies. Before outlining the research scope and objectives, this chapter introduces the background and challenges of ODD's ADS safety assurance problem.

1.1 Background and Motivation

Centuries of development in science and engineering have enabled the mobility revolution from horse carriages to automobiles. One appealing thought on the current transportation paradigm is still at the "horseless carriage" stage of mobility technology [13]. The carriage horses can learn for themselves through training and eventually carry out simple obstacle-avoidance skills, achieving a certain autonomy level. The motivation for autonomous driving is to recover the lost autonomy in modern vehicles and improve it further. It is expected that we will soon see the next transportation revolution where autonomous vehicles will be part of road traffic [14].

The idea of self-driving cars has had a long history, as scripts of the driverless vehicle came around as early as the 1920s [15]. During that time, the initial idea for implementing the driverless car was to use remote control, due to the enormous development in radio technology at that time [16]. The autonomy of the "horse" was substituted by remote human control. In the 1960s, an "automatically guided automobile" [17] was tested at GM's Technical Center in Warren, Michigan. The sensor technology development at that time enabled vehicles to follow wire laid in the road. The scene understanding and path planning problems were excluded from the driving task. Instead, the vehicle only needed to track the provided path (wire). Along with many successful applications of machine learning techniques in handwritten character recognition [18] as well as speech recognition [19] in the 1990s, the Carnegie Mellon University Navigation Laboratory made the first attempt at constructing neural networks for autonomous driving [20]. These efforts demonstrated that nearly autonomous driving was possible. In recent decades, numerous events and commercial efforts have further advanced to exclude human driver intervention in autonomous driving. Defense Advanced Research Projects Agency (DARPA) Grand Challenges [21] set up the goal for off-road driverless vehicles, as well as the mock urban environment. Other notable public test events [22] have demonstrated the ability of the current autonomous driving technology to manage real situations, including complex driving scenarios like roundabouts, junctions, pedestrian crossings, and traffic signs. The industry also pushed the development of autonomous driving for the potential market in future transportation. Commercial efforts like Google Waymo [23] and Tesla's Autopilot system [24] have further brought autonomous driving to the public's attention.

The deployment of self-driving cars can reduce the human operational error caused by driving fatigue and eventually reduce vehicle collisions. It is reported that driving autonomy could liberate human drivers from the tedious driving task, saving more time for humans and easing driving stress [25]. However, before humans can be confident that their driving safety is secure in the hands of autonomous driving functions, autonomous driving systems still need to demonstrate their safety and reliability. Such a safety demonstration is a prerequisite for authorities, society, end-users, regulatory bodies, the insurance industry, and OEMs to accept that ADSs make safety-relevant decisions with implications on human life [26].

A vital aspect of the safe use of automated vehicle technology is defining its capabilities and limitations and communicating these to the end-user, leading to a state of "informed safety". The first step in establishing the capability of an ADS is the definition of its ODD [27]. The ODD represents the operating environment within which an ADS can perform

Level	L0	L1	L2	L3	L4	L5
SAE (J3016)	No automation	Driver assistance	Partial automation	Conditional automation	High automation	Full automation
Control Authority	Driver only	Driver >> ADS	Driver> ADS	ADS in charge when activated	ADS in charge when activated	ADS
ADS Functions	-	ACC, LKA	ICA, TJA	TJP, HWP	RoboTaxi	FAV, CAV
Monitoring Task	Human driver continuously has to monitor the system and the driving environment			ADS is responsible for monitoring the driving environment		

Table 1.1: Levels of driving automation according to SAE J3016 [12]. ACC: Adaptive Cruise Control ; LKA: Lane Keeping Assist; ICA: Integrated Cruise Assist; TJA: Traffic Jam Assist; TJP: Traffic Jam Pilot; HWP: Highway Pilot; FAV: Fully Autonomous Vehicle; CAV: Connected Autonomous Vehicle. Starting with level 3, the system is fully responsible for monitoring the system itself and the driving environment. In this research proposal, levels 1-2 are summarized as advanced driver assistance systems (ADAS) and levels 3-5 as higher levels of driving automation.

the dynamic driving task (DDT) safely [28]. Therefore, knowing and real-time monitoring of the operational condition of the ADS is necessary for the system safety assurance [26]. In the context of ADS safety, this dissertation studies how to describe, extract and monitor the ODD in real time for high-level ADS functions, especially those with learning components.

1.2 Relevance

1.2.1 Driving Automation

To discuss the safety of automated driving, one has to differentiate the capabilities of ADS. These capabilities are classified by the Society of Automotive Engineers (SAE) J3016 [12] in terms of different driving automation levels. Table 1.1 summarizes the different levels of driving automation.

The most advanced commercially available automated driving functionalities are classified as level 2 systems (partial automation). A level 2 system can take over longitudinal and lateral vehicle control in specific driving situations, with the driver's restriction to continuously monitor the system and the environment. In case of an error or inadequate system behavior, the driver is responsible for overriding the advanced driver assistance system's (ADAS's) action [28].

As indicated in Table 1.1, a paradigm change occurs between a level 2 and a level 3 system because the human driver in level 3 does not have to monitor the system, or the environment, when the system is engaged. Hence, the human driver is not responsible for reacting (immediately) to a system failure or error. A level 3 system's restriction is that the driver must be receptive to take-over control after an adequate time frame whenever the system detects a failure or a situation it cannot handle. This restriction does not apply to level 4 anymore. The ADS is expected to reach a Minimal Risk Condition (MRC) when detecting a system failure or a situation it cannot handle [29]. While levels 3-4 are restricted to specific driving domains (e.g., highways or industrial zones), a level 5 system can handle any driving domain.

The human driver not being responsible for monitoring the ADS and its environment, and not making decisions for actuation, has profound implications on legal matters and liability [30], on ethical questions related to ADS actions [31], on ADS design, safety, and performance requirements, and on system safety testing procedures [27, 32].

1.2.2 Machine Learning in ADS

Machine Learning (ML) is increasingly applied to autonomous driving research, especially for perception and decision-making modules that cannot easily be mastered with traditional rule-based approaches [33]. ML components are incorporated to perform pattern recognition from very complex data, which is difficult to perform using algorithmic methods alone [34]. Perception modules extract features from high-dimensional sensory data from the camera, radar, and LiDARs to derive the "knowledge" of the dynamic traffic states and the attributes of the surrounding driving scenario, e.g., affordance mentioned in [35].

The learning-based system uses artificial intelligence techniques that make predictions from data based on empirical risk minimization [36]. ML is typically effective in processing high-dimensional data (image and point-cloud) with good performance in many applications [37, 38], and widely applied in scenario understanding [39] as well as the decision-making [40] functions in autonomous driving research. From a very high-level perspective, autonomous vehicles use ML models to generate localization, object detection, and tracking results. These results further construct states that forward into decision-making and control might also be composed of ML models operating during run time. A catastrophic collision may arise due to failures in one of the components mentioned due to the inaccurate incorporation of ML. It should be noted that the probabilistic nature of ML models

is less understood than algorithmic models since their functionality depends largely on parameters extracted from finite datasets [41] rather than being explicitly programmed based on rules. The failures in ML systems generate many safety issues in developing a safety-critical high-level ADS.

1.2.3 Safety Issues in ADS

ADS safety is paramount for customers, the road vehicle industry, and society. In 2017, the U.S. National Highway Traffic Safety Administration issued *Automated Driving Systems: A Vision for Safety 2.0* [42]. The Department of Transport Canada also published *Canada's Safety Framework for Automated and Connected Vehicles* [43] in 2019. In addition to the specifications of various countries, large companies such as Waymo and Tesla also released unmanned vehicle safety reports [44, 45], including software, hardware, test procedures, and human driver interaction safety issues. Besides the statistical report, Mobile Eye proposed a Responsibility Sensitive Safety (RSS) [46], which intended to formalize the "safety" and "responsibility" issue into quantitative mathematical models such that those semantics could correlate with parameters for planning and control.

The autonomous driving safety issue comprises two aspects: functional safety [47] and Safety Of The Intended Functionality (SOTIF) [48]. Functional safety concerns the potential hazards due to system, hardware, and software failure on road safety followed by the regulations in [47]. The ISO/PAS 21448 defines SOTIF as follows: "The absence of unreasonable risk due to hazards resulting from functional insufficiencies of the intended functionality or by reasonably foreseeable misuse by persons." As a complement to functional safety, the main focus of SOTIF is the possible errors caused by the perception and decision-making phase that does not conform to the expected behavior. The primary ingredient to the SOTIF problem in autonomous driving is verifying the algorithm robustness from various lighting conditions, dynamic scenes, and driving scenarios.

Research efforts have been committed to improving land vehicles' safety ever since the car came on the road in the 19th century. As the human driver takes all the responsibility of perception and decision-making during operation time, the low-level ADAS and safety functions mostly focus on the vehicle dynamic control assistance (e.g., ABS, ESP), and electronic body parts control (belt, airbag), as shown in Figure. 1.1. High-level autonomy shifts the load of perception and decision-making from human drivers to the ADS. Thus much focus has been shifted to the capability to **perceive and make judgement** over driving situations [46].

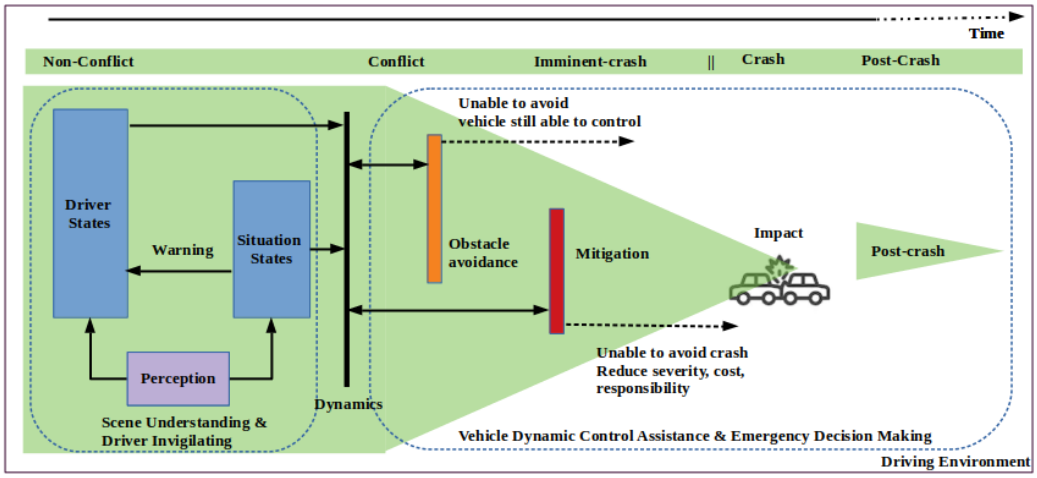


Figure 1.1: A holistic view of safety in modern ADAS system (figure adapted from [1]). The advanced ADAS monitors the condition of the traffic and driver and interacts with the driver by sending warnings and assisting with dynamic control and possible avoidance [2], including the crash mitigation and post-crash modules, to enhance safety.

Looking at Fig. 1.1 from right to left (from result back to cause), the mitigation and post-crash safety modules eventually aim to reduce the harm, which is characterized as physical injuries or property damage that is quantified by society [49]. By following the Automotive Safety Integrity Level (ASIL) proposed in ISO 26262 [1], the risk is the product between the probability of a crash during a conflict and the potential severity of the accident. A hazard is assessed with relative risk, which is a potential source of harm in the conflict situation. Uncertainty is highly connected with the safety issue in the non-conflict stage shown in Figure. 1.1 as a measure of the likelihood that conflicts originate from the ADS in the dynamical environment.

This dissertation focused on the non-conflict phase of safety issues associated with the defined operational domain given an ADS. As one crucial aspect of SOTIF, ODD needs to be carefully defined (before the release of a product) and monitored (during operation) to preclude an unacceptable risk of functional deficiencies from ADS modules, especially for those learning-based parts in perception. Hand in hand with the necessity of validating and monitoring the ODD of high-level ADS comes the challenge that classical automotive testing procedures and state monitoring methods are not directly applicable to modern ADSs and their perception parts.

1.3 Challenges

In recent years, modern vehicles have been transforming steadily from purely mechanical designs with chassis and engines to software-intensive cyber-physical systems. It was reported that Google's autonomous vehicle fleet was involved in 11 crashes from 2009 to 2015 during the approximately 1.3 million miles in autonomous mode test drives at low-speed [50]. The failure of perception and decision-making software modules can lead to disastrous consequences, such as a fatal collision of a self-driving car. For example, a Tesla autopilot vehicle crashed into a trailer because the perception system failed to recognize the obstacle against the background brightly lit sky [51]. Another report [52] showed that when Uber tested a self-driving car, the car was caught running red lights and did not appear to slow down when approaching a pedestrian, leading to a fatal accident in 2018.

In order to formally formulate the road driving task and its safety verification, I list a few challenges here and discuss how this dissertation relates to them.

1.3.1 Operational Design Domain Extraction of High-level ADS

Due to the complexity of ADS and the diversity of deployment environments, there is currently no clear framework or specification of extraction methods for the operational domain. For example, the operating conditions may range from a low-speed shuttle bus in a pre-mapped school area [53], to a Robo-taxi in complex city streets [23], to a very high-speed Highway-Pilot [54]. Even for a similar set of autonomous driving technologies, the operating scenarios would be different for a food delivery driverless vehicle operating in the street versus an autonomous trolley deployed in a storehouse. There are ongoing evolving standards for the operational design domain, starting with the SAE J3016, which set up an initial attempt to define ODD as "operating conditions designed to function for the AD" [12]. ISO 21448 addressed ODD in a similar way along with the introduction of SOTIF issues [48]. ISO 34503 brings up taxonomy and language schemes for ODD along with fundamental scenario classification schemes using a two-level abstraction [55]. However, the current standards only provide guidelines on what should be specified in ODD rather than a detailed methodology.

Another challenge is formally modeling the driving environment as part of the ODD. Environmental conditions are essential in influencing the safe operation of ADS-equipped vehicles. They also tend to pose one of the biggest challenges in ADS operation, particularly in early deployment or real-world trials. The environmental conditions can potentially

impact all ADS functions, from perception and planning to actuation control, as they might impact visibility, sensor fidelity, and communication systems. Most of the advanced functionality in ADS requires the perception of the environment [56], typically requiring human-level or even super-human-level ability of scene understanding. However, there is no detailed specification regarding "human-level" perception. Thomason et al. proposed a tree structure for digital scene representation, breaking the scene into more specific elements and arranging them by layers [57]. From the observer's point of view, Maurer et al. proposed that physical objects' spatial-temporal arrangement defines a scene [58]. When combined with automated vehicle testing, the initial scene needs to be formalized with all traffic elements, their behavior, and the vehicle's location. The developments over time are represented as a collection of landscapes, dynamic elements, and driving instructions [59]. There are promising works recently on scenario description language design, and taxonomy for autonomous driving [60, 28, 61]. However, it is still difficult to cover all the potential scenery and environmental aspects for driving scenario generation based on illustrational languages.

The complexity of ADS also poses difficulties for ODD extraction in modern ADS. The ADS can be viewed as a "systems of systems" in which many modules are black boxes. There is no way to explicitly know the specific state transitions of the modules, and only the input-output relations can be observed during tests. Each subsystem of the ADS interacts with the others but is not necessarily produced by the same company and is almost impossible to design in the same way. Such difficulties place new demands on testing and verification methods and require reliable measures of safety to specify the "designed" operating condition of the ADS.

1.3.2 Measurable Safety

In the context of ODD, miles and disengagement are not enough to show the quality of safety for the ADS. Before knowing any safety measures, it is not responsible for confirming the safe operation boundary through a large-scale, real-world deployment. During the Validation and Verification (V&V) process, ADS safety must be measured and quantified.

Various metrics and safety indicators exist for evaluating the vehicle motion and perception prediction mentioned in the current ANSI standard - UL4600 [62]. The most concerning issues with safety indicators, however, are how well they are correlated with safety and how to measure it. For example, the sensor performance on accuracy remains sufficient given changes in operational conditions and if they can see far enough ahead to

give accurate perception. At this point, these perception metrics need to be aligned with safety requirements. Outstanding sensor performance beyond the planning horizon might help with ride comfort or efficiency, but it might not be directly related to safety. On the other hand, the sensor failure may not directly result in an accident if the vehicle is driving in an open area. A major challenge in these metrics is that they usually require a ground truth to obtain the measure, such as the Euclidean distance between estimation and the ground truth. Even though it's easy to get to the ground truth in the simulation environment, there is still a difference between the simulation and the real driving situation. In the natural environment, acquiring the ground truth often relies on manual annotations and costly sensors.

1.3.3 Real-time ODD Monitoring

The ODD is designed to prevent potential accidents by limiting the driving environment for a given ADS. The autonomous vehicle is expected to fall back to the minimum risk condition (MRC) when it exits the ODD. Therefore, real-time monitoring of the system operating condition is crucial for ensuring the safety of ADS deployment.

Despite the designer's best efforts, it is always possible that the ODD can be violated in the operation stage. For example, roads can get slushy when it rains or snows a lot, which can make it hard to see lane lines and other signs. When ODD violations occur, some means are needed to monitor or even predict in advance that the environmental conditions in which the ADS is operating have exceeded its own pre-defined ODD. ODD monitoring in operation time for a cyber-physical system requires estimating all the related attributes of the system and environment. To the extreme, the ODD monitoring system is an "oracle observer" or "omnipotent" perception module that could access the ground truth of the integrated system.

The major challenge falls into scenario checking and driving risk monitoring during the operation. Scenario checking could be based on comparing the existing profiles with known information from geofencing [63] and speed sensors. However, over-reliance on geofencing is projecting the very high-dimensional ODD to a simple restriction on vehicle location. Advanced ODD monitoring aims to cover all the different factors that must be handled based on a detailed model. It is expected to have ODD violation detection formulated with driving risk evaluation that elucidates the underlying principles governing driving behavior. This driving risk evaluation must be formulated online and provides an accurate, human-consensus estimation of the safety level of driving.

1.4 Research Scope and Contributions

Motivated by the challenges discussed above, the subject of the proposed research is specifying, extracting, and augmenting the ODD of ADS functions in the development phase; and effectively monitoring the ODD or assessment of ADS function safety in the operation phase (structure shown in Figure 1.2). As with the modern ADS formulation presented earlier, multiple information sources are used that are processed by both data-driven and model-based procedures that generate either internal states, decisions, or control outputs, which are known to be safe and trustworthy in some rigorous operating conditions. From this standpoint, the ODD extraction problem involves offline testing procedures to use explainable, measurable metrics to clarify the system's designed ODD, which is a descriptive subset of the "real" safe operating condition. Although the problem of the standard language used to describe the ODD is not well established, the solution to the ODD extraction is proposed and examined in Chapter 3.

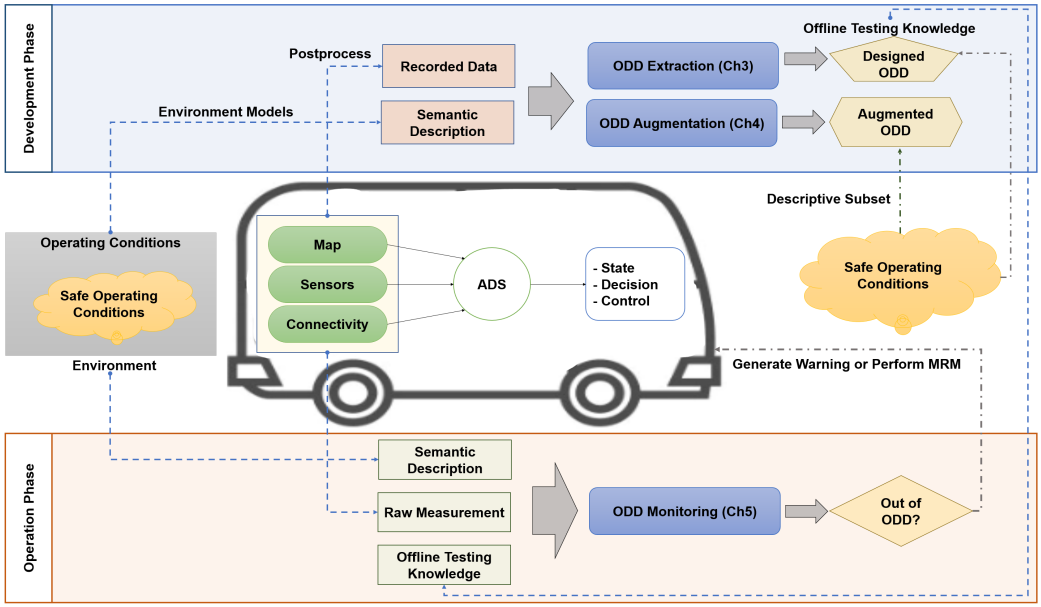


Figure 1.2: An overview of research scope.

A great effort has been devoted to improving the overall robustness of the modules with ML components. From the continuous engineering viewpoint, the problem is to extend the ADS's ODD to include new challenging operating conditions. The issue of transferring model knowledge to new environment conditions for camera-based perception modules is

the central study in Chapter 4 of this thesis. The operating conditions included in the system's ODD are introduced as specific perturbations to the original anchor data distribution. The model robustness is improved by utilizing the data augmentation techniques, and the probability of failures is measured during the offline training/testing procedures.

Another important theme in this thesis is the issue of assessing whether the ADS is safely operating in its ODD in real-time. It is possible to use the map and redundancy modules to generate the pseudo reference in the operation stage. The measurable error metrics of detection in real-time combined with the rough type of the operating condition are adopted as evidence of a probabilistic graphical model with embedded conditional dependencies to the system ODD and model fault diagnosis. The proposed solution is compared with commonly accepted geofencing and rule-based techniques in Chapter 5 and examined on both virtual and real driving data.

1.5 Chapter Outlines

The chapters are, to a large extent, self-contained and can be read independently. Chapter 2 contains a literature review on the relative matters for ODD and validation approaches for ADS. At the end of each of the main chapters, (Chapters 3 to 5), the proposed methods and algorithms are demonstrated on experiments with synthetic data. Chapter 4 includes experiments with benchmarks with both virtual and real data. Further tests on the real driving data collected from the WATonoBus shuttle are included at the end of Chapter 5. A summary of the thesis follows:

Chapter 2: Literature Review on ADS's ODD and validation approaches

This chapter includes a literature review on estimating the ODD of ADS and corresponding SOTIF ideas for autonomous driving systems. It addresses the operating condition modeling and the situation awareness for ADS. The validation approaches for ADS and common metrics are discussed. In addition, the common ODD monitoring strategies are listed.

Chapter 3: ODD Encoding and Extraction Procedure

The ODD extraction problem, with the operating condition encoding schemes, is proposed in this chapter. The framework comprises a scenario database, environment modeling, and the formal specifications for the ADS functions. A six-layer environment representation model is presented for ODD encoding, and the validation strategy is suggested to traverse layer by layer to control the exploration space. The formal specifications are formulated in a quantitative way that can adapt to vision tasks (data-driven models). Virtual driving simulation of preset task scenarios examines the case study on data-driven ADS policies. The ADS functions are validated using the black-box model, and the safety requirements are encoded in STL formulas.

Chapter 4: ODD Augmentation and Robust Learning

This chapter investigates the ODD augmentation problem for perception models that heavily rely on data-driven learning. The problem of extending the operational domain for the

perception system is modeled as a robust learning problem when facing unseen natural variations in the continued engineering process. The natural variation when ADS outside of its designed ODD defects the perception model performance is investigated, and the typical models that synthesize such perturbation are compared, and both white-box and black-box strategies are considered. The perception model robustness is characterized by the source data and the target data (the semantic perturbation direction) and the model that can shift such data to the "unseen" challenging data. Leveraging the neighbour accuracy property, a filtering strategy to extract more challenging data for future model validation and continued learning is proposed. The advantage of the proposed robust training based on model-based perturbation is evident in all tests, especially with more challenging natural perturbation.

Chapter 5: ODD Monitoring Based on Probabilistic Graphical Model

This chapter applies the measured offline performance from Chapters 3 and 4 to the real-time ODD monitoring case. The ODD monitoring in operation time problem does not have any labeled ground truth, and, as a result, one has to construct references from redundancy resources, such as a pre-collected map with lane and landmark features. The chapter proposes to use model-based map checking to compute the error metrics from the real-time perception results. The effectiveness of the method is demonstrated by applying it to the data collected from both virtual and real driving data from the WATonoBus shuttle testing platform.

Chapter 6: Applications and Conclusions

This final chapter summarizes the theoretical and experimental results and discusses the related applications and directions for further research.

Appendices:

The appendices contain related standards, requirements for ADS's sub-modules, and other ancillary information. The hardware and software setup for the shuttle bus testing platform and the data-driven models used in each chapter are also included.

Chapter 2

Literature Review

This chapter discusses related works on estimating the ODD of ADS and the associated SOTIF ideas for autonomous driving systems. The core of recognizing ODD is situation awareness, which eventually leads to operating boundary awareness. Modeling the driving environment and expressing the situation quantitatively would be the first step toward a formal ADS-ODD framework. Extraction of the ODD and effectively extending the ADS safety boundaries will depend a lot on the testing and validation procedures, especially for those with a black-box model. Finally, typical strategies to monitor the ADS during the real-time operation phase are investigated.

2.1 Recent ODD Related Standards

As the fundamental supporting concept of SOTIF, in-depth studies have been conducted on ODD for autonomous driving, and several standards and requirements have emerged recently. In this context, an overview and analysis of the standards related to ODD are presented.

The SAE-J3016 [12], and ISO-21448 [48] proposed the primary automotive industry-wide standard for all levels of automated driving. They were the first to address the SOTIF and ODD aspects (since 2018) and have continued to be updated. Although many concepts are still defined from a top-level perspective and are not very clear, they have laid the foundation for the definition of ODD.

Standards	Name	Organization	Context of ODD
SAE J3016	Standards for autonomous driving systems	SAE - US	Operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway.
ISO21448	Safety of the intended functionality	ISO	ODD is "specific conditions under which a given driving automation system is designed to function"
ISO 34501 - 34505	A family of standards for scenario-based safety evaluation, ODD taxonomy, and scenario attributes.	ISO	Top-level taxnomomy of the ODD consist with scenery, environment and dynamic elements
ISO 4804	Safety and Cybersecurity for Automated Driving Systems	ISO	The design, verification and validation for safety and cybersecurity for ADS
UL 4600	Standard for Safety for the Evaluation of Autonomous Products	ANSI - US	ODD should include infrastructure, weather and road conditions, objects and events, own and other vehicle conditions. Current ODD attributes are not enough.
BSI PAS 1880-1883	A family of standards for assuring safety of control systems for automated vehicles and related ODD taxonomy	BSI Group - UK	Proposed ODD comprises the attributes applicable to Level 3/4 ADS
ASAM OpenODD	The ongoing project aims to define a format that can represent an abstract ODD defined for a vehicle	ASAM - EU	Proposed requirements on ODD format, that should be searchable, exchangeable, extensible, machine-readable, measurable, verifiable and human-readable.

Table 2.1: The recent standards proposed for assuring SOTIF aspects of ADS and the corresponding context related to ODD. (ISO: International Organization for Standardization; SAE: Society of Automotive Engineers; ANSI: American National Standards Institute; PAS: Publicly Available Specification; ASAM: Association for Standardisation of Automation and Measuring Systems)

The ISO 34501-34505 [64] address that defining the limitations of ADS is more important than the capabilities. The family of standards then proposed a top-level taxonomy for ODD with scenery, environment, and dynamic elements that can be defined in the scenario-based testing procedure. The BSI PAS 1880-1883 [65] family of standards also similarly proposed the ODD taxonomy for testing the control systems for ADS and have complimented the use case constraints in part of ODD. Very recently, P. Koopman et al. proposed UL-4600 [62], which focuses on the evolution of vehicle intelligence and is developed and oriented to high-level autonomous driving. The UL-4600 [66] focuses on the safety risk assessment of fully autonomous driving without human driver intervention, requiring even more specific operational domain definitions. Most recently, the ASAM has called out the OpenODD project, aiming to create an open standard for defining a format for representing the ODD in a programmable and readable way. This is substantial progress from ISO-34503 and BSI-1883 standard toward programmable and exchangeable ODD for the automotive industry.

These recent regulatory developments show the efforts of research agencies and automotive industries to provide a clear and exchangeable taxonomy for ODD specifications. The ODD-related standards have a development trend toward a more exchangeable, more specific, and more easily understood by both humans and machines. In addition to the standard ontology, there is an urgent need in the work progress to describe operational

conditions with environment attributes (description), even with uncertainty and risk prediction, and the capability of determining whether or not a scenario is within the ODD (monitoring).

2.2 Scene Understanding and Situation Awareness

With all the standards mentioned above, one can agree that the ADS should consider the intended design of the ODD to address possible risks. When the system exceeds the defined ODD, a takeover request should be sent to the driver, and sufficient time should be allowed, or the ADS system should execute the backup safety operation to reach the MRC. One crucial task for ODD is driving situation awareness, e.g., how to describe the driving environment and understand the driving task during the operation. The complete scene comprehension tasks are critical for identifying potential hazards and ensuring safety. As laid out in a recent review on scene understanding, [67], the temporal cognition of the driving scene consists of background representation, event detection, and intention prediction.

2.2.1 Background Representation

From a cognitive point of view, background understanding provides a context that is even imprinted in the memory during driving in distinct scenarios. Moreover, the events, potential participants in events, and intentions of the participants are all derived from this context. For example, drivers never considered stopping signs and pedestrian jaywalking on a normal-operating highway while frequently pondering these two elements for motion planning in urban driving.

There are two general approaches to finding the core elements in the driving background representation. One common strategy is to follow traffic rules and the topology and road construction to decide which elements should be included to form the driving background [68]. The other is focused on the human drivers' perspective, where expert human drivers' attention is extracted and studied to determine the anchorage for the driving scene understanding [69].

The topology rules are historically collected road types by traffic state modeling in the topography area of study [67]. The HD maps [70], and Geographic Information System

(GIS) [71] provide clear guidance and reminder for safe driving nowadays. The road features such as directionality, and functionality are pre-collected and stored in digital maps such as ArcGIS [72] and OpenStreetMap (OSM) [73]. From the autonomous agent's point of view, one of the most critical tasks is using observation (GPS, visual sensors, or LiDAR) to learn these road attributes. In [74], the authors first proposed a method to classify the traffic scene based on the meta-features extracted from an ego-centric image observation. They used a two-stage system, first to extract the coarse semantic segmentation, and then to use this layer of features to distinguish between road types and detect vehicles and pedestrians. Later, Geiger et al. [75] extended the understanding to 3D content in the scene. They derive a reversible jump MCMC scheme that can infer the traffic scene's geometric and topological properties. One intriguing aspect mentioned in their conclusion is that the reasoning in the background representation can improve the accuracy of the object detectors, which aligns with the common sense of the inherent model between a scenario and the agents residing in that scenario. A parallel area of study of background representation is derived based on the detection of static components in the driving scenario, such as Simultaneous Localization and Mapping (SLAM) applications [76, 77]. The context, including the road, traffic lanes, and all other traffic participants, is considered part of the background. One original work proposed in [78] proposed a vanishing point detection method for visual-based road detection. They decomposed the road detection process into two steps: estimating the junction point associated with the main structured road and then segmenting the road based on edge detection. This study was then extended to the unstructured road by using Gaussian filter [79], and temporal tracking [80]. From the human driver's perspective, authors in [69] collected the driver attention data by an eye tracker in a general driving environment. They found that drivers' attention is mainly concentrated at the end of the road in front of the vehicle. Their finding of vanishing point echoes with the research [78] mentioned earlier. However, the vanishing point-based method imposes a strong assumption of vehicle heading and road curvature, which does not generalize well on curve roads and large vehicle headings. Instead of finding the lane markings or curbstones, [81] proposed a grid-based approach that assigns a lane and driving direction to each road patch, which performs more robustly if there are no clear road markings. The motion of vehicles may form a "road" in the unstructured driving scenario without any road markings. The performance of lane detection and background understanding in unstructured terrain can be further improved by combining indirect cues with conventional road layer detection.

2.2.2 Event Detection

Event detection can be considered a layer containing dynamic elements anchored to the background representation. Primarily, the autonomous agent aims to perceive other traffic participants' actions or behaviours around the intelligence vehicle. Based on the estimated behaviors of other vehicles and pedestrians, the autonomous agent should decide their driving or communication policy accordingly.

Kasper et al. [82] proposed an approach to driving maneuver recognition in structured highway scenarios using object-oriented Bayesian networks. They exploited the lane coordinate system and the individual occupancy of the vehicles. The resulting network can classify 27 driving maneuvers on the highway based on the defined situational features. As a constrained scenario, lane changes [83], overtaking [84] and rear-ending [85] are the most focused events in the studies of highway driving. In [86], authors use a dynamic Bayesian network as a filter to estimate traffic participants' behaviours by explicitly taking the interactions between vehicles into account. Notably, they decomposed the overtaking event on the highway into five discrete phases: acceleration, overtake, sheer out, and free ride. The ego-centric overtaking and receding detection are further studied using naturalistic driving data in [87]. The authors select Haar features and use Adaboost-cascaded classifiers for temporal detection of the driving events. Besides, authors in [88] applied recurrent neural networks to the trajectory analysis and classification of the surrounding vehicle's trajectories. Twelve different kinds of events based on different driving directions and road directions are formulated.

In contrast to the highway scenario, pedestrian event detection is a must in the urban driving scenario. A pedestrian event takes diverse forms under different conditions due to the high degree of uncertainty and lack of modeling. The authors in [89] present schemes to recognize pedestrian crossing events by mapping the spatio-temporal trajectories to traffic patterns. The traffic light sequences and timetables of nearby public transports are also provided as cues for the classification task. Mueid et al. [90] use a histogram of the oriented gradient of the motion history image for feature extraction and then use SVM for pedestrian action classification. Combining with the pedestrian tracking techniques [91, 92], authors in [93] investigate the evasive actions of the pedestrian in traffic conflicts based on permutation entropy for discovering dynamic characteristics of a time-series recording. In their examination, the learning-based method outperformed the traditional time-proximity measures such as time-to-collision and post-encroachment-time.

2.2.3 Intention Prediction

The intention prediction of traffic participants can be viewed as an extension of the event detection task. Especially when the vehicle or pedestrian's current state and action are known, the intention and future trajectories can be predicted with proper models of the traffic agents. Intention prediction seeks to provide anticipated trajectories and reachable sets for safe autonomous driving in the future. The set of future movements depends on both the background scenario and events in the traffic environment.

One common practice is to collect the features of the traffic agent and then apply learning methods to classify the prediction or assign a reward to the agent's possible behavior. Typically, the vehicle velocity, heading angle, and speed of the front vehicle are selected as features in [94], and a hidden Markov model is used for state transiting reasoning. The moving patterns of pedestrians are selected as features in [95, 96] to estimate the intention. Many complex scenarios, such as the uncontrolled intersection, are investigated in the work [97]. The author uses a continuous hidden Markov model to predict high-level motion, such as turning right, left, or straight. They also hard-code the driver's reward function based on driving safety, traffic rule, and time efficiency for optimal policy calculation. Phillips et al. [98] applied the direct learning methods using LSTM to learn intention prediction directly from the NGSIM [99] dataset concerning the ego position, dynamics, history features from past states, traffic features based on neighbouring vehicles, and rule features.

2.2.4 Remarks

The lack of a standardized structure for environment modelling makes it difficult to formalize the scenario understanding problem. The perception tasks (object detection, state estimation, and segmentation) remain independent and are not strongly associated with the environment model. This is a massive barrier to large-scale testing and verification for ADS functions and to exploring their "real operation boundary". Also, the perception stack relies heavily on deep learning techniques to extract features from high-dimensional data and map them to the specific task outputs (position, velocity, and intention). Because deep neural networks are black boxes, there is no way to tell how reliable or understandable these results are. Therefore, the measurable challenge for the ODD in real-time still exists.

2.3 Safety Validation and Related Metrics

Besides the scene understanding and situation awareness for autonomous driving, measurable safety is also at the core of the validation and verification process proposed by ISO-21448. The self-driving companies and Tier-I suppliers would be transparent about their validation strategy and corresponding operational safety metrics for road testing. This section reviews the recent safety validation strategy and the metrics.

2.3.1 Validation and Falsification of Learning-based Components of ADS

Validation examines system properties with either exhaust search or test case reports. In [100], the differential testing framework is proposed to detect erroneous behavior of the model, especially when the labeled data is not always available. The metamorphic testing is used in [101] with GAN, assuming the label-consistent property under feature space variations for autonomous driving applications. The metamorphic testing automatically generates tests to detect model defects, majorly by increasing the test cover space [102]. In addition to the test framework that reports the erroneous test cases to the model, there are also quantification measures of the uncertainty and confidence of the neural network output. For example, [103] utilizes the dropouts in the neural network structure to generate model uncertainty. This type of research uses various configurations of dropouts to generate fuzzy models that can obtain the confidence level of the model but not necessarily the correctness nor the robustness of the model.

Falsification aims to find any input or perturbation sequences such that a system violates a safety contract. The falsification of perception systems is still challenging, and most studies rely on naive random testing [104, 105]. This kind of random generation doesn't learn anything from the failed system iterations that came before. Many studies focus on generating adversarial samples to fool the trained model. For example, the authors addressed recent strategies proposed to adversarial training [106]. The basic assumption of this type of study is that the input is perturbed in a way that people cannot identify the change or are bound by some norm constraints, and the model's robustness to the bounded norm type of perturbation is analyzed by its aggregated behaviour on the shifted distribution. However, during autonomous driving operation time, the perturbation is primarily based on natural semantic variations that can be distinguished.

2.3.2 Road Testing and Progressive Metrics

With industry publicity and policymakers' requirements, public road testing has gradually become more important before a product can be launched. The companies, including but not limited to Waymo [107], Uber [108], and Baidu [109] have already tested their ADS in a confined section of public roads. A recent study also looked into the public's acceptance of the road test and the explanation model for it based on the trust heuristic and affect heuristic [110].

There are several progressive metrics in the ADS development cycle. The most common one in road testing is the number of miles driven safely. In 2016, the famous Rand report discussed the issue of how many miles of driving would be needed to demonstrate the reliability of autonomous driving [111]. Based on the statistical model adopted in [111], it would take roughly 8.8 billion miles of driving to demonstrate certain reliability at the required low fatality rate. Two years later, authors in [112] organized the road driving data from the California Department of Motor Vehicles (DMV) reports, which include 5,328 disengagements and 42 accidents involving autonomous driving products from various brands (L1-L2 autonomy) on public roads. Recently, Waymo has publicized the road testing performance data, which covers more than 6.1 million miles of automated driving in the Phoenix, Arizona, metropolitan area in [113]. The Waymo road test data includes operations with a trained operator behind the wheel and driverless operations from 2019 to 2020. According to reported data, autonomous vehicles are 15-40

The disengagement in autonomous driving detects internal system problems, or the test driver takes over control due to safety concerns. The disengagement rate also helps to measure the overall safety performance post-analysis. The idea behind the number of miles driven safely and the disengagement per mile is simple, where every test mile adds some risk to the autonomous vehicle trial. However, not every mile or each disengagement is equal; furthermore, the disengagements also depend on the test driver to judge whether it is operating unsafely. It is also hard to use these metrics as a debugging tool for the developed ADS without careful root-cause analysis since every mile driven or disengagement that happens is not the same. These progressive metrics are useful for post-analysis based on the data collection to judge if the developed ADS covers the intended ODD.

2.3.3 Validation Benchmarks

Road testing is expensive and not suitable for early-stage ADS function development. The research works in [114, 115, 116, 117, 118] have introduced challenging benchmarks for motion estimation, object detection, tracking, and semantic behavior estimation to closing the gap between laboratory research and challenging real-world driving situations. The recent popular autonomous driving-related datasets are listed in Table. 2.2. The development of datasets with many thousands of labeled examples led to spectacular breakthroughs in many Computer Vision disciplines by training neural network models in a supervised fashion. The publication of KITTI [114] set the modern benchmarks for autonomous driving-related vision tasks such as stereo, object detection, and odometry. CityScapes [119] explored the semantic segmentation study of the egocentric RGB recordings in multiple geo-locations and provided detailed annotations on the semantic segmentation. Further, with the pursuit of "data-quantity", authors in [120] proposed the distributed data collecting and annotation tool applied on Uber taxis across New York and San Francisco. However, due to the sensor quality limitation, the overall raw data quality is an issue in [120]. As the quality and quantity of the driving data become an implicit measure of the R&D and quality control potential of autonomous driving-related corporations, companies like Waymo [121], Baidu [122], and HONDA [123, 124] army their professional data sampling fleet for data collection.

Modern computer graphic techniques offer an alternative to manual annotation, generating large-scale synthetic datasets with pixel-level ground truth. However, the creation of photorealistic virtual worlds is time-consuming and expensive. Nevertheless, the popularity of movies and video games has led the industry to create realistic 3D content, which nourishes the hope of replacing real data with synthetic data. Simulation is a valuable approach for validating the safety of autonomous vehicles. Simulators could provide a virtual world to test the autonomous vehicle software and control algorithm exhaustively at a meagre cost. Recently released open-sourced simulation platforms, CARLA [4], and AIRSIM [130] are built on Unreal Engine, whose image generation is more realistic and has the flexibility to define several kinds of weather and environment settings. These simulation test benches are handy for pre-testing and validating path planning and control algorithms, especially for learning-based algorithms. Indeed, [131] directly applies the networks trained on the simulation platform to the real model. Several synthetic datasets [5] have been proposed and are being used by AI researchers (samples are shown in Figure. 2.1). However, the problem is that the simulator does not represent reality. Validating the system in the test does not guarantee that it will function at the same level of safety

Dataset	Year	Locations	Size (hr)	RGB-images	LiDAR	Ann. Frames	CAN/IMU	Bboxes	Segmentation	Night/Rain/Snow	Classes	Semantic-Action	Traffic Lane
CamVid [125]	2008	Cambridge	0.4	18k	NA	700	No	NA	Yes	No	32	NA	Yes
CaltechData [126]	2011	LA	10	250k	NA	250k	No	250k	No	No	Pedestrian	NA	No
KITTI [114]	2012	Karlsruhe	1.5	15k	15k	15k	Yes	200k	Yes	No	8	NA	Yes
CCSAD [115]	2014	Guanajuato	1.4	96k	NA	80k	Yes	NA	No	Yes/No/No	NA	NA	No
CityScapes [119]	2016	3x Europe	33	25k	NA	25k	No	NA	Yes	No	30	NA	Yes
Oxford [127]	2015	Oxford	-	2M	600k	NA	Yes	NA	No	Yes/Yes/No	NA	NA	No
Ford [116]	2011	Dearborn	-	20k	20k	20k	Yes	NA	No	No	NA	NA	No
TuSimple	2017	SF	-	4k	NA	4k	No	NA	No	Yes/No/No	Lane	NA	Yes
BDD100K [120]	2017	NY, SF	1k	120M	NA	100k	No	+	Yes	Yes	22+	NA	Yes
KAIST [128]	2018	Seoul	-	8.9k	8.9k	8.9k	Yes	NA	No	Yes	+	NA	Yes
Apollo [122]	2018	4x China	100	144k	144k	144k	Yes	140k	Yes	Yes	35	NA	Yes
nuScenes [118]	2019	Singapore	5.5	1.4M	400k	40k	Yes	+	Yes	Yes	23	NA	Yes
HONDA [117, 123, 124]	2019	SF	1	83k	27k	27k	Yes	+	Yes	Yes	+	Yes	Yes
Waymo Open [121]	2019	3x USA	5.5	1M	200k	200k	Yes	12M	Yes	Yes	+	NA	Yes
Lyft L5 [129]	2019	Palo Alto	17	60k	60k	60k	Yes	55k	Yes	No	+	NA	Yes

Table 2.2: Popular Benchmarks in Autonomous Vehicle Research. The datasets for their sample location, size, raw-data construction, and annotation are compared. The sensor sources (RGB, LiDAR and IMU) and the corresponding annotations (Bounding-Boxes, Segmentation, Action, etc.) are available for Stereo, Reconstruction, Detection, and E2E field of research in autonomous driving. The recent Lyft L5 follows the same data format as nuScenes, and the current Waymo Open dataset share a comparable size to nuScenes, but at a 5x higher annotation frequency. (-) indicates that no information is provided. (+) indicate the dataset provided annotation or dev-kit for a customized extension. (NY: New York, SF: San Francisco.

in the future.Challenges include complex object shapes and appearances and adversarial environmental conditions such as direct lighting, reflections from specular surfaces, fog, or rain.

In recent studies, Nidhi Kalra indicated in [111] that at least hundreds of millions of miles of safe driving performance is needed to demonstrate overall safety for autonomous vehicles. Shalev-Shwartz et al. mentioned a similar idea in [46] that in order to verify "safety" using the data-driven approach, at least thirty billion miles of data is needed to guarantee a probability of 10^{-9} fatality per hour of driving. The amount of data required to demonstrate safety takes many decades to complete. Furthermore, even with the number of driving recordings, annotations' effort would need decades of human power to finish. To cope with the safety verification in autonomous driving, interpretable frameworks to prove safety is needed. From this point of view, to sustain a business, the cost of validation and scalability of testing is an essential part of the autonomous driving industry [46].

Figure 2.1: Recent popular synthetic datasets and open-sourced driving Simulator. Examples from SYNTHIA [3], CARLA [4], Virtual KITTI [5], and AirSim [6].

2.3.4 Motion Metrics

Motion metrics that address driving risk ultimately boil down to Newton's laws. Kinematic-related factors, such as time, distance, or acceleration, are often used to achieve the risk assessment. A list of common motion-related metrics is presented in Table 2.3. Time-Head-Way (THW) is calculated by taking the time stamp that passes between the leading vehicle and the ego vehicle reaching the same location, whereas Time-To-Collision (TTC) is the maximum time that vehicle can continue the current trajectory [132]. Time-To-Reaction (TTR) further takes the driver's reaction time into account in the computation to calculate the remaining time to take the latest emergency maneuvers. The time-related metrics are usually based on the assumption of straight lanes, where the driving scenarios are limited to vehicle following and lane changing.

Like time-related metrics, distance and acceleration metrics are often based on the constant velocity or constant acceleration model. The distance threshold is used in [133] to implement the collision warning system. In [134], the deceleration required for an ego vehicle to bring the relative speed down to zero is used for emergency braking. Mobileye recently integrated the simple motion metrics into a formal Responsibility-Sensitive Safety (RSS) model, which attempts to clarify a safe zone for various driving scenarios [46].

Motion Metrics	Applications	Advantages	Limitations	References
THW	ACC	Simple and efficient	Insensitive to colllision risks	[139, 140, 141]
TTC	Collision Warning	Simple and efficient	Insensitive to lateral risks	[142, 143, 144]
TTR	Collision Avoidance, Mitigation Systems	Related to urgency actions	Affected by driver and environment	[145, 146]
Distance	Collision Warning	Straight forward	Low adaptability to environment change	[133]
Acceleration	AEB	Easy to implement in control	Depend on road conditions and braking ability	[134, 147]
RSS	Collision Warning, Mitigation Systems, System Monitoring	Formally safe	Depend on road geometry; Fixed safety boundary; Conservative	[148, 46]
Potential Field	LKS, Collision Avoidance, Motion Planning	Risk considering factors from driver, vehicle and environment	Complicated calibration; Depend on the field initialization	[149, 150, 151, 152]

Table 2.3: The motion metrics in risk assessment for autonomous vehicles.

The potential field theory was adapted from the obstacle avoidance of mobile robots research in [135]. The previous time-based or kinematic-based metrics are based on the point-mass motion model, whereas the potential field methods attempt to assign artificial fields to the traffic participants and the static components. For instance, in the design of a lane-keeping system, the potential field is used to describe the limits of the lane line and the possible collisions between the traffic components around it. These techniques were further implemented in the driving risk assessment, the planning modules in [136] and crash mitigation in [137]. But it is still hard to make artificial potential field models for complex and changing situations, which is usually the case for autonomous driving. Most recently, authors in [138] extracted the driver's risk field from the safety field and modelled it as a human-centric perception field independent of the driving scenario.

Overall, the motion metrics are straightforward in their physical meanings. In principle, the motion metrics are easy to implement in real-time given the assumption of known fixed scenarios, and; the robust and accurate state measures of the ego and the surroundings are always available. However, the scenarios go on from lane following to merging, intersections, roundabouts, etc. Furthermore, it is nontrivial to estimate the relative distance acceleration in real-time accurately, let alone correctly detect and identify the dynamic objects (vehicle, pedestrian) in the first place. The uncertainty of state estimation and the unexpected behavioural changes of traffic participants are not well captured using

motion metrics.

2.3.5 Statistical Metrics

Statistics-based measures, such as collision probability and estimation based on the Bayesian approach, are often used to assess driving risks [132]. The collision probability is calculated based on how likely each vehicle's reachable sets intersect with each other based on the probabilistic motion prediction results. Like the motion metrics, the collision probability is computed with the assumption of trusted perception results, often used for decision-making and planning modules. The first step to obtaining the collision probability is to make motion predictions for related traffic participants [153], with the Markov model [154] or Monte Carlo simulation [155, 156]. The collision detection is then checked by trajectory intersections or shared space and the final collision probability is obtained by the integral of joint probability distributions over potential collision regions [157].

Instead of using the explicit Markov model, another type of statistical metric is aligned with naturalistic driving data by learning from how people drive. For example, the authors in [150] use an unsupervised learning strategy to extract risk levels based on the natural driving behaviour of braking maneuvers. Similar features from the kinematic-based strategy were used to construct the risk function, including the relative position, speed, and acceleration of surrounding vehicles. In [158], the risk is extracted from NN based on the sequential dynamic features. Such risk can be designed in Reinforcement learning (RL) for risk avoidance tasks with the potential to align risk function with the statistical data using Inverse Reinforcement Learning (IRL) [159]. Overall, the aforementioned statistical metrics can model the uncertainty of the traffic participants' unexpected behavioural changes and are mostly applied to motion planning and decision making [160]. However, the uncertainty generated from the perception modules is not well-addressed since the estimated states are directly used to propagate the Markov model. Moreover, running the Monte Carlo simulation and deriving the probability distribution could be computationally expensive in a real-time deployment.

2.3.6 Remarks

Safety validation acts as a process to decide the system ODD, whereas the metrics are quantification aspects to describe and measure the details of that ODD. Most of the current

benchmarks are applied to the test-post-analysis approach, and the validation strategy is often offline, with the overall idea of verification being to extend the test cases and scenarios to expand the test coverage. The metrics play an important role in terms of measurable safety. The metrics under consideration are based on the assumption of reliable state estimation in order to analyze driving safety on the road with a certain degree of uncertainty. However, for high-autonomy ADS, the perception safety metric needs to be studied to address the ODD regarding sensor performance and state estimation algorithms. Especially in the open-ended driving environment, it would be helpful to have a metric for measuring how often the state estimation is not sure, such as object detection, classification, and localization, since they are not perfect in the real world. Such a metric represents how well the ADS can handle the current driving environment, e.g., whether the vehicle is driving in its ODD.

2.4 Encoding and Improving the ODD

2.4.1 ODD Encoding

An ODD captures the operational limitations on the generic road-environment components, such as road types, traffic volumes, and weather conditions [161]; working state of the vehicle and the corresponding ADS features, such as speed limitations, interrupt signals for ADS features, and the availability of data [61]; the state of the human driver in the event of a necessary disengagement [28, 162]. The list of peculiarities could escalate quickly with the expansion of autonomous driving tasks in complex scenarios and exploration of esoteric edge cases, including every aspect from perception and decision-making to manoeuvres and fault management [163].

Testing and validation for ADS safety by exploring the overall ODD search space require a holistic approach. The intended functional performance metrics and online behaviour of the ADS in a given ODD need to be assessed for safety testing and validation. The "safe" functional boundaries of an ADS for performing driving tasks are confined to its software and hardware implementation and strongly constrained by the driving environment. Furthermore, whether the multi-modal constraints on ODD can be expressed rigorously and ultimately accepted by both automakers and the government remains an open question.Ontological models [164, 165] are taken into account in the recent studies of traffic scenario modelling for expressiveness. An ongoing trend is finding a way to represent

the contextual entities in a formal and explicit conceptualization for traffic components [166, 167, 61]. Some preliminary works have applied ODD ontology models to driving data collection stage [168], and as driving affordance in static traffic scenarios [165]. However, a holistic framework is needed to test and validate the ODD for ADS formally and qualitatively.

Besides the regulation suggestions proposed by the government and industry mentioned in Sec. 2.1, many researchers dive into the driving scenario modelling and analysis of ADS use cases to formalize the corresponding ODD. Krzysztof et al. [167] defined a taxonomy for ODD of autonomous vehicles where the function specifications are subject to the vehicle model, the operational road environment model, and the operational world model. The PEGASUS project [169] adopted a similar scene-based testing and development cycle for ODD, which takes the traffic infrastructure, environmental conditions, and traffic level into account for highway autonomous driving applications. The vehicle model in ODD for surveillance and safety monitoring [39] implicates the self-representation of the running vehicle, including position accuracy, grip, system operation status, and reaction time. In general, formalizing ODD aids in the identification of scenario specifications that the ADS feature must handle during the function design process.Previous works decompose the problem of determining the delineation of safety operation boundaries for driving tasks in complex scenarios into multiple model specification layers.The research on formalizing ODD is still in the prototype stage. Most works attempt to categorize many aspects that may limit the performance of an autonomous driving system and eventually construct the ODD search space.However, follow-up quantification and validation are needed to specify the ODD standards, thus completing the testing and development cycle.

2.4.2 Robust Training

Robust training aims to minimize the loss corresponding to a bounded region of the adversarial samples [170]. It improves the model's robustness against the domain shift generated by the adversarial noise. In [171], authors studied the model robustness in terms of the adversarial attacks and demonstrated that small amounts of adversarial perturbations could cause the faulty behavior of a neural network classifier. Recent researches in [172, 173, 174] investigate adversarial perturbations with bounded-norm constraints and robust training strategies that regulate the loss to improve the overall robustness. It is worth mentioning that authors in [175] argue that increased robustness against adversarial training may decrease the normal clean data due to the data imbalance issue. This behaviour is not

expected in our ODD augmentation applications.

Other than the adversarial types of perturbations, studies in [176, 177, 178] investigate the neural network's robustness when facing natural distribution shifts. Closely related to our work, authors in [178] introduced model-based robust training methods for natural variations. The clean data are supplemented with variants generated by a fixed model that generates potential natural variations. The training loss is minimized by considering both the anchor data distribution and the perturbed ones.

2.5 Summary

The ODD may reflect the requirements of a driving automation feature interacting within the driving environment, which requires cross-disciplinary research efforts on the scenario model and representation, ADS function testing, and driving risk evaluation. The standardized structure for environment modelling is a critical prerequisite for large-scale testing and verification of ADS functions' ODD. The quantitative reliability measure of the ADS with neural network modules is needed for the safety assurance of high-level ADS. The development of ADS is unprecedentedly fast; however, the promotion of safety validation and online performance measurement is still at the beginning stage. It is vital to find failure examples and prove that the ADS can simultaneously handle any tasks generated in the given scenario, which is not adequately covered in the existing research.

Chapter 3

ODD Encoding and Extraction

One practical question for high-autonomy ADS is how to regulate their ODD definition, especially when the autonomous vehicle needs to take responsibility for driving state estimation and monitoring, where the operating environment sets limitations on perception. Setting up the operating limits for ADS requires encoding the right restrictions. In this chapter, the approach and architectural design are demonstrated to encode and extract the ODD of the ADS based on the task scenario and the corresponding requirements in the development and validation cycles. Moreover, the implementation examples are examined with two learning-based agents to demonstrate their feasibility.

3.1 Framework for ODD Acclimatization

Determination of the ODD for an ADS function can be considered as finding the boundaries of the driving environment condition that satisfies a particular evaluation criterion based on the potential scenarios, use cases [179] or driving tasks [28]. The exploration of ODD is similar to the logic handled by Hazard Analysis and Risk Assessment (HARA) as both attempts to determine the system's criticality under consideration in various running situations. HARA aims to identify functions and operational situations in E/E where hazards may occur and their combinations to form potentially hazardous events following FuSa [180]. Established techniques including HAZOP [181], FTA [182], and STPA [183] are used across industries to identify hazards that arise from expectations or hazardous scenarios based on descriptive studies [184].

Nevertheless, when the higher-level ADS function's normal performance needs to be considered under SOTIF, HARA alone can only handle "known-knowns" [185]. To handle the unknowns, ODD development with systematic testing of the ADS functions for learning-based components attempts to quantify performance and identify the failure cases. The definitions and processes provided in [180, 186] are prescriptive and not specific enough for the ODD development given an ADS system. In this section, the proposed ODD acclimatization method based on the use case following the goal-oriented approach suggested in the recent UL4600 draft [163] is presented. The framework demonstrating this proposed method, as seen in Fig. 3.1, is described in the following.

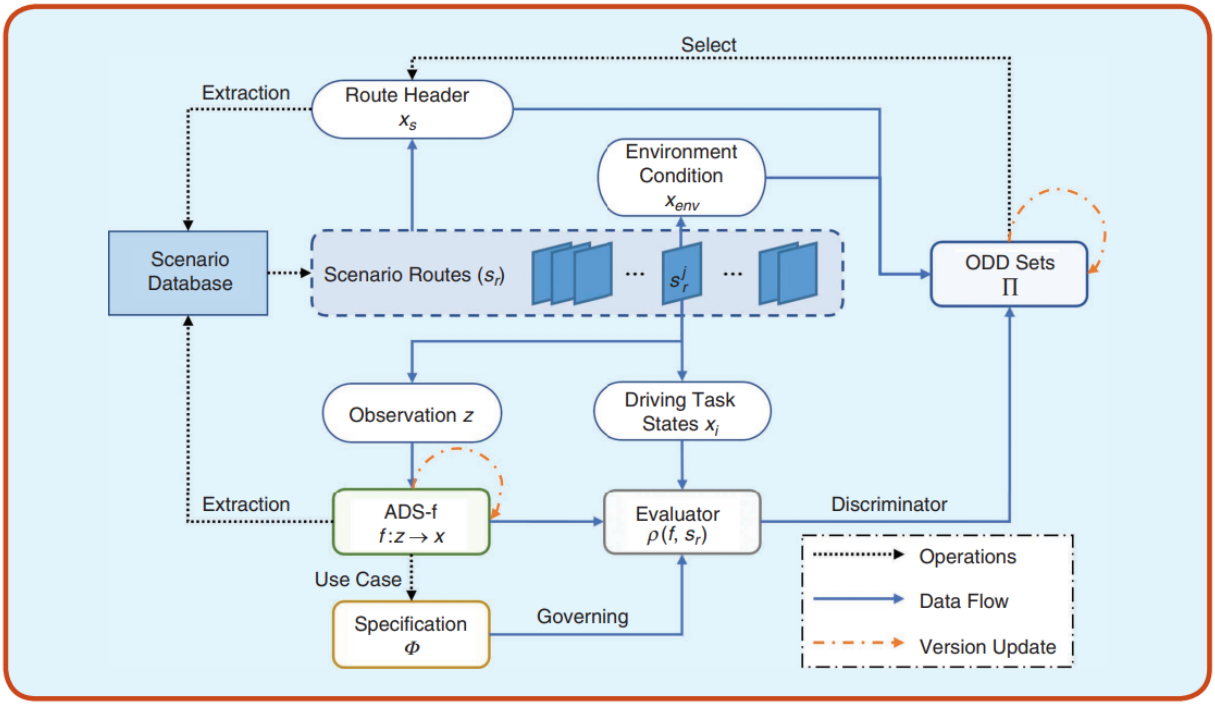


Figure 3.1: Framework of the proposed approach.

3.1.1 Scenario Database

The scenario database shown in Fig. 3.1 can consists of recorded data from real test runs, a virtual driving environment, and additional parallel test cases developed iteratively. Real scenes such as KITTI [114] and test drives [187, 188], have the advantage of being

realistic but are too expensive to collect and annotate. The lack of ability to configure traffic components in recorded real drives makes it challenging to explore agnostic corner cases. On the other hand, driving simulators such as CARLA [189] and the virtual scene generation language such as SCENIC [190] are affordable ways to propagate the driving data. As the test cases begin to swell iteratively, parallel testing [191] attempts to transfer the features from real driving scenes into the virtual platforms to exploit the advantages of both scenes.

The test drives and driving paths in simulation platforms can be grouped into various scenario routes s_r (see Definition 3.1.2), each corresponding to a track from the starting location to the targeting point. Following the same proposition in [186, 163], the scenario routes are composed of sequential scenes. At the same time, each of the scenes is considered as a snapshot of the driving environment. Scene s_r^j (see Definition 3.1.1) corresponds to the collected sensory data sampled in the real driving data, or the overall simulation world representation at a fixed time step j. To clarify the definition of scene and scenario routes used here in the scenario database, the idea of ISO-21448 is adopted and developed here.

Definition 3.1.1 (Scene, adopted from ISO-21448 [48])**.** A scene is a snapshot that contains dynamic elements (e.g. road users), describes the environment (e.g. road course, fixed obstacles, environmental conditions), and the system that is in this situation.

Notice that the typical scene is in the form of $s_r^j = \{sensorData, objectList, envConfig\}^j$. Sensor data can be a combination of RGB images, depth images as well as cloud points. The object list contains the description of dynamic elements and their states whereas the environment configuration stores the fixed description of the static driving situation. This definition can be applied to both real situations and the virtual world. The sensory data gathered from real driving scenes enjoy the realistic features while the labeling procedure to obtain environment and element description is laborious compared to the simulation.

Definition 3.1.2 (Scenario Route, modified from ISO-21448 [48])**.** The scenario route is a compilation of scenes that run in chronological order. The scenes in the same scenario route should share the same route header x_s, e.g. share the same configuration to achieve consistency in each test.

The scenario route is modified from the scenario definition in ISO-21448 [48] where constraints of consistency are applied. The scenario defined in ISO-21448 also mentioned that the order of scenes can be "different or branched, which is based on actions and events". A scenario route is then one sampled run in a confined scenario, as Heraclitus

states, "No man ever steps in the same river twice". The scenario consistency constraint is applied here to ensure the scenario route for bench-marking shares the same configurations, such as "2-lane highway", and "round-about". It is possible that there exist city roads and highways in the same map or scenario, however, this is not preferable in the validation process. The consistent features are used as query and extraction labels in coverage-driven verification [192] and further ODD selection.

3.1.2 Environment Modeling and States

The driver's knowledge of the driving scene is interpreted as knowledge states $x \subset \mathcal{X}$ where $\mathcal{X}$ includes all the environmental conditions, driving situation parameters, and task-related affordances [168]. Expressing the environment and organizing the knowledge in structured data is the foundation for describing the driving tasks and exploring the ODD for autonomous agents [59]. A two-layer model was proposed in [59] that separates the static structures and the dynamic traffic participants for test case generation. This model was further extended into a five-layer one in [193] and accepted in the PEGASUS project [169]. This thesis agrees with the most recent six-layer modeling structure as [194], where the authors added one additional layer incorporating the data availability based on [169]. The six-layer model for driving scenarios and corresponding layer presentation, quantification, and monitoring sources are depicted in Fig. 3.2. Layer-1 describes the geometric road-lane network, including the number of lanes, curvatures, and central line of the road. Traffic infrastructures such as barriers, traffic lights, and vegetation are represented in layer-2. The representations of the construction site are interpreted as temporal static modifications such as cones, bollards, and signage in layer-3. The layer 1-3 are often stored as base map in the form of SQL [195], XML [196] or Lanelet [197]. The first three layers of environment modeling can be considered static in daily frequency, especially for a given scenario route s_r. Hence, the header x_s stores the properties of layer 1-3 and is consistent among all the scenes in scenario route. For example, the following describes a segment of 4-lane highway with cones on the road.

$$x_s[L1] = \{[\text{roadType: highway}], [\text{laneNumber:4}]\}, \tag{3.1}$$

$$x_s[L2] = \{[\text{Barriers: null}], [\text{trafficLight:null}]\}, \tag{3.2}$$

$$x_s[L3] = \{[\text{Cones: True}], [\text{signage:null}]\}. \tag{3.3}$$

The final x_s is a union of (3.1) - (3.3) and could also include a pointer to the map file for the implementation purpose.

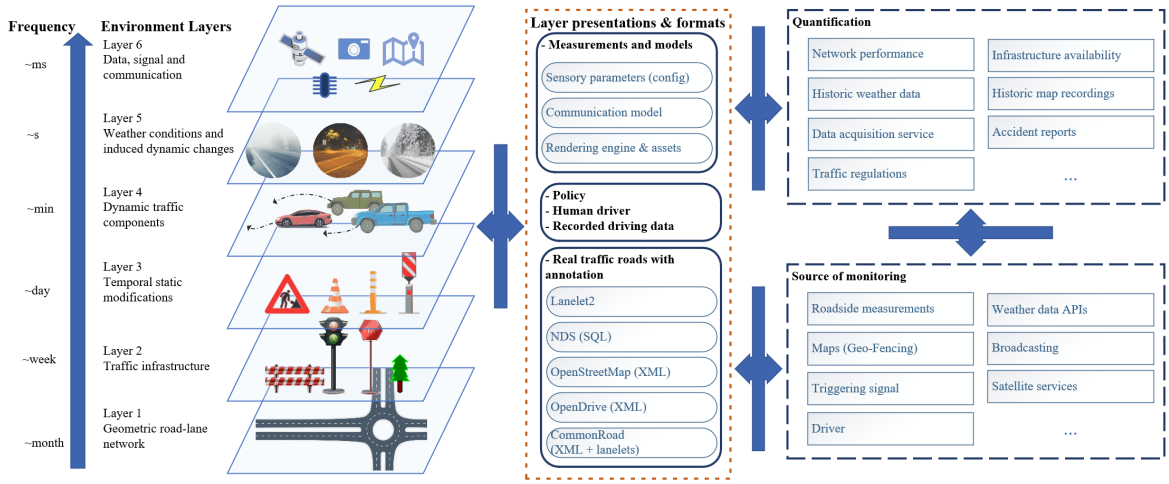


Figure 3.2: Environmental modeling and the corresponding quantification, redundancy, and monitoring. SQL: Structured Query Language; APIs: application programming interfaces; NDS: Navigation Data Standard.

Layer 4 includes the dynamic traffic components, which can be represented by driving policies, human participants, and even recorded traffics [138, 198]. Layer 5 describes the weather conditions that impact the sensory data, and parameters can be obtained by annotating real driving data and the rendering engine configuration in simulations. The data, signal, and communication availability are covered in the last layer including the information on sensors and possible V2I communications. For the same scenario route s_r, the environment condition x_{env} described in layers 4-6 may differ among the subsequent scenes. For example, an instance of the environment x_{env} describing the 4th layer with Level of Service (LOS) as level A (less than 10s passing intersection) [199], averaged speed of 70 km/h can be expressed as

$$x_{env}[L4] = \{[\text{LOS: A}], [\text{averageSpeed: 70 km/h}]\}\,. \tag{3.4}$$

At the same time, a parametrization example of the last two layers can be formulated as

$$x_{env}[L5] = \{[\text{cloudiness: 30}], [\text{precipitation: heavy}]\}\,, \tag{3.5}$$
$$x_{env}[L6] = \{[\text{map: } P_{map}], [\text{V2I: } P_{com}]\}\,. \tag{3.6}$$

The weather level in (3.5) can be quantified in the rendering engine or coarsely annotated with human understanding, such as 'light' or 'heavy'. The map and communication parameters P_{map}, P_{com} are structured data, including digital data delays, covered range, and

list of available information. In this thesis, only static settings of P_{map}, P_{com} are considered. Both the matching between virtual engine quantification, human annotation and the dynamic change of (3.6) will be explored in future research.

Aside from the attributes for environment modeling, the goal-oriented states x_i describe the affordances or knowledge required for the driving tasks [168]. For example, the camera-based perception module for lane following functions aims to extract the lanes and identify the front vehicle or obstacle. Thus, the lane attributes and the front object attributes are considered goal-oriented states. The goal-oriented states are typically the outputs of the under-evaluated ADS function, used to examine the performance of the ADS functions under the given route and environment. The goal-oriented states are simplified as x for convenience in the next section.

In summary, the expert knowledge from a scenario route has decomposed into three parts: the route header x_s, condition variable x_{env}, and the task states x_i. The expert knowledge $x_s \cup x_{env} \cup x_i \subseteq x$ is expected to articulate with the development of the autonomous driving industry. Multiple tests run on the scenario with the same route header could have various simulation configurations or the data augmentation techniques [200], e.g. the total of m pairs of operational domain couple $[x_s : x_{env,1}]$, $[x_s : x_{env,2}]$ ... $[x_s : x_{env,m}]$ can be established with m different configurations on the same testing scenario s_r. The operational domain pairs will be tested and falsified over iterative engineering development, and eventually, help to acclimatize the ODD sets.

3.1.3 Formal and Quantitative Specifications

In this part, the evaluator and specification part is presented in Fig. 3.1.

Coverage Driven Verification

The execution flow of the presented framework follows the coverage-driven verification flowchart depicted in Fig. 3.3. The coverage-driven verification method was evolved in the early 90s [201] which was applied to verify the complex microprocessor designs. Within the scope of ODD extraction, the verification procedure explores the coverage of "known-knowns" scenarios by better specification and identifies the potential violations from "unknown knowns". The goal is to maximize the coverage using constrained random scenarios and test generation. In the procedure depicted in Fig. 3.3, the verification plan is assumed

to be available in this study since there already exist various hazard analysis methods including Failure modes and effects analysis (FMEA) [202], Fault Tree Analysis (FTA) [203], HAZOP [204] and STPA [205, 206] that can generate corresponding rough metrics and scenario criteria. In short, the type of hazards, test scenario cases for the identified hazards, and criteria is assumed to be available in the first place. The major purpose of the work in this chapter is to make use of the verification plan and obtain a quantitative description of the ODD of an ADS. The augmentation methods (if not passed the specifications in Fig. 3.3) and strategy to generate random tests will be presented in the next chapter.

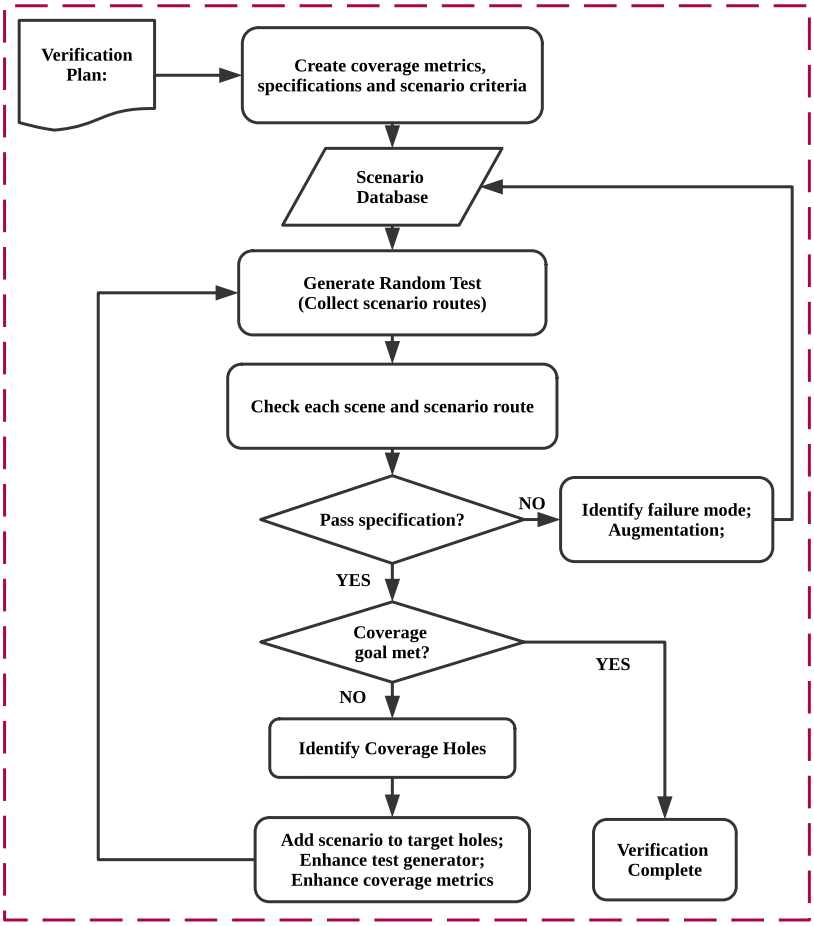


Figure 3.3: The flowchart of the coverage-driven verification.

ADS Function as Black-box

The modules of an ADS are treated as integrated logic $f : z \to x$, where the observation z corresponds to the function input as depicted in Fig. 3.1. For example, the radar and camera perception modules in Highway Assist System are responsible for object detection and classification. The radar subsystem alone takes the radio waves as input z and estimates the task knowledge x, including detecting instances of semantic objects and the relative distances. The z and x pairs may vary corresponding to different systems, such as RGB images and steering angle for the end-to-end (E2E) lane following function [207].

Quantitative Specifications

Designed use cases often confine the ADS function, and the specification Φ defines expected behavior or "safety contract" in the scope of ODD analysis. Specification Φ provides mathematical formulations for evaluating responsibilities, risks, or performance ranking of individual modules [46, 138]. The mathematical rules can be interpreted as safety contracts in an ADS function's design and testing phase. The temporal logic is a powerful tool for formal specification expression [208]. As a variance of the temporal logic, the Signal Temporal Logic (STL) enables the expressiveness over real-valued signals, which has been widely applied in robotics [209]. STL is one variety of temporal logic, which will be minimally introduced here. Other temporal logic such as Linear Temporal Logic (LTL) and Metric Temporal Logic (MTL) can be applied in a similar manner with slight change in the grammar.

Definition 3.1.3 (STL formula [210])**.** The STL specification is a propositional structure that can be generated by the following grammar:

$$\Phi := \sigma \mid \neg\Phi \mid \Phi \cap \Phi \mid \bigcirc_I \Phi \mid \Phi U_I \Phi, \tag{3.7}$$

where $\sigma : \mathbb{R}^n \to \mathbb{B}$ is an atomic proposition predicate defined over sequence $x_T^1, ..., x_T^m$ (including true and false), Φ is the production rule variable, $\neg$ is the symbol for not operator, $\cap$ stands for the Boolean or operator. The closed non-singular interval I defines the range of the superscript j between the sequence from 1 to m. The notion $\bigcirc_I$ stands for the next sample operator and the U_I denotes the symbol for the temporal until operator.

The robustness function $\rho : \mathbb{R}^m \to \mathbb{R}$ are often used to decide the true or false of the predicate by comparing the evaluation over the trace and given specification $\rho(x_T^1, ..., x_T^m; \Phi)$

with a cut-off boundary. Common temporal operators can be defined syntactically with (3.7), for instance the union $\Phi_1 \cup \Phi_2 := \neg(\neg\Phi_1 \cap \Phi_2)$; Finally ($\diamond\Phi$) $F_I\Phi := \top U_I\Phi$; Globally ($\Box\Phi$) $G_I\Phi := \neg F_I\neg\Phi$.

A scenario route consists of the sequential trace of signals for the under-tested ADS function; the task states x. The qualitative semantics of the STL formula Φ over the trace x with time index t is inductively defined as the following

Definition 3.1.4 (STL Robust Semantics, [210])**.** One practical question for high-autonomy ADS is how to regulate their ODD definition, especially when the autonomous vehicle needs to take responsibility for driving state estimation and monitoring, where the operating environment sets limitations on perception. It is necessary to encode the proper limitations for setting up the operating boundaries for ADS. In this chapter, the approach and architectural design are demonstrated to encode and extract the ODD of the ADS based on the task scenario and the corresponding requirements in the development and validation cycle. Moreover, the implementation examples are examined with two learning-based agents to demonstrate their feasibility.

Given the trace x and the evaluation function ρ, then the robust semantics of an STL formula Φ with respect to x at time sample t is

$$x, t \models \sigma \text{ iff } \sigma(x(t)) \text{ is true ;} \tag{3.8}$$

$$x, t \models \neg\Phi \text{ iff } x, t \not\models \Phi; \tag{3.9}$$

$$x, t \models \Phi_1 \cap \Phi_2 \text{ iff } x, t \models \Phi_1 \text{ and } x, t \models \Phi_2; \tag{3.10}$$

$$x, t \models \Phi_1 U_I \Phi_2 \text{ iff } \exists t' \in t + I, \\ \text{s.t. } x, t' \models \Phi_2 \text{ and } \forall t'' \in [t, t'], x, t'' \models \Phi_1. \tag{3.11}$$

The robustness evaluation ρ determines how robustly the specification is satisfied based on the sampled trace. Given an ADS function f, the safety specification Φ, the accepting environment candidate set $\{x_s : x_{env}\}$ is the subset of the tested ODD Π_Φ for which traces of the ADS function output satisfies the specification. The remaining set of the domain is denoted by $\Pi_{\neg\Phi}$.

The evaluator in Fig. 3.1 are expected to quantitatively evaluate the function performance over a specific task s_r. Here, several examples on safety-related metrics in realization are presented.

Example 3.1.1 (Offline Error Metrics)**.** *Under offline settings, where the ground-truth are typically available, the following metrics are used to evaluate the system under test given*

a test set $\mathbb{D}$,

$$MAE = \frac{1}{|\mathbb{D}|}\sum_{i\in\mathbb{D}}\|x_i - \hat{x}_i\|_1, \tag{3.12}$$

$$MSE = \frac{1}{|\mathbb{D}|}\sum_{i\in\mathbb{D}}\|x_i - \hat{x}_i\|_2, \tag{3.13}$$

$$\textit{Speed-weighted MAE} = \frac{1}{|\mathbb{D}|}\sum_{i\in\mathbb{D}}|v_i|\|x_i - \hat{x}_i\|_1, \tag{3.14}$$

$$\textit{Cumulative Speed-weighted Error} = \frac{1}{|\mathbb{D}|}\sum_{t=0}\|(x_{i+t} - \hat{x}_{i+t})v_{i+t}\|. \tag{3.15}$$

The speed v_i *denotes the speed of the ego vehicle at the* i*-th sample and the time range* t *indicates a period of time to investigate the metrics.*

The offline metrics in Example 3.1.1 are often used to evaluate the normal performance over batches of data. Notice that the offline metrics does not guarantee online performance due to the domain differences. It is demonstrated in [211] that the offline metrics such as MSE and MAE do not necessarily hold a strong correlation between the successful performance of online vision-based E2E policy. This is reasonable because autopilot may make a fatal mistake in the short term, such as failing to perceive lane lines for five consecutive frames, and even if the rest of the perception is correct, a single fatal error can still lead to a catastrophic accident. The Speed-weighted MAE and Cumulative Speed-weighted error mentioned in Example 3.1.1 can increase the correlation between the accidents that happen in driving, however, they are not directly related to the requirement or safety contract in real-time performance. The STL statements to specify the safety contract required for the ADS function can be used, here list some examples.

Example 3.1.2 (Safety Contract - Object Detection Ability)**.** *The vehicle is expected to detect the objects within the field* $\hat{\Omega} = \{obj_1, ..., obj_k\}$ *and the position error smaller than a given bound* ϵ*. The corresponding STL statement can be written as*

$$\square\ \Omega = \hat{\Omega}; \textit{ and } \quad \square\ (\|p_i - \hat{p}_i\| < \epsilon, \forall i \in \{1, ..., k\}). \tag{3.16}$$

Example 3.1.3 (Safety Contract - Static, Passive and Passive Friendly [212])**.** ***Static Safety:*** *The agent at position* p *is expected to have a positive distance to all static obstacles, where the position of the obstacle is* o*,*

$$\square\ \|p - o\| > 0. \tag{3.17}$$

Passive Safety: *The agent needs to remain a positive distance to all static obstacles while driving at speed* v

$$\square\ (\|p - o\| > 0) \cap (\|v\| > 0). \tag{3.18}$$

Passive Friendly Safety: *The agent needs to have sufficient maneuvering space for static obstacles, with minimum braking capability* b*, and maximum reaction time* τ

$$\square\ (\|p - o\| > \frac{v^2}{2b} + \tau v) \cap (\|v\| > 0). \tag{3.19}$$

Example 3.1.4 (Longitudinal Safety for Vehicle Following [213])**.** *Suppose both agents travel in the same direction, the acceleration of the front vehicle must be within the longitudinal bound* $a_f \in [-a^{long}_{max,brake}, a^{long}_{max,accel}]$*, and if the distance between the front vehicle* $\|p_f - p_r\|$ *is greater than the safety distance* d*, then the acceleration of the rear vehicle can set into the range* $[-a^{long}_{max,brake}, -a^{long}_{max,accel}]$*; otherwise should be limited to* $[-a^{long}_{max,brake}, -a^{long}_{min,brake}]$*. The safety contract in STL form can be written as*

$$\square\ (v_f \geq 0 \cap v_r \geq 0); \tag{3.20}$$
$$\square\ a_f \in [-a^{long}_{max,brake}, a^{long}_{max,accel}]; \tag{3.21}$$
$$\square\ (a_r \in [a^{long}_{max,brake}, a^{long}_{max,accel}] \cap \|p_f - p_r\| \leq d) \tag{3.22}$$
$$\rightarrow a_r \in [-a^{long}_{max,brake}, -a^{long}_{min,accel}] \tag{3.23}$$

The aforementioned examples are common safety contracts in robotics and autonomous driving that be encoded in the assume-guarantee STL requirements. These requirements are applied to test and falsify the ADS functions to further explore their ODD in this thesis. More advanced STL safety contracts are available in [214] that address the RSS rule, which can be implemented in the proposed framework as well.

3.2 Case studies

The experiments are designed to extract the given ADS module's ODD according to a specific task scenario. The testing platform is CARLA 0.9.10, with the interface to toggle the Unreal Engine's weather parameters. In this experiment, the primary focus is on extracting the daytime and weather aspects of operating constraints as a demonstration. Notice that the other aspects of ODD can be elicited in a similar way.

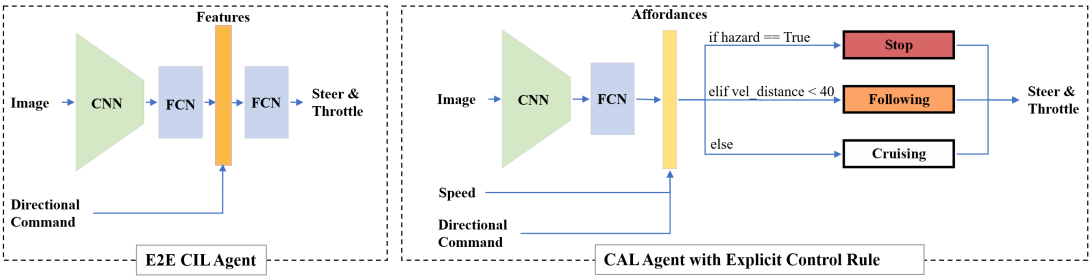


Figure 3.4: Two network architectures for imitation learning based autopilot module tested in the experiment. The E2E CIL Agent (left) takes an RGB image as input and directly maps to action based on the high-level command. The CAL Agent (right) maps the image to explicit states, and the action is decided explicitly based on the perceived affordances.

3.2.1 ADS Policies

Two imitation learning-based ADS modules are chosen for the ODD acclimatization experiment, as shown in Fig. 3.4. The CIL Agent [131] maps the observation image to implicit features and then maps to action based on the high-level directional commands (right, left, straight). The CAL agent [215] maps the image directly to affordances that could explicitly execute the control logic. The directional command and vehicle speed are also concatenated to form the affordance states. The affordances are similar to those in [215]. The explicit states include the hazard-stop trigger, distance to the front vehicle, distance to the center line, and the relative angle between the vehicle heading and the road. For simplicity, the traffic light and speed sign identification are desolated compared to [215]. There are three modes in the explicit control module. If the hazard stop state for the pedestrian is triggered, the vehicle will enforce a stop mode in the longitudinal controller. The vehicle following and cruising mode is based on the PID controller with the same setting as [215]. The image size of the Carla dataset is 200×88 [131]. The VGG16 network is used as a CNN decoder in both ADS policies. The training data is sampled from Town 01 and 02 in CARLA with three types of weather: ClearNoon (CN), ClearSunset (CS), and WetNoon (WN). The weather can be quantitatively expressed as the following with direct reading of the weather parameters from Unreal Engine. For example, the WetNoon

Figure 3.5: The testing scenarios used in this study. Left: driving straight following the lane at single-lane tertiary. Right: one-turn road following task based on the directional command at T-junction. The ego-vehicle is initiated at the spawning point (SP), and the mission is to driving towards the corresponding ending point (EP).

can be described as

$$\begin{aligned} x_{env}[L5] \quad = \quad & WN = \{[SunAltitude : 90], \\ & [SunAzimuth : 10], [Cloudiness : 10], \\ & [Precipitation : 30], [Wetness : 30]\}, \end{aligned} \tag{3.24}$$

with other zero-valued parameters neglected in (3.24).

3.2.2 Task Scenario & Environmental Conditions

The testing driving scenarios being considered are located in a selected region of Town 03, demonstrated in Fig. 3.5. Each time, the ego-vehicle with the policy under-tested initiated at the spawning point and drove towards the task ending point. The left scenario of Fig. 3.5 depicted a single-lane tertiary road with no cones and signage. The right scenario is corresponding to the turning task at the city road. There are three different spawning points for the ego-vehicle to be initiated randomly, and the high-level directional command

is set as constant for each route. The traffic condition was initiated randomly with two variables, the number of vehicles and pedestrians on the map. Both the number of total cars and pedestrians are randomly set, ranging from 20 to 80.

The weather and daytime effect considered in this paper is listed in Fig. 3.6. With the cloudy, rainy, foggy weather and its discrete classification by degree; combined with time-space from daytime, sunset, and night, the total discrete space of the environmental conditions x_{env} includes 27 combinations. The weather parameters such as cloudiness, precipitation, precipitation deposits, wetness, and fog density are continuous values ranging from 0 to 100 in CARLA. In this set-up, the light condition corresponding to any parameter value ranges within $[0, 30]$, medium with $(30, 70]$ and heavy with $(70, 100]$.

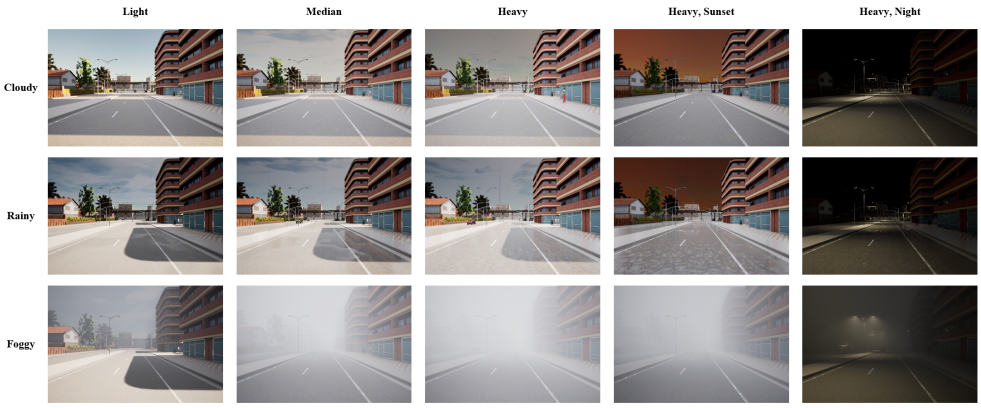


Figure 3.6: The RGB image inputs under various environmental conditions. The left three columns correspond to the daytime (SunAltitude = 60), whereas the right two columns are sunset (SunAltitude = 0.5, SunAzimuth=180) and night (SunAltitude = -90). The cloudy, rainy, and foggy weather are demonstrated in rows with three discrete conditions, light, medium, and heavy.

3.2.3 Evaluation

For each of the driving tasks shown in Fig. 3.5, 100 test cases are generated under any environmental condition. The ADS agent is initialized at SP in the map, and the surrounding traffic condition is initiated with a random number of vehicles and pedestrians. Other actors' behavior in the environment is strictly based on traffic rules and designed to follow the next way-points in the simulation environment. The number of test cases that

need to generate for each subject is expected to increase to cover enough traffic cases for more complex driving scenarios. The time limit to reach the goal is set as the time needed to drive along the optimal path at 10 km/h. Notice that the red lights are ignored in the T-junction scenario since the signals are enforced to match the spawning position and the corresponding route. The cruising speed of the agent is set as 30 km/h.

Table 3.1 shows the two learning-based agents driving under various operating conditions. The test case will be marked as not completed if the vehicle did not reach the corresponding EP within the time limit or if there is a collision between the ego-vehicle and other actors during the task. Two different sets of safety contract violations are investigated for both the CIL agent and the CAL agent. Due to the black-box nature of the CIL agent, only the expected lane following behavior is examined. That is, the vehicle should remain in its current lane at all times during the task. The distance between the vehicle's position and the center-lane d can be obtained in real-time in the simulation. The staying-in-lane specification can be expressed as

$$\Phi_{SL} = \square\ d < \frac{w}{2}, \tag{3.25}$$

where w is the lane width. In addition to Φ_{SL}, CAL agent further takes the detection ability listed in Example 3.1.2 into account. The following STL specification can monitor the perception accuracy of the affordance, such as the distance to the front vehicle l in real-time,

$$\Phi_P = \square \sum_{i=t-5}^{t} \|l_i - \hat{l}_i\| < \epsilon. \tag{3.26}$$

The specification Φ_P in (3.26) implies the requirement that the cumulative error for observations of the front vehicle distance over the last five consecutive frames should be less than the predefined boundary ϵ. In this experiment, ϵ is chosen as 3 m. Table 3.1 also include one of the offline metrics, MAE, of the estimated front distance for each task. It can be noticed that the offline metric MAE shares a weak correlation with online performance. It is still ongoing research on how to find offline metrics that share a strong correlation with online performance, as mentioned in [211]. Furthermore, contract violations exist even when the vehicle completes the route as the safety specification is a more vital constraint than finishing without collision. Occasionally, CIL and CAL agents failed to stay in the current lane. This violation often happens on the right turns, resulting from the sharper curve of the right turns than the left turns in the simulation environment.

The testing results in Table 3.1 agree with the results in [215]. The CAL agent performs particularly well in staying in the lane and avoiding collisions with other actors in the

new environment than the CIL agent. The CAL agent has better generalization ability due to the explicit control logic based on affordance mapping. The acceptable operating domain for an ADS module can be determined with multiple Safety Performance Indicators (SPIs) together, as proposed in [163]. Consider the SPI that the frequency of real-time specification violation should be less than 5%, and at the same time for CAL agents, the MAE should be less than 0.2 m. The accepting operating domain under the SPI mentioned above can be concluded for both agents based on Table 3.1,

$$\Pi^{CIL}_{x_{lf},\phi_{SL}} = null,\ \Pi^{CIL}_{x_{turn},\phi_{SL}} = null \tag{3.27}$$

$$\begin{aligned}&\Pi^{CAL}_{x_{lf},\phi_{SL}\cap\phi_P} = \{[No \times (C, R : L - M, F : L)],\\ &[Ss \times (C, R : L - M, F : L)]\}\\ &\Pi^{CAL}_{x_{turn},\phi_{SL}\cap\phi_P} = \{[No \times (C, R : L - M)],\\ &[Ss \times (C, R : L)]\},\end{aligned} \tag{3.28}$$

where the acronyms Noon (No), Sunset (Ss), Cloudy (C), Rainy (R), Foggy (F), Light (L), and Medium (M) are used. The complements of (3.27) and (3.28) can be used as a data augmentation environment handle in the further development cycle. The cloudy weather does not have a severe impact on the model performance, as depicted in Table 3.1, since the image background variation does not appear strongly related to the driving tasks. On the other hand, cloudy weather has a more substantial influence on the CIL than the CAL agent. The compact features extracted by CIL can be easily perturbed by the background textures, thus losing weight for relevant features in the driving tasks. However, rainy, foggy, and night have a massive impact on the model performance, as useful information is taken over by the noise. In particular, both models exhibit a substantial performance loss at night and in a foggy environment.

3.3 Discussion and Future Work

In this chapter, the framework for ODD encoding and extraction for the ADS function is presented. For ODD encoding, a six-layer model for representing the environment is shown, and it is suggested that the validation strategy should move from one layer to the next to control the exploration space. In fact, the construction of the scenario library and the test environments are similar to what is addressed in [194]. The ADS functions are validated using the black-box model, and the safety requirements are encoded in STL formulas. It has been shown how to use simple coverage-driven validation to test how

Driving Tasks		Lane Following in Scenario-1			Left/Right Turns at Scenario-2 T-Junction		
E.C. \Metrics		Avg. Completion (%)	MAE (m)	Avg. Violation (%)	Avg. Completion (%)	MAE (m)	Avg. Violation (%)
Noon	Cloudy	93 / 91 / 88 100 / 100 / 100	- / - / - 0.14 / 0.15 / 0.15	8 / 8 / 14 0 / 0 / 0	86 / 85 / 80 100 / 100 / 100	- / - / - 0.15 / 0.14 / 0.17	20 / 25 / 39 0 / 0 / 0
	Rainy	96 / 86 / 68 100 / 100 / 96	- / - / - 0.14 / 0.16 / 0.22	6 / 15 / 40 0 / 1 / 5	82 / 80 / 63 100 / 100 / 93	- / - / - 0.16 / 0.16 / 0.29	20 / 23 / 41 0 / 1 / 12
	Foggy	89 / 85 / 74 100 / 94 / 92	- / - / - 0.14 / 0.19 / 0.8	20 / 25 / 48 0 / 9 / 9	92 / 85 / 66 100 / 97 / 90	- / - / - 0.2 / 0.21 / 0.56	12 / 30 / 55 1 / 9 / 16
Sunset	Cloudy	94 / 92 / 88 100 / 100 / 100	- / - / - 0.14 / 0.14 / 0.16	14 / 26 / 30 0 / 0 / 1	95 / 91 / 90 100 / 100 / 100	- / - / - 0.1 / 0.12 / 0.17	10 / 19 / 20 0 / 0 / 0
	Rainy	92 / 86 / 66 100 / 99 / 97	- / - / - 0.14 / 0.18 / 0.21	10 / 20 / 49 0 / 3 / 9	88 / 74 / 53 100 / 98 / 92	- / - / - 0.16 / 0.25 / 0.66	20 / 39 / 56 0 / 9 / 12
	Foggy	89 / 86 / 75 100 / 93 / 90	- / - / - 0.16 / 0.24 / 0.87	18 / 24 / 50 0 / 9 / 18	90 / 85 / 71 100 / 92 / 79	- / - / - 0.17 / 0.24 / 0.94	16 / 19 / 34 1 / 12 / 39
Night	Cloudy	83 / 83 / 82 100 / 94 / 93	- / - / - 0.24 / 0.26 / 0.28	25 / 32 / 30 1 / 6 / 9	81 / 80 / 76 100 / 95 / 94	- / - / - 0.21 / 0.24 / 0.26	26 / 29 / 35 1 / 8 / 9
	Rainy	84 / 73 / 66 98 / 90 / 76	- / - / - 0.24 / 0.3 / 0.53	23 / 37 / 48 3 / 12 / 30	88 / 85 / 62 99 / 92 / 73	- / - / - 0.21 / 0.24 / 0.69	19 / 25 / 49 4 / 10 / 36
	Foggy	85 / 76 / 61 98 / 84 / 85	- / - / - 0.25 / 0.58 / 0.81	29 / 34 / 52 8 / 20 / 31	90 / 71 / 43 99 / 81 / 65	- / - / - 0.28 / 0.43 / 0.98	24 / 40 / 71 8 / 24 / 49

Table 3.1: Infraction analysis across various operating conditions. The testing results of the two agents are grouped in two rows at each cell. The upper one corresponds to CIL and the CAL below. Each row's results are organized based on the qualitative order (light/medium/heavy).

the perception system and the E2E ADS work. Using the proposed framework, the overall operating conditions of an ADS can be found, and the violations can be recorded for further development and ODD expansion.

However, it should be noticed that the overall automotive safety problems related to ODD addressed in [186, 61] still have many open challenges. Often, the test needs to be done more than once to cover enough environmental conditions. The challenge of performing as many tests as possible based on quantifying parameters is an indispensable challenge in automated driving safety. This challenge is still achievable in simulation environments, often through hardware acceleration and multi-node parallelism. Furthermore, validation solely in the simulation environment cannot cover the real driving scenario since there are always domain differences. Finally, it is recondite to achieve the claim of 'absolute safety". Instead, claiming that the multiple ADS function always satisfies the operational domain specifications would be much more implementable.

For future work, it is interesting to include the RSS model [46] in the validation framework. Currently, the benchmark requirement used in this work for the extracted op-

erational domain is a contract violation rate of less than 5%. The resulting ODD is a static boundary set for each environmental condition without considering its probability distribution, which could result in more false alarms (passive behavior). It is possible to encode the ODD condition as a conditional probability table, which can then be processed to determine whether the ADS function works dynamically in the given ODD.

Chapter 4

ODD Augmentation

The increase in autonomy level relies on expanding the ODD range of the autonomous driving functions. To expand the scope of ODD, it is crucial to understand the defects of the current system and boost its performance. Specifically, in perception systems, as one essential step for ADS to understand the environment, performance degrades when facing unseen variations. This chapter attempts to expand such a data-driven subsystem in ADS through scenario data augmentation and robust training. The main goal is to make the given model more resilient to difficult natural changes that the perception system couldn't deal with well before. Boosting the model's robustness towards harsh weather and bad light operation conditions helps extend the ADS' ODD to further cover those semantic operation domains.

4.1 Problem Description and Preliminaries

The system's ODD excludes situations in which the specific components exhibit diminished performance that impacts driving safety. Thus, the ODD augmentation explores the situation that the current version of ADS's operational domain cannot cover well. Because downstream decision-making, planning, and control will rely heavily on accurate estimation of the surrounding driving environment, the operation domain of the perception modules will be the cornerstone for the overall ODD of the ADS. Thus, in this chapter, the augmentation for the ODD of the perception models based on validation results of previous version development is investigated.

The perception model, f_{PM} takes the environment observation and maps it to the states of the interpretable driving situations, such as scenario classification [216], object detection [217] and tracking [117]. These models are usually obtained with a data-driven strategy that trains a neural network with labeled data sets. The validation process examines the operating domain of the perception model f_{PM} under some criteria Φ as specified in the previous chapter to provide a fixed operating domain Π_Φ similar to the remained sets of not behaving well, denoted by $\Pi_{\neg\Phi}$. Both Π_Φ and $\Pi_{\neg\Phi}$ contain a semantic description of the operational domain, where the $\Pi_{\neg\Phi}$ corresponding to the conditions that the current f_{PM} have performance lower than the predefined criteria. The overall goal of ODD augmentation is to boost the performance of the f_{PM} under the operating conditions described in $\Pi_{\neg\Phi}$ while maintaining good robustness on the original domains. In this chapter, the image-based perception system f_{PM} is formulated by the following ODD augmentation problem.

Problem 4.1.1 (ODD Augmentation)**.** *Given the fixed f_{PM} and its challenging operating conditions described in a semantic set $\Pi_{\neg\Phi} = \{s_1, s_2, ..., s_k\}$, and some known anchor data distribution $\mathcal{D}$, it is known fixed f_{PM} performed well in anchor data distribution such that $x \models \Phi,\ \forall x \in \mathcal{D}$. The $\mathcal{D}_{s_1}$, ... $\mathcal{D}_{s_k}$ are distributions that are generated by natural perturbation based on the anchor data (same content but different texture in image). It is also known that f_{PM} does not meet the required level of performance from previous validation, e.g. $x \not\models \Phi,\ \exists x \in \mathcal{D}_s$. The goal is to boost the performance of f_{PM} in the natural perturbed distribution described by $\mathcal{D}_{s_1}$, ... $\mathcal{D}_{s_k}$.*

The Problem 4.1.1 can be divided into two parts, one is how to obtain the unseen/uncovered distribution described by the semantic sets in $\Pi_{\neg\Phi}$, which direct to a data augmentation problem. The other one is how to train the existing model properly that boost its robustness to the given perturbations. In this thesis, the assumption is that perturbations are natural variations rather than the adversarial attacks mentioned in [170, 174].

4.2 Methodology

4.2.1 Transfer Learning

The ODD augmentation problem can be formulated as a transfer learning problem for the existing model towards a specific distribution. Transfer learning aims at improving the

model performance by transferring the knowledge across domains [218]. Consider there is a perception task that aims to map $x \in \mathcal{X}$ to $y \in \mathcal{Y}$. A typical example task on image classification would have $\mathcal{X} = \mathbb{R}^{W \times H \times C}$ and $\mathcal{Y} = [k] = \{1, 2, 3, ...k\}$ where W, H and C corresponds to the image width, height and channels, and the k corresponds to the total of k categories in this task. The total input dimensions are represented by $d = W \times H \times C$. The target space may get more complicated as the task difficulty increases, for example, the output is formulated as a list of bounding boxes in the format with x and y coordinate of the bounding box anchors, the bounding box width, height, and the corresponding class in the object detection task. The perception model f_{PM} with weights $w \in \mathbb{R}^p$ is trained on fixed data distribution (x, y) D based on some loss function $l(x, y; w)$. Here the notion $w \in R^p$ denotes the weights in the parameter space of f_{PM}. And a suitable loss function should be highly correlated with the validation metrics mentioned in the previous chapter, such as MAE or cross-entropy.

The commonly used machine learning approach solves the following problem with knowing the training data represent the expected distribution in the deployment stage [219].

$$w^* \in \arg\min_{w \in R^p} E_{(x,y)\sim D_{train}}[l(x, y; w)]. \tag{4.1}$$

Literature often uses first-order methods such as Stochastic Gradient Descent (SGD) based methods [220, 221] to approximated solve (4.1) to obtain weights. In our problem setting, f_{PM} with w^* failed to pass the requirement in the validation data or in general a shifted distribution D', that is

$$trace[f_{PM}(x), y]_{D'} \not\models \Phi. \tag{4.2}$$

The trace corresponds to the sample sequence from D' that feeds to validate the system performance. In the image classification case, if both the loss function and evaluator are selected as the MAE, e.g. $\Phi.\rho(x, y; w) = l(x, y; w) = MAE$, then this performance insufficiency in (4.2) can be expressed in the similar format in (4.1) as

$$E_{(x,y)\sim D'}[l(x, y; w)] > \Phi.\epsilon > E_{(x,y)\sim D_{train}}[l(x, y; w)] \tag{4.3}$$

The error bound for the requirement is denoted as $\Phi.\epsilon$ as defined in the previous chapter. Thus, the transfer learning problem is

Problem 4.2.1 (Transfer Learning after Testing)**.** *Given $f_{PM}(w^*)$ developed in some previous distribution D, and the performance is validated under some new input distribution D' and there exists performance insufficiency as (4.3). The goal is to find following*

$$w'^* \in \arg\min_{w \in R^p} E_{(x,y)\sim (D' \cap D)}[l(x, y; w)]. \tag{4.4}$$

4.2.2 Robust Learning to Natural Perturbation

There are infinite potential combinations of the new distribution when considering any possible natural perturbations. Typically the distribution D' is down-sampled compared to D such that the expectation term in (4.4) is difficult to estimate. In the ADS development phase, the natural perturbation in the inputs of the perception systems is investigated, such as the lighting condition or adverse weather, e.g., layer 5 in Fig. 3.2. Instead of formulating the adversarial perturbations as in many computer vision studies [222, 223], it is assumed that there exists a model representing the natural perturbation $G : R^d \rightarrow R^d$ with a semantic nuisance parameter δ such that

$$x' = G(x, \delta), \quad \delta \in R^c, \quad c \ll d \tag{4.5}$$

where the input $x \sim D$ which is transformed into a geometrically and semantically similar data x' that follows a new distribution D' by varying the nuisance parameter δ. As shown in Fig. 4.1, the semantic nuisance parameter δ controls the perturbation level of the natural variation. The constant c corresponds to the dimensions to explore in the perturbation space related to the semantic directions of the perturbation.

The data distribution D' mentioned in (4.4) in our setting is now generated by the natural perturbation model depicted in Fig. 4.1. The natural perturbation model can be defined explicitly, using physical-based rendering methods in [224], or using a black-box model such as using the Unreal Engine [225], or image-to-image translation based on neural networks [226]. The input data can be perturbed based on multiple known models such that $x' = g_1(g_2(...g_m(x, \delta_m), \delta_2), \delta_1))$ where the same notion $x' = G(x, \delta)$ is being used for simplicity.

Before exploring the robust learning to natural perturbations, here list the terminologies for examining the model robustness.

Anchor Data: corresponds to the original data instance in the dataset, $x \sim D$, which is image in this study.

Neighbour Set: corresponds to the perturbed data by the model G based on the anchor data. There can be an infinite number of neighbour data generated from the anchor, however, the distance between neighbours and the anchor data can be measured using the nuisance parameter δ. Then the set of neighbours bounded by some factor ϵ is given by

$$\Delta(\epsilon) = \{x' = G(x, \delta) \in R^d : \delta \leq \epsilon\}. \tag{4.6}$$

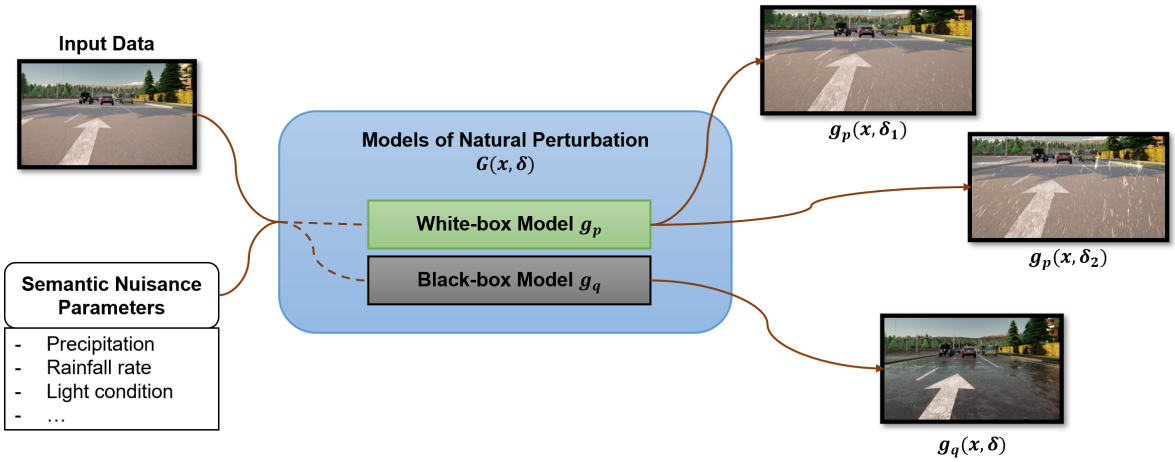


Figure 4.1: Models of natural perturbation. The model of natural perturbation creates a variation of the input data based on the semantic nuisance parameters, $x' = G(x, \delta)$. The perturbation model can be white-box with known rules g_p, black-box model g_q, or a composite of a series connection of the models $G = g_1(g_2(...g_m(\cdot)))$. In our paradigm, the model of natural perturbation directly uses the G notion and nuisance parameter $\delta \in R^c$ with c sets of perturbation direction.

For larger δ, the difference between x and x' should be larger. Notice that this distance can be measured only in one certain semantic dimension in the current study, which is quite different from the typical studies treating the perturbation as the norm-ball where the distance can be computed and compared using p-norm [222]. In our case, the neighbours generated using $\delta = [\text{light} : 10, \text{rain} : 20]$ does not necessarily generate a more perturbed neighbours based on $\delta = [\text{light} : 20, \text{rain} : 1]$ even though the p-norm is larger. It is only meaningful when comparing the distance in one specific semantic direction.

Neighbour Accuracy/Loss: is the model accuracy evaluated in the neighbour set of a given data anchor. The neighbour accuracy is then the ratio between the number of samples that are correctly classified and the total number of samples selected from the neighbour set $\Delta(\epsilon)$, i.e.,

$$\frac{1}{K}\sum_{i=1}^{K} \mathbb{1}[f_{PM}(G(x, \delta_i)) = y], \ \delta_i \leq \epsilon. \tag{4.7}$$

Notice that (4.7) forms negative of basic MAE loss function for the neighbours in the classification task. In fact, the equation above can be generalized into the batch loss

by substituting the inner part of (4.7) to any loss function considering the samples from neighbour set as a fixed batch, for example, the cross-entropy

$$\xi(f_{PM}, x, \epsilon) = \frac{1}{K}\sum_{i=1}^{K}[-y\log(f_{PM}(G(x, \delta_i)))], \ \ \delta_i \leq \epsilon, \tag{4.8}$$

where the label y is the correct classification and the model f_{PM} outputs the predicted probability for the class. To sum up, $\xi(f_{PM}, x, \epsilon)$ can express the neighbour loss in general for fixed data

$$\xi(f_{PM}, x, \epsilon) = E_{(x,y)\sim\Delta(\epsilon)} l(x, y; w). \tag{4.9}$$

Definition 4.2.1 (Data-wise Robustness)**.** model f_{PM} is **robust** to the perturbation specified by δ at the certain anchor data point x if the neighbour loss $\xi(f_{PM}, x, \epsilon)$ is smaller than a predefined threshold.

This robustness idea can be applied to the fixed input data in a similar manner, that the input data is strong/robust for the f_{PM} under evaluation under the fixed semantic augmentation direction. Based on this formulation, **data-wise robustness** by sampling the instance in the neighbour set and comparing the neighbour loss with the threshold.

Moreover, the robustness idea can be further applied to given data distribution. That is, computing the summation of neighbour loss for N samples in distribution D as following

$$\sum_{i=1}^{N}\xi(f_{PM}, x_i, \epsilon) = \sum_{i=1}^{N} E_{(x,y)\sim\Delta(\epsilon)} l(x_i, y_i; w), \ \ (x, y) \sim D. \tag{4.10}$$

Notice that the above expression reflects the robustness of given distribution D to the perturbation specified by δ for the model f_{PM}, the larger value corresponds to the weaker robustness facing the perturbation. Furthermore, (4.10) provides us a closed form for the inner part expression of (4.4), that is

$$\begin{aligned} E_{(x,y)\sim(D'\cap D)}[l(x, y; w)] &= E_{(x,y)\sim(D)}[l(G(x, \delta), y; w)] \\ &\approx \sum_{i=1}^{N}\xi(f_{PM}, x_i, \epsilon) \\ &= \sum_{i=1}^{N} E_{(x,y)\sim\Delta(\epsilon)} l(x_i, y_i; w). \end{aligned} \tag{4.11}$$

This equation above provides a measure of the **distribution-wise robustness** for the given model and a fixed perturbation direction. The distribution-wise robustness can then be determined by comparing (4.11) with a predefined threshold, or utilizing the performance drop comparing with the normal performance in the original distribution, which is defined as follows,

$$\varrho = E_{(x,y)\sim(D)}[l(G(x,\delta),y;w)] - E_{(x,y)\sim D}[l(x,y;w)]. \tag{4.12}$$

In fact, the easy or hard perturbation direction can be obtained by iterating each type of perturbation with the fixed f_{PM} and D, this effectively examine the **model sensitivity** towards various perturbation. The worst-case perturbation in the neighbour set for all samples of the given distribution can be expressed as

$$\delta^* \in \arg\max_{\delta\in[0,\epsilon]}[l(G(x,\delta),y;w)]. \tag{4.13}$$

The transfer learning problem defined in Problem 4.2.1 can now transform into a robust learning problem to the natural perturbation defined following

Problem 4.2.2 (Robust Learning to Natural Perturbation)**.** *Given neural network model $f_{PM}(w^*)$, known distribution D, perturbation model G and a set of semantic nuisance parameters δ, the robust learning problem given the known perturbation solves*

$$\min_{w} E_{(x,y)\sim D}[\max_{\delta\leq\epsilon} l(G(x,\delta),y);w)]. \tag{4.14}$$

In fact, the optimization problem in (4.14) comprises of an inner maximization problem illustrated in (4.13) and an outer minimization problem

$$w'^* \in \arg\min_{w\in R^p} E_{(x,y)\sim D}[l(G(x,\delta^*)),y;w] \tag{4.15}$$

In the inner optimization problem (4.13), the goal is to search for nuisance parameters in the given semantic perturbation space that can produce a high loss/low accuracy among the data instances and their neighbours. If the search space of (4.13) extends to a vector of multiple perturbation directions, for example

$$\delta = \begin{bmatrix}\delta_1 & \delta_2 & \dots & \delta_m\end{bmatrix}, \text{ where } \delta_i \in [0,\epsilon_i],\ i = 1,...,m; \tag{4.16}$$

then, the optimal nuisance parameter δ^* characterizes the most-challenging perturbation in the given search space that the current model f_{PM} can handle for D' that is captured

by G and D. The outer optimization problem (4.15) seeks the best weight w in the neural network parameter space that minimizes the risk against the perturbation of the form $G(x, \delta^*)$. Ideally, solving the inner and outer optimization problem together should make the model with new weights $w'*$ invariant to the perturbation generated by model $G(\cdot, \delta)$ for any perturbation parameter bounded in the search space $\Delta(\epsilon)$. That is, the model is ϵ-robust to the natural variation G for the anchor data D.

4.2.3 Perturbation Models

Solving (4.13) and (4.15) both rely on the existence of the perturbation model G. This section addresses the perturbation model similar to the image augmentation schemes used in [227] and discusses the two cases depicted in Fig. 4.1. In general, two synthesizing methods on images can be used: 1) using a white-box model such as physics-based rendering that explicitly computes the dynamics and radiometry of weather effects (rain, fog) in images [227, 228]; 2) black-box model such as Unreal Engine [225], domain adaptation [229], when there is no explicit form of G. The expressiveness and effectiveness of the generated perturbation will be examined.

White-box model

Perturbation Model for Brightness and Contrast: There is various white-box model for perturbation that is designed for improving model generalization on computer vision, such as the imgaug library [230] which comprises a wide range of augmentation techniques such as tuning the brightness, contrast, and applying noise directly to the image. Let the input clear scene radiance image to be $R(\varkappa)$, where $\varkappa$ corresponds to the source image pixels, the brightness and contrast are often tuned using the following model

$$I(\varkappa) = \alpha_1 R(\varkappa) + \beta_1, \tag{4.17}$$

where $\alpha_1 > 0$ and β_1 are called the gain and bias parameters which control the contrast and brightness respectively. The generated image $I(\varkappa)$ is effectively the x' in the following form

$$x' = G(x, \delta) = \delta_{(1)} x + \delta_{(2)}, \ \delta = [\alpha_1, \beta_1] \tag{4.18}$$

where the number inside the brackets corresponds to the index of vector and the overall search space for δ is confined in a rectangular in R^2-space.

Perturbation Model for Foggy Weather: In addition to normal contrast and brightness, the physics-based synthesis methods are investigated for foggy and rainy situations, which are reported to impact the perception performance in a recent review [231]. The atmospheric scattering model is considered for this task as it is used extensively in studies related to image dehazing and hazy image rendering [232, 233, 234] with the following formulation.

$$I(\varkappa) = T(\varkappa)R(\varkappa) + A[1 - T(\varkappa))]. \tag{4.19}$$

The air-light vector A in (4.19) is assumed to be a constant globally following evidence showed in [235] which generally holds for daytime images. The transmission map $T(\varkappa)$ corresponds to the relative scene radiance that manages to reach the camera. Notice that (4.19) is in the similar form as (4.18) with the $T(\varkappa)$ and $A[1 - T(\varkappa))]$ terms are matrix project in every pixel, e.g. in the R^d space rather than a readable parameter in (4.18). In fact, the transmission map can be rewritten as follows based on the homogeneous medium assumption

$$T(\varkappa) = e^{-\beta_2 d(\varkappa)}. \tag{4.20}$$

The parameter β in (4.20) is the medium extinction coefficient that controls the generated fog thickness [236]. In this case, the perturbation model G for foggy weather can be explicitly written in the form of (4.19) combined with (4.20) that is controlled by the semantic parameter $\beta_2 > 0$.

The typical foggy image synthesize based on the physical model is a two-step task: (1) estimate the transmission map in (4.20) with either depth estimation [237] or using depth map if available with the preset nuisance parameter; (2) substitute the estimated $T(\varkappa)$ and input $R(\varkappa)$ to obtain the final perturbed image based on (4.19). The first step could be trickier compared to the second step due to the requirement of depth estimation. Notice that the detailed implementation of the physics-based rendering is not the focus of this thesis. The strategy proposed in [236] is used for those single image depth estimations. The method presented in [238] for those cases in that a noisy, incomplete estimate of depth is available.

Perturbation Model for Rainy Weather: Adding synthetic rain can be considered as adding a rain layer to the existing image

$$I(\varkappa) = R(\varkappa) + I_{rain}(\varkappa). \tag{4.21}$$

A simplified version of [228] is deployed where the raindrop layer is obtained in the following steps. First, the rain streaks s are selected from the proposed database of [20] and warped

using the physical simulator mentioned in [239]. In this case, the rainfall particle size distribution, raindrop size, and velocity are directly employed based on the models proposed in Table 1 of [239], with a parameter $\delta_{rain} \geq 0$ controlling the rainfall (precipitation) rate (mm/hr). Especially, the rain particle distribution is computed based on

$$N = 8000 * e^{-\delta_{rain}^{-0.21} * \text{diameter}}, \tag{4.22}$$

in [239], where the raindrop diameter are sampled from $0.1 - 10$ mm and (4.22) provides the number of particles in 1 m^3. Finally, based on the raindrop dynamics from [239], the perturbation rain layer is warped from the selected rain streaks s with a homography $\mathcal{H}(\cdot)$, e.g.

$$I_{rain}(\varkappa) = \mathcal{H}(s), \tag{4.23}$$

with the distribution follow (4.22). In this case, larger δ_{rain} corresponds to the more frequent appearance of the rain streaks in $I_{rain}(\varkappa)$, e.g. more perturbation noise.

Black-box model

The physics-based synthesis methods can provide intuitive ways to generate natural perturbations with semantic nuisance parameters compact to the physical meanings to tune the perturbation. However, in many situations, the white-box model may not be accessible or too difficult to implement explicitly. For example, the depth map is required in (4.20) for fog weather generation, however, in many cases, the accurate depth estimation is not accessible which leads to problems using the white-box model. Also, even though there is a relationship between the natural perturbation and the baseline data, such as sunny-snowy, day-night; the explicit formulation of G that models such perturbation generation is problematic. For these cases, the black-box models are investigated in the following.

Domain Adaptation: In this part, the domain adaptation approach is used to obtain the black-box perturbation generation model G. Domain adaptation type of approach is particularly suitable for the transfer learning problem mentioned in 4.2.1, especially when faced with limited unlabelled target domain data [240]. Recent progress such as [7, 241, 242] showcases the strategy to learn the image-to-image translation using conditional Generative Adversarial Networks (cGAN) and able to generate various semantic variations using the same network structure.

Weather Configuration in Unreal Engine: Modern simulation engine such as Unreal [225] and Unity [243] has been widely applied in reconstructing a parallel scene

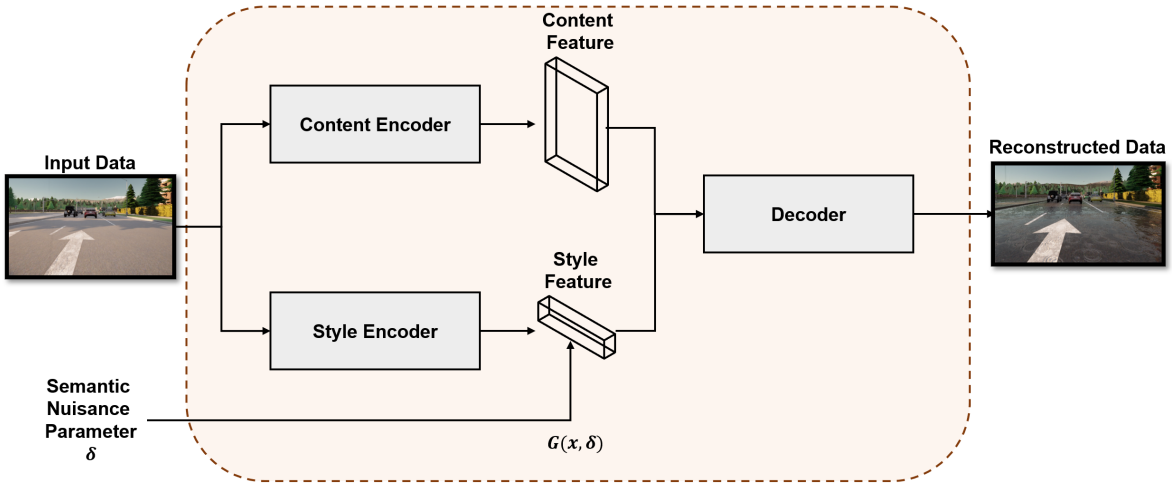


Figure 4.2: The general architecture for generating natural perturbation using a learning model. It is assumed that the input data and generated perturbed data can be mapped into the same content space as [7].

or scenario with high fidelity in recent autonomous driving studies [244]. In this study, CARLA with Unreal Engine is utilized to produce the natural perturbation for the expansion of the validation sets. The perturbation data is generated using the interface provided by CARLA which provides a diverse set of factors such as precipitation, daytime, and fog intensity [189].

Discussion

The white-box model perturbation in (4.17), (4.19) and (4.21) generally provides larger norm-distance in pixel level for larger semantic nuisance parameter (fixed in one direction), which align with typical adversarial robustness studies (but with larger norm-ball to be considered) [223]. On the other hand, the norm distance between the original image and the black-box perturbed image does not hold to be larger for a larger δ. Moreover, the larger δ does not necessarily correspond to the bad performance of the prediction model or a large loss (4.11) over the data. In fact, the inner maximization task defined in (4.13) still needs to be solved to search for the worst perturbation case to properly address the robust learning problem 4.2.2.

4.2.4 Robust Training for Model-based Perturbation

In the previous sections, a learning algorithm robust to the natural perturbation problem has been formulated to solve the inner and outer optimization problem (4.13), (4.15) corresponding to some known perturbation model G. This section focuses on the procedure to robustify the model f_{PM} to the natural variation.

To proper address Problem 4.2.2, the assumption that the finite base dataset $\mathcal{D}_n = \{(x^i, y^i)\}_{i=1}^n$ sampled from the anchor data distribution $\mathcal{D}$ is accessible is taken. The perturbation model G is also given as a known priori for the operational domain to be extended on. For example, the anchor data could be clear images sampled from clear sunny days whereas the target domain could be rainy or foggy which the current model does not perform well on. By reformulating (4.14) over the finite base dataset, the goal is to solve

$$w^* \in \arg\min_{w \in R^p} \frac{1}{n} \sum_{i=1}^{n} \max_{\delta \in \Delta(\epsilon)} l[f_{PM}(G(x^{(i)}, \delta)), y^{(i)}; w]. \tag{4.24}$$

The primary goal is to learn optimal weights that parameterize the neural network model f_{PM}. However, the inner maximization and outer minimization problems are non-convex and the locally optimal solution is found by first-order optimization techniques [245].

Robustify with Model-based Augmentation: Following the data augmentation idea [246], the perturbation model generates perturbed samples that mimic the target domain that the original model has never seen before. Hence during training, the base dataset $\mathcal{D}_n$ can be augmented with the model G and the search space for semantic nuisance parameters $\delta \in \Delta(\epsilon)$.

The proposed strategy is given in Algorithm 1. For each batch, the sample data are augmented for k-times that generate the k neighbours of the anchor data with the generation model G with perturbation bounded by $\Delta(\epsilon)$. The total batch loss is the sum of the batch loss of batch anchor samples and the loss from the k neighbours. The Stochastic Gradient Descent (SGD) with the learning rate η for the solver is used.

Robustify with Model-based Worst-case Learning: Algorithm 1 attempts to expose the diverse model-generated data to the neural network model during training in order to improve the model's robustness. Algorithm 2 is proposed with the finding of the worst-case perturbation in each batch to guide the weight learning process.

k neighbours for each anchor data are generated in batch. Instead of calculating the total average of the neighbour loss for the batch as in Algorithm 1, the maximum loss for

Algorithm 1 Robustify with Model-based Augmentation (RMA)

Input: previous model $f_{PM}(w)$, base dataset $\mathcal{D}_n$, perturbation model G

Parameters: perturbation search space $\Delta(\epsilon)$, sample neighbor number k, learning rate η

Output: updated model $f_{PM}(w')$

1: **repeat**
2: **for** batch $\{(x^{(1)}, y^{(1)}), ..., (x^{(M)}, y^{(M)})\} \subset \mathcal{D}_n$ **do**
3: sample δ uniformly from $\Delta(\epsilon)$ for k times to obtain $\delta_1, ..., \delta_k$
4: **for** k steps **do**
5: $x_j^{(i)} = G(x^{(i)}, \delta_j)$ ▷ generate perturbation k times for each sample
6: **end for**
7: $\mathcal{L} = \frac{1}{M} \sum_{i=1}^{M} l(f_{PM}(x^{(i)}; w), y^{(i)})$ ▷ normal batch loss
8: $\mathcal{L}^{\delta} = \frac{1}{kM} \sum_{i=1}^{M} \sum_{j=1}^{k} l(f_{PM}(x_j^{(i)}; w), y^{(i)})$ ▷ perturbed batch loss
9: $w = w - \eta \nabla_w (\mathcal{L}^{\delta} + \mathcal{L})$ ▷ update weights with SGD per batch
10: **end for**
11: **until** convergence
12: get final updated weights $w' = w$

each data point is found. $\mathcal{L}_{\max}^{\delta,(i)}$ is then the largest loss among all the perturbation sampled from $\Delta(\epsilon)$ based on the i-th anchor data point in batch. The corresponding δ is then the worst-case perturbation control for that data instance. In the Algorithm 2, the decision boundary is reshaped by updating the weights when the worst-case perturbation is on the wrong side of the decision boundary.

Weak Data Extraction: During the iterative development of the ADS function, it is expected to extract hard samples or edge cases for further extending the operational domain. The edge cases are used to further improve the robustness of the previous model. The training procedure of Algorithms 1 and 2 requires going through the entire base dataset by batch no matter whether the data sample is easy or hard for the previous model. In the transfer learning process, it is expected to expose the model to the hard samples and their neighbours such that the learned decision boundary will shift further away.

The Algorithm 3 is proposed to split the hard samples and easy samples from the base dataset. In line 4 of Algorithm 3, the k-neighbour accuracy is compared for each anchor. Notice that this expression can be converted to the loss computation based on (4.9). In fact, the inner term of perturbed batch loss of line 8 in Algorithm 1 is the sampled version of

Algorithm 2 Robustify with Model-based Worst-case Learning (RMWL)

Input: previous model $f_{PM}(w)$, base dataset $\mathcal{D}_n$, perturbation model G

Parameters: perturbation search space $\Delta(\epsilon)$, sample neighbor number k, learning rate η

Output: updated model $f_{PM}(w')$

repeat
  for batch $\{(x^{(1)}, y^{(1)}), ..., (x^{(M)}, y^{(M)})\} \subset \mathcal{D}_n$ **do**
    Initialize worst case perturbation loss per data $\mathcal{L}^{\delta}_{\max} = [0, ..., 0]$
    sample δ uniformly from $\Delta(\epsilon)$ for k times to obtain $\delta_1, ..., \delta_k$
    for k steps **do**
      $x_j^{(i)} = G(x^{(i)}, \delta_j)$ ▷ generate perturbation k times for each sample
    end for
    $\mathcal{L} = \frac{1}{M} \sum_{i=1}^{M} l(f_{PM}(x^{(i)}; w), y^{(i)})$ ▷ normal batch loss
    $\mathcal{L}^{\delta,(i)}_{\max} = \max l(f_{PM}(x_j^{(i)}; w), y^{(i)})$, for $j = 1, ..., k$ ▷ worst case loss per data
    $w = w - \eta \nabla_w (\frac{1}{M} \sum_{i=1}^{M} \mathcal{L}^{\delta,(i)}_{\max} + \mathcal{L})$ ▷ update weights with SGD per batch
  end for
until convergence
get final updated weights $w' = w$

(4.9) whereas the worst case loss per data $\mathcal{L}^{\delta,(i)}_{\max}$ in Algorithm 2 corresponds to the maximum version of (4.9). Thus, Algorithm 3 can be run along with the training procedure by simply conditioning on $\mathcal{L}^{\delta,(i)}$ each time.

Algorithm 3 Weak Data Extraction

Input: fixed model f_{PM}, base dataset $\mathcal{D}_n$, perturbation model G

Parameters: cutoff neighbor accuracy C_{na}, perturbation search space $\Delta(\epsilon)$, sample neighbor number k

Output: $\mathcal{D}_{weak}$, $\mathcal{D}_{strong}$

procedure WEAK DATA EXTRACTION
  for $(x, y) \in \mathcal{D}$ **do**
    sample δ uniformly from $\Delta(\epsilon)$ for k times to obtain $\delta_1, ..., \delta_k$
    if $\frac{1}{k}\sum_{j=1}^{k} \mathbb{1}[f_{PM}(G(x, \delta_j)) = y] \geq C_{na}$ **then** ▷ compare neighbour accuracy
      $\mathcal{D}_{strong} = \mathcal{D}_{strong} \cup (x, y)$
    else
      $\mathcal{D}_{weak} = \mathcal{D}_{weak} \cup (x, y)$
    end if
  end for
end procedure

4.3 Experiment

The experiment settings, metrics, and evaluation results are presented in this section. First, in Sec 4.3.1, the datasets and metrics used in this chapter are elaborated. Next, the base model sensitivity under various data and perturbations are examined, including contrast, fog, and haze on various virtual and real datasets on classification and detection tasks. The model sensitivity is related to the model robustness when facing the same natural perturbations. Based on the sensitivity analysis and the data characteristics, finally evaluate proposed robust training methods for ODD augmentation in Sec. 4.3.4. The robust learning performance is evaluated in known distribution and the unseen shift. The proposed training processes are compared with recent norm-ball-based adversarial training [247] as well as data augmentation-based training procedure [248]. The effectiveness of training strategies and the impact of the exploration space settings are evaluated. The training details and parameter settings for all models can be found in Appendix A.4.

4.3.1 Datasets and Metrics

In the experiment, a variety of datasets are compared, including CURE-TSR [8], CURE-TSD [249], and sampled synthetic data from CARLA [4] for evaluation. Both white-

box and black-box-based are examined perturbation on the datasets above. In Fig. 4.3, the signage is perturbed with contrast, brightness, and fog, respectively, with increasing semantic nuisance parameters.

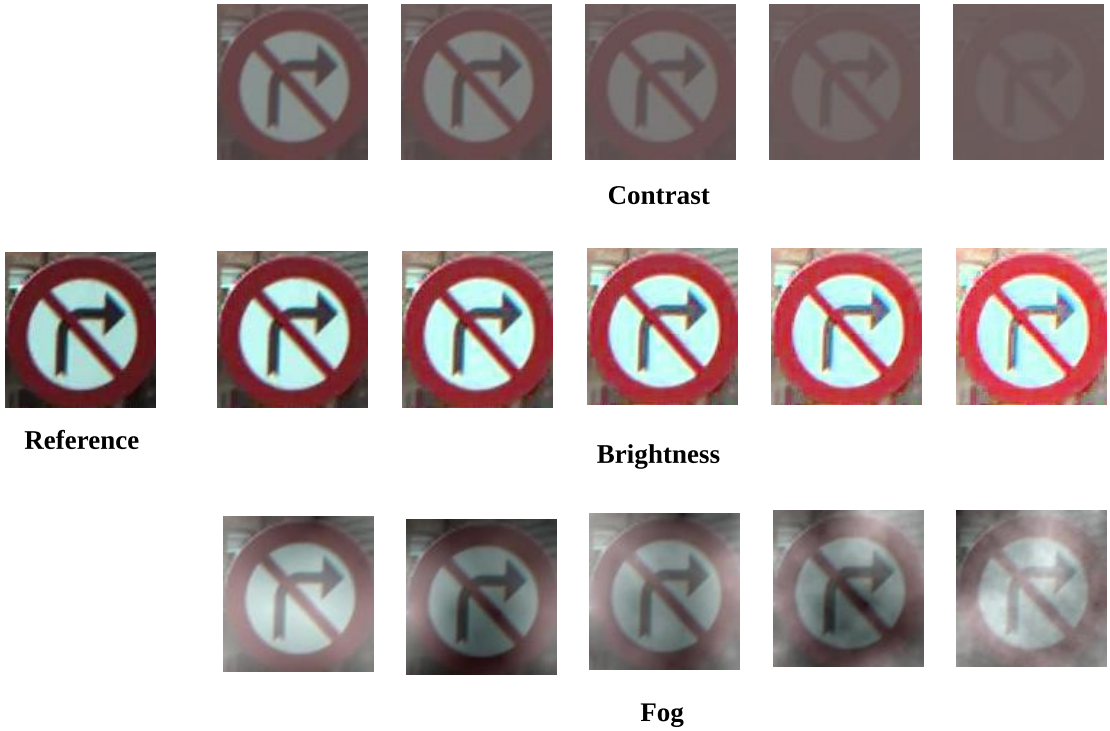


Figure 4.3: Reference traffic sign image from CURE-TSR [8] with various white-box based perturbations by increasing δ.

The CURE-TSR [8], CURE-TSD [249] datasets also have provided images with various challenges as shown in Fig. 4.4 and Fig. 4.5. The CURE-TSR consists of over 1.7 million real-world signage images, including speed limit, stop sign, hump, yield, etc. The images are labeled with different perturbation schemes and challenge levels. The CURE-TSR [8] is utilized to examine the classification task whereas the CURE-TSD [249] dataset is mainly used for the detection task. Compared with the white-box-based perturbation, the challenge level in black-box perturbation does not have a directly projected nuisance parameter δ.

In addition to the CURE-TSR/TSD dataset, 500 reference images in the CARLA simulation environment are sampled as additional data. And based on the reference samples, a

Figure 4.4: Example reference image and the perturbed ones provided in CURE-TSR [8] with challenge levels from 1 to 5.

Figure 4.5: Example images from CURE-TSD with challenge levels 1 to 5 (from left to right). The top row corresponds to the dark perturbation, the middle is associated with rain and the bottom row is the haze effect.

black-box based perturbation is deployed by adding fog, rain and changing the sun altitude and azimuth through the CARLA interface as shown in Fig. 4.6.

Figure 4.6: Example images CARLA [4] in foggy (top), sunset and night (middle), and the rainy (bottom) weather perturbation.

In the following experiment, 60% of data are used for training and the rest for evaluation without specific notice. It should be noted that the trained perception models never have access to the test data during the training phaase.

There are several widely-accepted metrics for classification and detection tasks such as accuracy, precision, and recall [250]. The aforementioned metrics are defined as the following. Given the True Positive (TP) as the total number of correct detections and False Positive (FP) as the total number of false detections, the precision is defined as $\frac{TP}{TP+FP}$ and recall is $\frac{TP}{TP+FN}$. The detection is considered correct using the Intersection over Union (IoU) threshold of 0.7 with respect to the ground truth bounding box.

On the other hand, the change of these metrics to measure the robustness of the perception models is evaluated. The change is evaluated in two directions, the first one is the

change with respect to the different data but the same model,

$$\varrho^{\Delta(\epsilon)}_{accuracy} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[f_{PM}(x_i) = y_i] - \frac{1}{kN}\sum_{i=1}^{kN} \mathbb{1}[f_{PM}(G(x_i, \delta_i)) = y_i]. \tag{4.25}$$

The change in accuracy above measures the robustness of fixed model f_{PM} with respect to the perturbation defined by G and $\Delta(\epsilon)$. The first term in (4.25) corresponds to the estimated performance of f_{PM} over distribution $\mathcal{D}$ whereas the second term corresponds to the estimated performance of f_{PM} over the perturbed neighbour set $\mathcal{D}_s$. The number k corresponds to the sample times in augmentation as defined in Algorithm 1 - 3. The change in metrics compared with before and after robustify procedure is also investigated based on

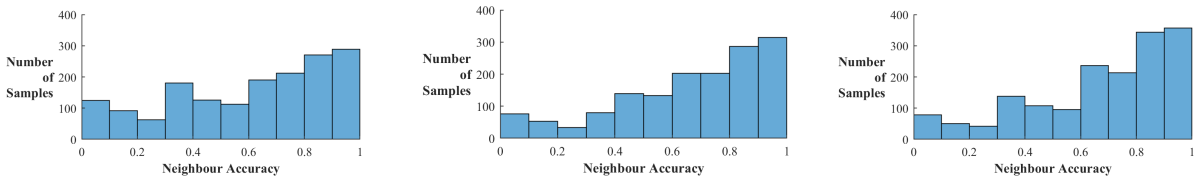
$$\varrho^{r}_{accuracy} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[f'_{PM}(x_i, \delta_i) = y_i] - \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[f_{PM}(x_i, \delta_i) = y_i], \tag{4.26}$$

where the two terms correspond to the estimated performance of the robustified model f'_{PM} and the previous model f_{PM} over the normal data distribution $\mathcal{D}$, respectively. Notice that the evaluations defined in (4.25) and (4.26) can be further extended to precision and recall as well based on the perception task.

4.3.2 Characteristics of Neighbour Accuracy and Robustness

The neighbour accuracy for perturbation for trained models is examined in this section. A subset of CURE-TSR dataset [8] is chosen that consists of 20,000 clear images and the white-box perturbation model for brightness perturbation as (4.18). Fig. 4.3 presents the sample clean and perturbed images.

The benchmark classification models are trained using the clean images with the VGG [251], ResNet [252] and GoogleNet [253] architecture respectively. All the benchmark models are trained from scratch using widely accepted hyper-parameters where the implementation details can be found in Appendix A.4. The clean image without any perturbations is fed into the models. The trained models were evaluated on the test set which consists of 1660 clean images as anchors. By letting the search space for the brightness perturbation $\delta \in [-10, 10]$ in (4.18) and sample in a uniform distribution by 20 times for each anchor image, the distribution of the neighbour accuracy can be obtained for each model among the test clean sample and the perturbation direction shown in Fig. 4.7.

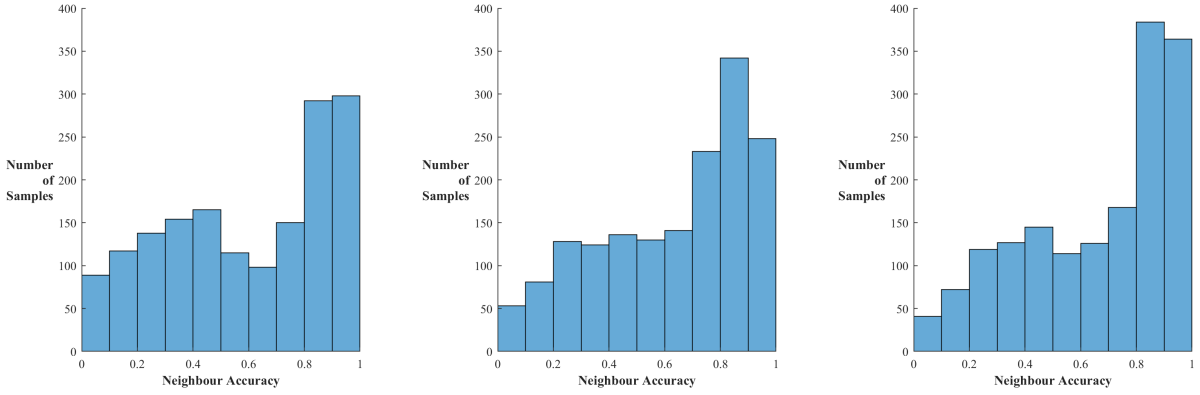


(a) VGG [251] (b) ResNet [252] (c) GoogleNet [253]

Figure 4.7: The histogram of neighbour accuracy for a subset of CURE-TSR [9] with respect to white-box based brightness perturbation among different benchmark models.

In Fig. 4.7, there are non-trivial data points with a relatively low neighbour accuracy subject to all the trained models and the fixed brightness perturbation. The reason is that all the models are trained with only clean images, which cannot handle properly when facing the natural perturbation since the validation set are not in the same distribution as the training set. In fact, all the models have over 90% classification on the clean images in the test set. It can be observed that 34% of the test data have a neighbour accuracy of less than 0.5 for the baseline VGG model in terms of the white-box brightness perturbation. The same finding holds for the other two baseline models. The ratio of the data points that have less than 0.5 for the other two models is around 25%. This implies that a large

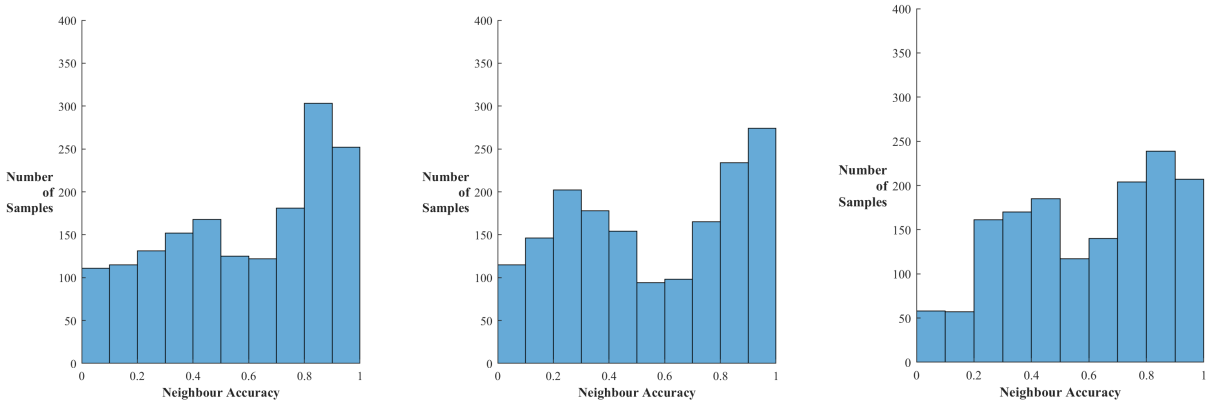
portion of data is **weak** to the given brightness perturbation. This portion will increase further if setting the required neighbour accuracy for **strong** points higher up to 0.8. The presence of **weak** data points corresponding to the specific semantic perturbation reflects the degradation of the model robustness. The observation in Fig. 4.7 shows that the neighbour accuracy can be a distinguishable measure for the robustness in terms of the specific natural perturbation direction as indicated in (4.7), given the trained model.



(a) VGG [251] (b) ResNet [252] (c) GoogleNet [253]

Figure 4.8: The histogram of neighbour accuracy for a subset of CURE-TSR [9] with respect to white-box based contrast perturbation among different benchmark models.

The neighbour accuracy under the contrast and fog weather variation are examined in Fig. 4.8 and Fig. 4.9 respectively. Similar to the brightness perturbation, the neighbor accuracy varies widely across data points for the given natural variation. The non-robust data points (neighbour accuracy less than 0.5) take about 40%, 32% and 31% among the three models for the contrast perturbation. Similarly, it can be observed that over 42%, 48% and 36% of the anchor data are identified as **weak** data points in the fog weather perturbation shown in Fig. 4.9. The models trained on clean images are considered fixed decision boundaries for any data points, and the neighbour accuracy reflects the local robustness of the natural perturbation. The weak data points are those close to the decision boundary, and the natural perturbation results in neighbours across the side, whereas the strong points are those further away from the decision boundary, and the neighbours stay on the same side for the given perturbation in the representation space. It can be concluded that the neighbour accuracy and its conjugate neighbour loss are distinguishable measures



(a) VGG [251] (b) ResNet [252] (c) GoogleNet [253]

Figure 4.9: The histogram of neighbour accuracy for a subset of CURE-TSR [9] with respect to white-box based fog weather perturbation among different benchmark models.

for model robustness related to both perturbation direction and the data point itself. In this case, to robustify the model in terms of particular semantic perturbation, the goal is to train the model such that the perturbation will not cross the model decision boundary, which agrees with the analysis in Sec. 4.2.4.

In addition to the characteristics of neighbour accuracy across different perturbations and models, the association between data points and the local robustness is investigated. First, three hundred weak data (neighbour accuracy less than 0.5) are sampled associated with the VGG model and fog perturbation. For each weak data, the corresponding neighbour accuracy is calculated for the other two models and perturbation directions. The final neighbour accuracy across models and perturbation on the selected weak data are associated in Fig. 4.10. It shows that most of the weak anchor data for VGG model on fog perturbation remain low neighbour accuracy in other models and perturbation direction. The observation in Fig. 4.10 indicates the strong correlation of the neighbour accuracy across model (trained on the same clean data) and the natural variations. In fact, the distribution of neighbour accuracy on different perturbations looks similar in Fig. 4.10. It can be further concluded that neighbour accuracy is more related to the data as a measure of local robustness. Different perturbations will affect the local accuracy, but not decisively. From the data distribution perspective, the model trained with clean data learns

a decision boundary that distinguishes the label from the representation of clean data. The data point and its perturbed neighbours form a new distribution in the representation space where the distance between the feature distribution and the model decision boundary finally reflects the local robustness. The correlation of the neighbour accuracy across the models is very strong, with a spearman correlation of 0.85. It is not surprising to find such a high correlation since the model is trained with the same clean data set with only the backbone difference.

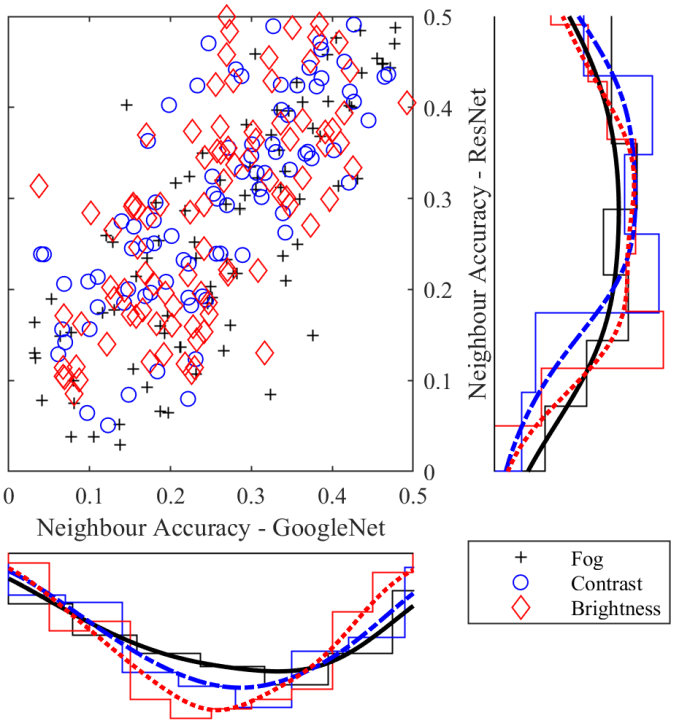


Figure 4.10: The scatter plot for the test performance of the ResNet and GoogleNet models (trained on clean image) selected weak data from the VGG benchmark.

Similarly, Fig. 4.11 presents the correlation for various models and perturbation direction for the strong data (with neighbour accuracy greater than 0.8) identified in the VGG benchmark. It can be observed that the characteristic of neighbour accuracy shares similar traits for strong data points as well. Even though there are many points drop off

in Fig. 4.11 since they are not considered as strong data in the other two models, it can be still observed that the strong correlation for the remaining ones across the two backbone structures (with a spearman correlation of 0.87).

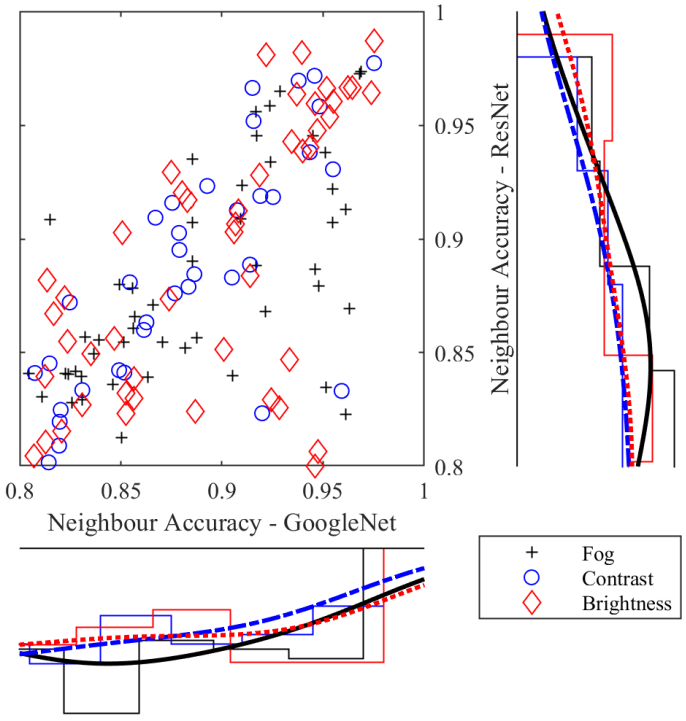


Figure 4.11: The scatter plot for the test performance of the ResNet and GoogleNet models (trained on clean image) selected strong data from VGG benchmark.

Fig. 4.10 and 4.11 indicate that the attributes of local robustness are shared across models with different backbone given the same training set. An image data with relatively low neighbour accuracy for one specific model and perturbation stays weak on the model with other backbones and perturbations. For example, it is easier to identify a forbid sign than a bicycle lane sign because the former has a much more straightforward representation in the feature space and is easier to identify than the latter.

In ODD augmentation, the fixed perception model f_{PM} can either have a structure update or a major update based on transfer learning on new data distribution. The results

in this section indicate that even with a structure update, the performance on the model robustness will not have a clear improvement given the same training data distribution. The ODD augmentation should focus on the training data distribution augmentation and target to obtain the decision boundary further away from the feature clustering in the representation space of both clean data and the perturbed ones.

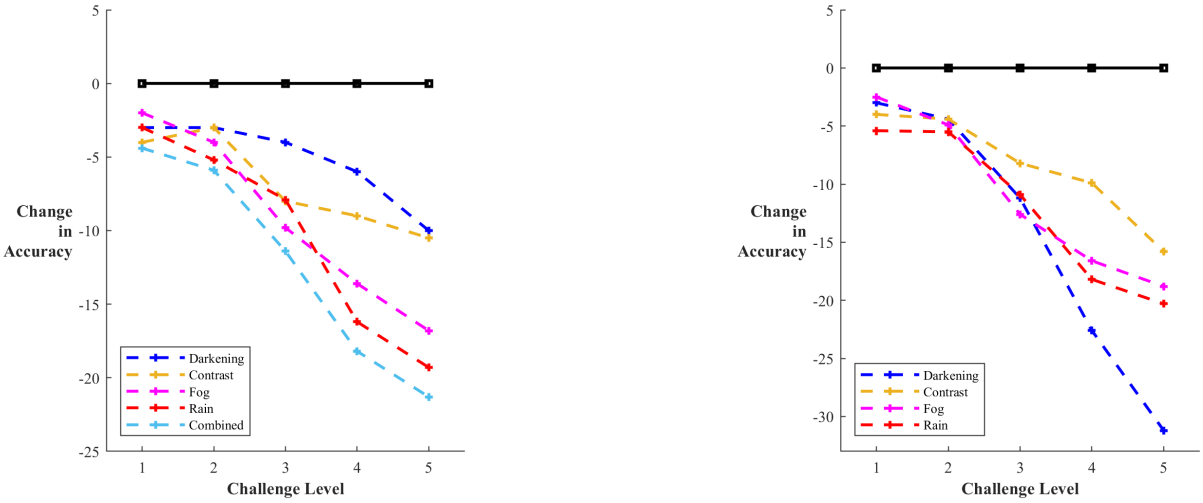
4.3.3 Sensitivity to Natural Perturbations

In this section, the sensitivity to different natural perturbations is examined by comparing it with the reference performance. The GoogleNet [253] backbone is used to train a standard reference model using the clean data as well as the Level 1 in CURE-TSR data [8]. The reference performance shows a test accuracy of 93.13% on the clean data. The robustness of the model is measured by the change in accuracy in terms of different natural perturbations. This accuracy change also represents the model's sensitivity corresponding to different types of perturbations.

The performance drop due to various perturbations on the clean data is evaluated on the same type of model, e.g., the previous perception model trained with clean images without considering any natural perturbations is shown in Fig. 4.12. Results show a significant drop in accuracy when evaluating the subset with natural perturbation since the tested dataset is shifted from the clean data due to the natural perturbation mapping G. The trained model never saw such shifted distribution in the training phase. Based on the results in Fig. 4.12, the model classification performance dropped significantly in a linearly-quadratic fashion. The white-box model generates the natural perturbation by randomly sampling the nuisance parameter in a predefined range to control the challenge level. For example, $\delta \leq 5$ is considered to be the level 1 challenge, $5 < \delta \leq 10$ for level 2, and so on for the white-box-based brightness (darkening) perturbation. Similar search spaces are defined for fog, contrast, and rain. The combined version takes the uniform sample of all the perturbations mentioned above space and processes the image data in a sequence of natural variance generation model G.

In the white-box natural perturbation category (Fig. 4.12 (a)), all perturbations have a clear drop in accuracy after challenge level-3. The darkening and contrast perturbation has less performance drop than fog, rain, and combined cases. It can be noticed that there is similar degradation of performance after challenge level-3 in 4.12 (b). However, the model performance is much more sensitive to the CURE dataset's darkening factor than the white-box model's perturbation. This is partly because the unknown black-box model generated for CURE dataset is different from the white-box model. Another more important reason is the difference in terms of the "challenge level" between the two. From our eye inspection, the image generated from the white-box darkening process of challenge level 4-5 is at the same level as 2-3 in CURE dataset.

The effect of natural perturbation types over the perception model performance is further examined on detection tasks compared to the relatively easy ones. The input of

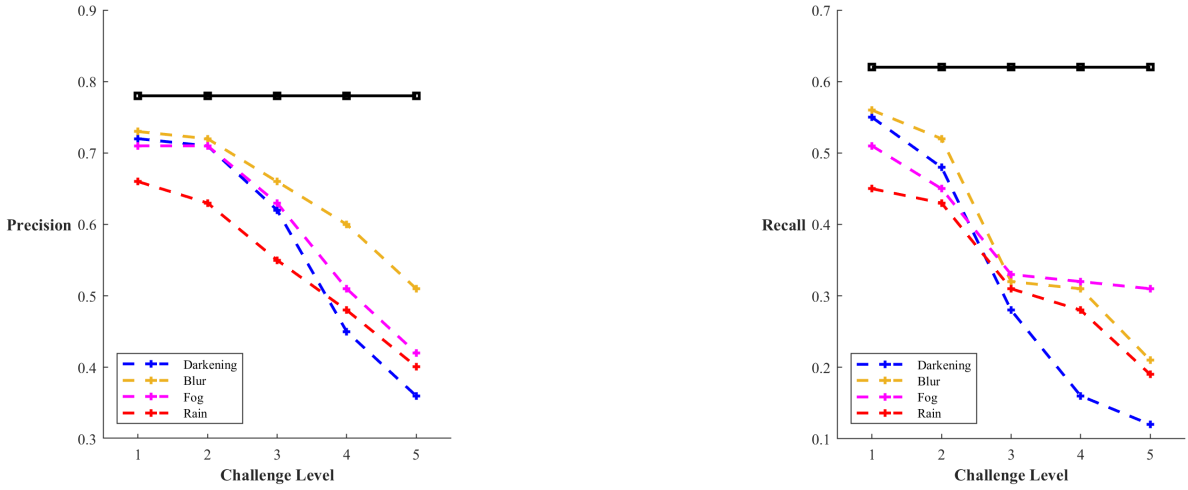


(a) White-box Natural Perturbation

(b) CURE-TSR Challenges

Figure 4.12: Comparing the drop of classification performance in various natural perturbations in terms of different challenge levels. The black line corresponds to the reference accuracy evaluated on the clean image data for the model (trained on clean image). There are different levels of performance drop in terms of various semantic perturbation directions and levels. (left: perturbed using white-box models, right: obtained from CURE-TSR dataset, considered as black-box)

this task is the full image shown in Fig. 4.5 for the CURE-TSD data and Fig. 4.6 for the virtual environment data. The evaluation metric is then the precision and recalls, where the former represents the ratio of the number of true positives to the total number of positive predictions, whereas the recall is the number of true positives to the total number of relevant objects. The reference model reports the detection performance with 0.78 and 0.63 precision and recall on the data without any challenge levels (the black reference line in Fig. 4.13). The resulting precision and recall are presented in Fig. 4.13 for each perturbation type and challenge level. Similar to the classification task, the model performance drop significantly on challenge level 3-5 for both precision and recall. The overall natural perturbation in blur results in a degradation of 0.08 to 0.23 in precision (10% - 29% to the reference model performance). The variation in fog, rain, and darkening further reduces the performance by up to 56% compared with reference precision. The



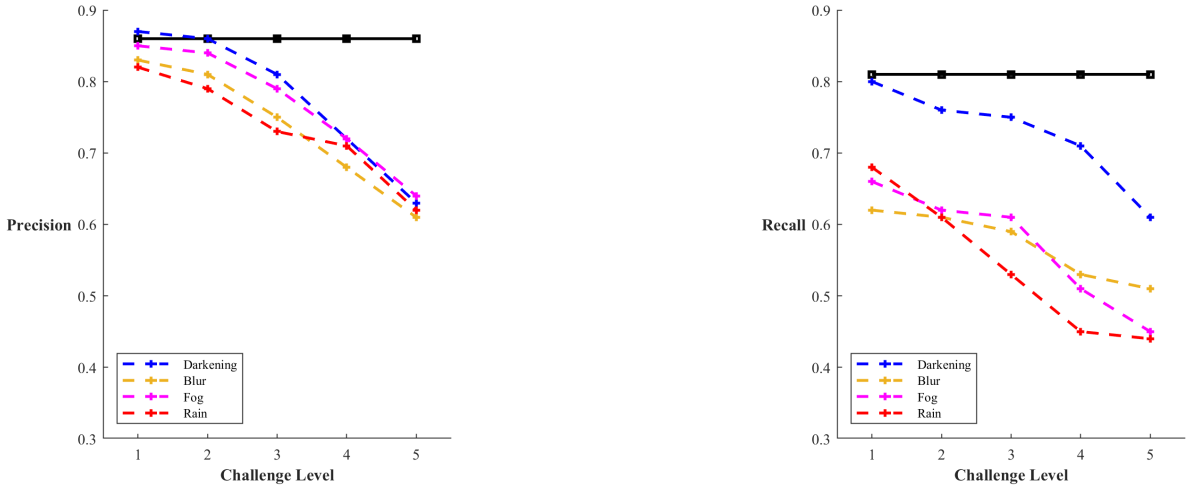
(a) CURE-TSD Precision

(b) CURE-TSD Recall

Figure 4.13: Comparing the drop of detection performance in various natural perturbations in terms of different challenge levels in CURE-TSD dataset. The black line corresponds to the reference accuracy evaluated on the clean image data for the model (trained on clean images). There are different levels of performance drop in terms of various semantic perturbation directions and levels. (left: model precision, right: model recall)

model performance degradation in the recall is slightly larger than precision which means that the natural perturbations result in more false negatives than false positives at the same challenge level. The performance drop in recall ranges between 13% to 64.5% compared to the reference one. Additionally, it can be noticed that the trend in performance drop in terms of the perturbation type agrees in Fig. 4.5 (a) and (b) since they share the exact challenge level definition. It reflects the importance of using the same standard to describe ODD.

The sensitivity to the perturbation is also examined for the virtual data and black-box model perturbation based on Unreal Engine as mentioned in Sec. 4.2.3. The example image has been illustrated in Fig. 4.6, and the challenge level is directly defined using the interface provided by the simulation engine. The challenge levels from 1 to 5 are sampled by setting the corresponding weather parameter in Carla from $(0, 20]$, $(20, 40]$, $(40, 60]$, $(60, 80]$ and $(80, 100]$ since the range of precipitation, fog and darkness are set from $[0, 100]$. The clean



(a) CARLA - Precision

(b) CARLA- Recall

Figure 4.14: Comparing the drop of detection performance in various natural perturbations in terms of different challenge levels in virtual data extracted from CARLA. The black line corresponds to the reference accuracy evaluated on the clean image data for the model (trained on clean images). There are different levels of performance drop in terms of various semantic perturbation directions and levels. (left: model precision, right: model recall)

image is directly sampled from sunny weather with all the perturbation-related parameters set to 0. It can be seen that almost all the perturbations degrade the detector performance with an increase in the challenge level, as illustrated in Fig. 4.14. One exception for the darkening effect in challenge level 1 is that the model has slightly better precision than the reference model since the natural variation generated based on the Unreal Engine black box in the low challenge level is not significant to change data distribution. Nevertheless, the precision dropped by about 30% in precision and 53% in recall compared to the reference model trained and evaluated in challenge-free virtual data.

The results in this part show that there will be a performance drop when evaluating the out-of-distribution (unseen) data for the model trained on clean images. In particular, the model sensitivity to different types of natural perturbations reflects the local robustness of the original model when facing the new data distribution out of its originally designed

domain of operation. The local neighbour accuracy and drop in model performance can be a good measure of the current perception model's robustness to natural perturbations. One can observe that the sensitivity trends agree with the local neighbour accuracy measure.

However, there is still a lack of universal standards for defining the qualitative description of natural perturbation. For example, the challenge levels used in the white-box, and black-box models are different compared with the drop in performance compared with the reference model. In order to calibrate such qualitative description more quantitatively, it is expected to use the degradation of performance with fixed reference and fixed data distribution which will be examined in the next section. Setting an industry standard for the quantitative description of the ODD challenge levels seems to be a promising future direction based on the proposed framework.

4.3.4 Evaluation of Robust Training

In this section, the effectiveness of robustness training is evaluated and compared with standard training methods such as Stochastic Gradient Descent (SGD) [254]; the recently proposed adversarial training method, projected gradient descent (PGD) [247]; data augmentation training method AugMix [248]; and the proposed RMA, RMWL strategy with and without strong/weak data filtering. The PGD method considers adversarial samples with a multi-step variant Fast Gradient Sign Method (FGSM), which is $\mathcal{L}_\infty$-bounded. On the other hand, the AugMix approach mix chains of augmentation operation and enforce the generated image with consistent embeddings to feed into the standard training process. The training procedure and settings details can be found in Appendix A.4.

Performance Evaluation on Known Distribution

The reference model is trained in the combined dataset of real (CURE-TSR) and virtual data obtained from the CARLA simulator. For reference purposes, the data being exposed to all the training models are the clean virtual data and the challenge-level 0-2 data in CURE-TSR. The performance of natural perturbed training is first evaluated on the seen distribution, e.g., the challenge-level 0-2 validation data. The models do not see these data in the training phase, but the test images are in the same distribution as the training data. The results are presented in Table 4.1 where the reference model trained using the SGD procedure reached a classification accuracy of 94.16%. It can be observed that both the proposed methods and the adversarial training with PGD and augmentation training

with AugMix provide about a 1-2% boost in classification performance compared with the reference one. This implies that the robust training will not reduce the model performance on the original dataset, which is a critical assumption in the ODD augmentation task. It is worth noting that the model that is trained with "only weak data" have an unfair comparison here since they only access a subset of the training data extracted using the procedure 3 and the anchor data with neighbour accuracy greater than 0.8 for corresponding perturbation are filtered out in the training process. Neglecting the strong data will result in minor degradation of the model performance on the original data distribution. However, in the continuous development process, these strong data points are already accessed in previous training steps and will not necessarily be needed for the transfer learning process for the model update.

Table 4.1: Comparing the performance and robust training on clean data.

		Change of test accuracy	
Methods		base dataset	only weak data
SGD [254]		94.16 (0)	NA
PGD [247]		+1.1	NA
AugMix [248]		+0.62	NA
RMA	Fog	+1.21	-1.3
	Rain	+1.31	+1.22
	Combined	+2.86	-1.31
RMWL	Fog	+0.87	-1.77
	Rain	+1.29	+1.26
	Combined	+2.12	-1.65

Effectiveness of Robust Training on Unseen Natural Perturbations

The robustness of natural perturbed training is first evaluated in the data set with challenge level 3-5. Notice that these data are not accessible during the training phase for all the training procedures. All the models are trained to map the challenge-level 0 to 2 data from

subsets of CURE-TSR and clean images from CARLA to predict the classification label. The results in presented in Fig. 4.15, where the reference robustness can be observed as the gap between the black dashed line and the solid line for the SGD reference.

By comparing each row in Fig. 4.15, the effectiveness of the robustify approaches in different perturbation directions is shown. The proposed RMA method even outperforms the reference model tested on clean data in the fog and rain natural variation. The robustness change can then be expressed as the gap between the robust training model and the dashed reference (reference model evaluated on challenge samples) over the gap between the solid and dashed lines (same model evaluated on different distributions). The percentage of increase of model robustness is then 61% for PGD, 67.7% for AugMix, 86% for RMWL, and 105.3% for RMA strategy in the fog perturbation direction. The trend looks similar for the rain perturbation direction as well. Notice that the weak data extraction process does not have an evident impact on the increased robustness in these two cases, even though the number of training samples is reduced by about $10 \sim 20\%$. In the darkness perturbation setting, the robustness improvement of the PGD method is around 68% and AugMix on about 58%. The RMWL method reaches a similar level of improvement between PGD and AugMix, whereas the RMA obtains the best among all strategies with an 84.5% increase in the model robustness.

The proposed strategies significantly improve the model robustness corresponding to the specific unseen distribution shift over the baselines. The results show that RMA and RMWL can cover well the unseen natural perturbation resulting in distributions with the generation model and predefined search space. Furthermore, it reflects that data augmentation often works better than updating the network weight search directions. Notice that the norm-ball-based adversarial training such as PGD works well on the simpler perturbation model, such as in the darkness case, because the natural variation generates a distribution shift in feature space covered in the bounded norm-ball.

Effectiveness of Exploration Space

The proposed robust training methods are further examined by varying the exploration space parameters in Fig. 4.16. The white-box model of fog (4.19), and rain (4.21) is used here for modeling the natural perturbation. In Fig. 4.16 (a), the exploration space is set with $\epsilon_1 = 0.4$ such that the exploration space for nuisance parameter is $\beta \in [0.05, 4]$. This range is selected based on the experiment in [236], where $\beta \in [0.05, 4]$ corresponds to a clear to thin fog based on human judgment with a visibility distance over 1 km.

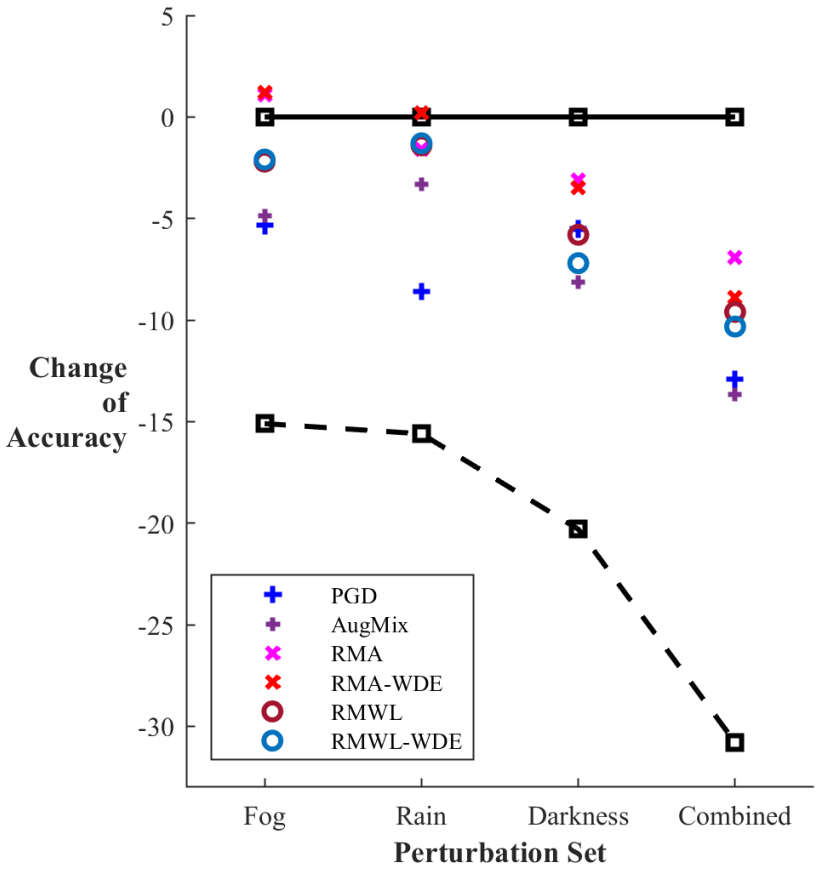
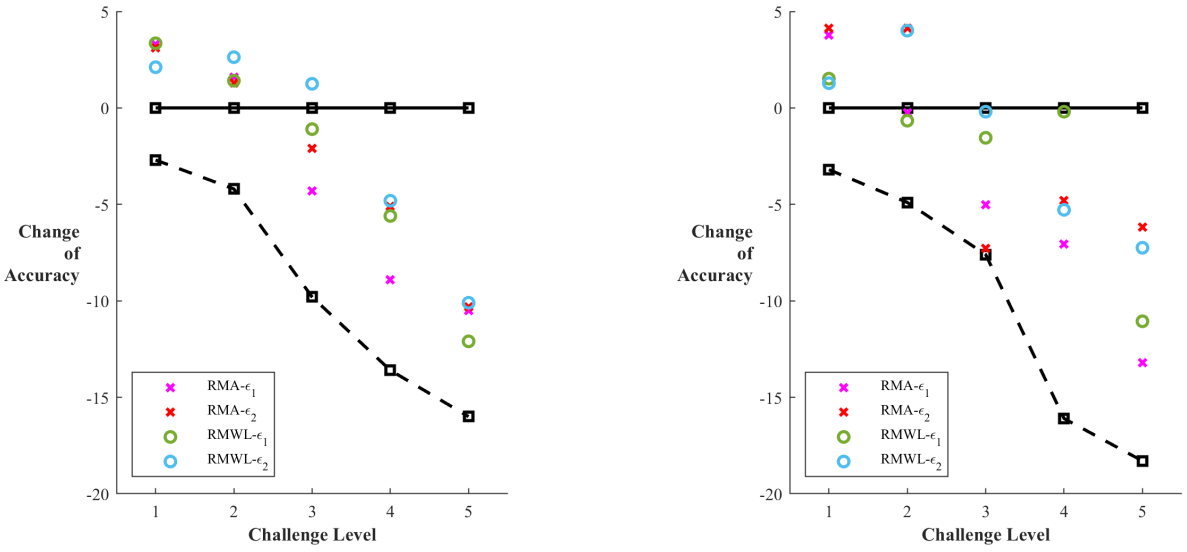


Figure 4.15: Model robustness evaluation on unseen perturbation for various training procedures. The black solid lines denote the reference model performance on clean data and the dashed denotes the reference model performance on the unseen perturbation data. The plus, cross, and circle signs correspond to the different trained models evaluated on the unseen perturbation set. The gap between the dots to the black solid line reflects the robustness (the lower the better).



(a) Fog variation

(b) Rain variation

Figure 4.16: The performance evaluation for the proposed methods with different settings of exploration space.

The search space is extended to $[0.05, 70]$ to reach a thick fog synthesis with less than a 200 m visibility range. It can be observed that with larger exploration space, the model robustness improved further for both RMA and RMWL processes with increased challenge levels. The larger exploration space configuration provides about 16% to 20% of the model robustness improvement in challenge level 3-5. Due to the data balancing issue, the larger exploration space results in lower performance in less challenging cases because the model is over-exposed to the perturbed data. Even though both RMA and RMWL can improve the model robustness when facing the unknown distribution shift, the RMWL algorithm reported slightly better performance in more challenging subsets. This is because the loss update direction always aims for the worst case in RMWL, whereas the RMA's loss update is related to the randomness included for k perturbed neighbours.

In Fig. 4.16 (b), the exploration space for rain synthesis $\delta_{\text{rain}} \in \Delta(\epsilon_1) = [0, 100]$ and $\epsilon_2 = 200$ with the unit (mm/hr) as the indication of rainfall rate in (4.22). The proposed methods RMA and RMWL can improve the model's robustness to the rain variation. No-

tice that the exploration space deviates from the robustness improvement in the challenge level since the larger exploration space means the model can access pseudo-perturbed data distribution generated by the white-box perturbation model during the training phase. The proposed robust training procedure utilizes the generation model to obtain the target distribution and update the model weights with the batch neighbour loss during the transfer learning phase.

4.4 Discussion

This chapter investigates the ODD augmentation problem for perception models that heavily rely on data-driven learning. The problem of extending the operational domain for the perception system is modeled as a robust learning problem when facing unseen natural variations in the continued engineering process. The natural variation when ADS outside of its designed ODD that defects the perception model performance has been investigated, and the typical models that synthesize such perturbation have been compared, and both white-box and black-box strategies are considered. Both natural perturbation generation strategies aim to reconstruct the corruption that the "unknown" environment could provide to the observation based on a semantic description such as rain, snow, or day and night.

This chapter proposes that the perception model's robustness should be characterized by the source data and the target data (the semantic perturbation direction) and the model that can shift such data to the "unseen" challenging ones. The characteristics of the data, such as neighbour accuracy and loss, are introduced to represent the model's local robustness to a given perturbation. Two robust learning processes were proposed by taking the general approximation of the natural perturbation model and the batched neighbour loss. Leveraging the neighbour accuracy property, a filtering strategy is proposed to extract more challenging data for future model validation and continued learning. The proposed methods are compared with recent norm-ball-based adversarial training and data augmentation methods. The advantage of the proposed robust training based on model-based perturbation is evident in all tests, especially with more challenging natural perturbation. It is suggested that a reference model be used to provide sensitivity (a reduction in perception performance) at anchor data and the corresponding noisy data. When comparing the robustness of two models, the performance degradation can also provide us with the pseudo-calibrated robustness measure.

The primary weakness of this study is that the transfer learning process cannot guarantee performance, especially after several cycles of ODD augmentation. Notice that this weakness is still an open question in the transfer learning domain of research. The other shortcoming is that, for now, the robustness training relies on the perturbation model's correctness. Since the perturbation model is fixed, the lack of randomness may be causing a drop in actual data with combined noises. For future work, borrowing the ideas from domain adaptation and randomization [255, 256] could be beneficial when extending the exploration space and further improving the model's robustness to combined perturbations.

The results of this investigation could also be helpful in guiding future research in challenging test case generation and validation for perception models. In particular, the weak data extraction procedure can be used to construct a challenging set to adjust the data balancing issue during the training procedure and for validation in the next cycle of perception model development. The quantitative description of the semantic perturbations could hopefully provide a standard in the ODD description to settle these high-level pieces of information that are important to the driving task.

Chapter 5

ODD Monitoring

The high-level autonomy requires the system to monitor the operational condition and respond to any violations by warning the user (Level 2-3), performing the DDT fallback (Level 4-5) [28]. Accepting ODD for safety assurance in the autonomous driving industry is based on an implicit assumption that ODD can be "perceived" in real-time vehicle deployment. Therefore, real-time monitoring of ODD is the foundation for achieving the Safety of the Intended Function (SOTIF). Real-time assessment for the perception modules (black box), usually faced with uncertainties during operation due to their data-driven nature, is the primary focus of this chapter.

5.1 Related Works

There are two typical strategies to monitor the ADS operational domain. The first of which is monitoring by software and hardware malfunction signals. Horwick et al. proposed a strategy and architecture for ODD monitoring in autonomous driving in [257]. Their safety monitor relies on the warnings and fault signals from functional modules that provide a sense of self-awareness. In [258], the authors discussed the detection framework and boundary degradation problem in ODD. They propose that the operational domain can be reduced during real-time operation depending on the hardware and software component malfunction.

Another type of ODD monitoring system utilizes redundancy information, such as geofencing with map information or rule checking based on a predefined truth table. In [257],

the authors discussed the necessary details for monitoring the system failures and troubleshooting based on redundancy for a driving assistance system. Reschka et al. proposed using skill graphs to monitor the system performance during operation and skill levels to make driving decisions in [259], which incorporates the idea of modeling abilities in a diagram with dependencies. Authors in [260] proposed an approach to identify the ODD with statistical data and risk tolerance attached to the geographical map. The authors attempt to make the ODD of the ADS part of the map attributes so that a simple geofencing strategy can be directly applied given the environmental conditions.

In this chapter, the ODD monitoring problem is investigated following the later strategy that utilizes the redundancy information, typically combining the likelihood of failure of the perception modules with map information and environment conditions.

5.2 Problem Formulation and Preliminaries

The operational design domain of PM mentioned in the previous chapter with the framework depicted in Fig. 5.1 is investigated. First, the ODD monitoring problem where the HD-map is unavailable is investigated. In this case, the only trusted source of clues is the real-time environment information (scenario and weather type) and the offline validation results on the PM (offline obtained ODD). The coarse scenario and weather type (highway or city road; sunny day or rainy) can be obtained by online querying with a rough localization in real-time [261]. The estimation checker design that only utilizes the PM's offline validated ODD for various sensors in online monitoring is presented in Sec. 5.3.

The HD-map can be considered a hidden sensor that can help the ADS accurately locate itself, and the information provided can be used as pseudo ground truth to check the results (lane lines, static objects) from the operating PM to estimate how well the perception works.

5.3 ODD Monitoring without Map Matching

The perception systems in an autonomous vehicle are heterogeneous and produce measurements with different accuracy in various driving situations. Considering the case when there is no HD-map for ground truth reference, it is proposed to use the offline performance measure to infer the online performance.

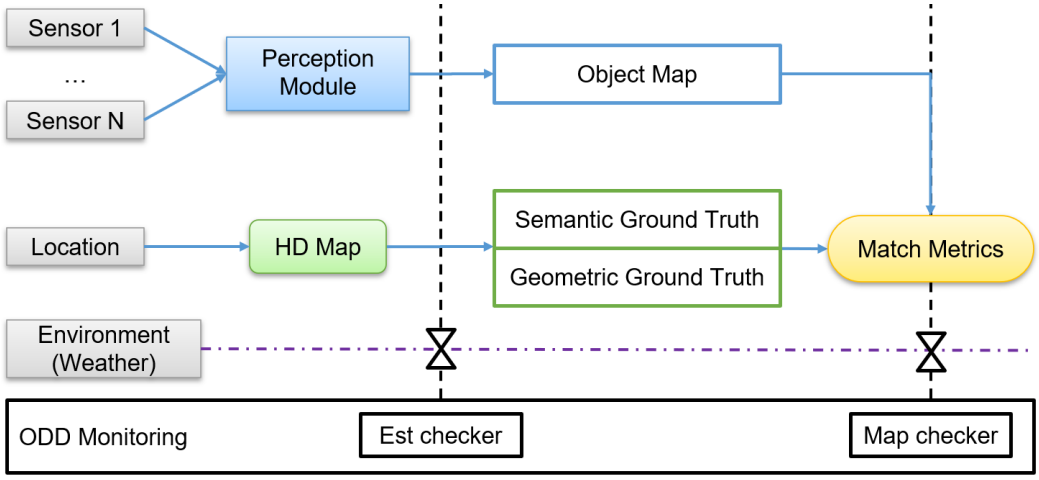


Figure 5.1: The proposed framework of ODD monitoring strategy with and without map information.

5.3.1 Probabilistic Model for Causal Relation

Based on the framework proposed in Chapter 3, the offline ODD model of the PM can be formulated as a conditional probability measured from the validation process. Bayesian Networks (BN) [262] can be used to propagate the probabilistic relationships between the failure in PMs and the ODD violation case. The BN allows for modeling in probabilistic relation among random variables in a graphical representation. For example, consider a known environment env, and the event as a random variable s_f that the PM failed to provide an accurate object map $\mathcal{OM}$. The relation between the two events can be described by

$$P[env, s_f] = P[s_f|env] \cdot P[env], \tag{5.1}$$

which can be depicted with the simple network depicted in Fig. 5.2. The oriented edge connects the pair of nodes in the BN model implying that the parent (env in Fig. 5.2) has a direct influence on the child (s_f in Fig. 5.2). The conditional probability $P[s_f|env]$ quantifies such effect from the parent node to the child. The random variable s_f that describes the PM failure can also interpret as a sensor or other function failures in the ADS.

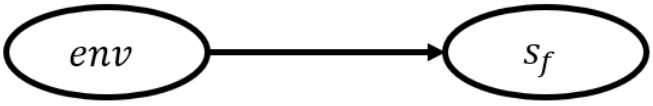


Figure 5.2: Two events BN example of PM failure and the driving situation.

5.3.2 Example

Consider an ADS with two perception modules, each of them having a different level of accuracy in various driving situations. The offline validated Likelihood of Failure (LoF) of the perception modules is obtained in Table 5.1. The LoF can be defined based on $LoF = 1 - mAP^{IOU=0.5}$ to represent the probability that the PM will fail under the given driving situation. The notion $mAP^{IOU=0.5}$ stands for the Mean Average Precision (mAP) in object detection task with the bounding box Intersection over Union (IoU) threshold set as 0.5. Other perception metrics can be used as they can be obtained from offline validation based on the framework proposed in Chapters 3 and 4.

Atom-Scenario	Weather	PM_1 LoF	PM_2 LoF
S1-Straight 2-Lane City Road	Normal	0.15	0.25
	Rainy	0.35	0.32
S2-Intersection	Normal	0.28	0.26
	Rainy	0.46	0.33
S3-Curve City Road	Normal	0.18	0.06
	Rainy	0.37	0.92

Table 5.1: Example of Validated Likelihood of Failure (LoF) for selected situations. PM_1 and PM_2 can be different perception modules, for example, an object detection module, and a lane estimation module.

CASE I: ODD Monitoring Encoded to Map

In the simplest case, the BN can be formulated as depicted in Fig. 5.3 with only one integrated PM. In this case, the PM can be implemented in various ways (camera-based, LiDAR-based, or sensor fusion) but only output one set of observation results. For

demonstration purposes, PM can be set to be the PM_1 in Table 5.1 and neglect PM_2.

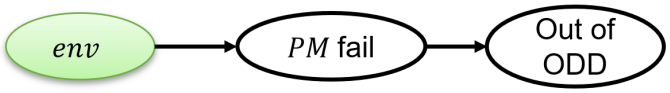


Figure 5.3: Formulation of the ODD monitoring based on the hard encoding for single perception module.

Notice that the *env* node (in green) is observable in this case, whereas the rest two nodes are hidden. When the *env* node is known as a constant, the only causal connection is between PM fails and the event of ODD violations. Thus, the ODD monitoring problem is to build a binary classifier based solely on the information from the prior conditional probability of PM failure. As a demonstration, a discrete threshold $\lambda = 0.2$ is set on the LOF, for example, $LOF > \lambda$, to indicate the out of the ODD event. The ODD monitoring classifier in this formulation can then be written as

$$\zeta(u; \lambda) \longrightarrow \{0, 1\}, \ u = env. \tag{5.2}$$

In this case, by examining the offline validated probability table in Table 5.1, the out-of-ODD warning should be activated at every intersection or when operating on rainy days.

The offline information can be encoded directly to the map as a geofencing function so that all the ODD condition is hard-coded with a geopositioning coupled with the weather condition. This is similar to the ODD formulation concept proposed in [260] that the conditions are directly encoded in the map. The ODD formulation can be extended to other regions given similar atom scenarios. A novice human driver might follow the same pattern to increase their attention on the road, similar to the CASE I formulation of ODD monitoring. For example, human drivers would pay extra attention to the harsh weather or the region that they have no experience driving before.

CASE II: ODD Monitoring with Redundancy Module

The autonomous vehicle can implement a redundant perception module for robust state estimation, and safety assurance purposes [263]. Consider the safety module of the ADS has a redundant PM_2 that outputs the objects in the same format of PM_1, but they have a different level of accuracy as listed in Table 5.1. The graphical model depicted in Fig. 5.4 can be obtained based on the probabilistic modeling using BN.

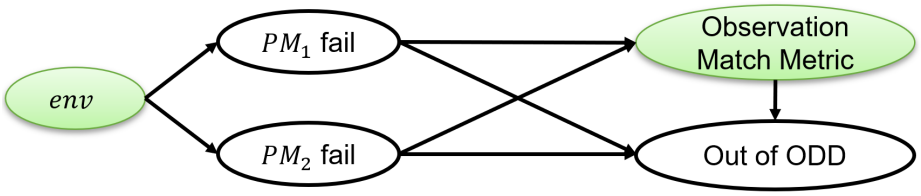


Figure 5.4: Formulation of the ODD monitoring based on the Bayesian Network with two perception modules in parallel.

It is assumed that the two perception modules generate $\mathbf{x}_1$ and $\mathbf{x}_2$ to represent the vector of states of the observed objects. Since the two perception module generates the object map in the same format, it can be observed a match metric between the two perception results. The perception module has two states regarding the observation $\mathbf{x}$, namely valid and invalid. However, they are hidden due to the missing ground truth, and the only evidence is the *env* and the match metric $k(\|\mathbf{x}_1, \mathbf{x}_2\|)$. The match metric measures the correlation between the two observations, and the function $k(\cdot, \cdot)$ should be smooth and decrease when the difference between $\mathbf{x}_1$ and $\mathbf{x}_2$ increases. One potential function choice of k can be the Gaussian kernel [264] (shown in Fig. 5.5):

$$k(\|\mathbf{x}_1, \mathbf{x}_2\|) = e^{-\frac{\beta_c}{2\pi}\|\mathbf{x}_1 - \mathbf{x}_2\|^2}. \tag{5.3}$$

In this case, the ODD monitoring classifier can then be written as

$$\zeta(u; \lambda, \lambda_m) \longrightarrow \{0, 1\}, \ u = \{env, k(\|\mathbf{x}_1, \mathbf{x}_2\|)\}, \tag{5.4}$$

where both thresholds for the LOF λ and the threshold on the estimated state distance of the redundancy modules λ_m are compared when classifying the results. The classification rule takes account of all the posterior distribution of the nodes in the network shown in Fig. 5.4. Typically, the ADS user should be warned in the following two cases: (1) the two perception modules provide contradicted estimations (2) both of the perception modules are not working well in the given driving situation by our previous knowledge. Thus, the classifier can be formulated as

$$\zeta(env, k(\|\mathbf{x}_1, \mathbf{x}_2\|); \lambda, \lambda_m) = \begin{cases} 1, & \text{if } (LoF^1(env) > \lambda) \ \& \ (LoF^2(env) > \lambda) \\ & \text{or } (k(\|\mathbf{x}_1, \mathbf{x}_2\|) > \lambda_m), \\ 0, & \text{otherwise.} \end{cases} \tag{5.5}$$

The ODD monitoring with redundancy module is developed based on the case I formulation with an extra observable metric node in favor of the redundant module. This

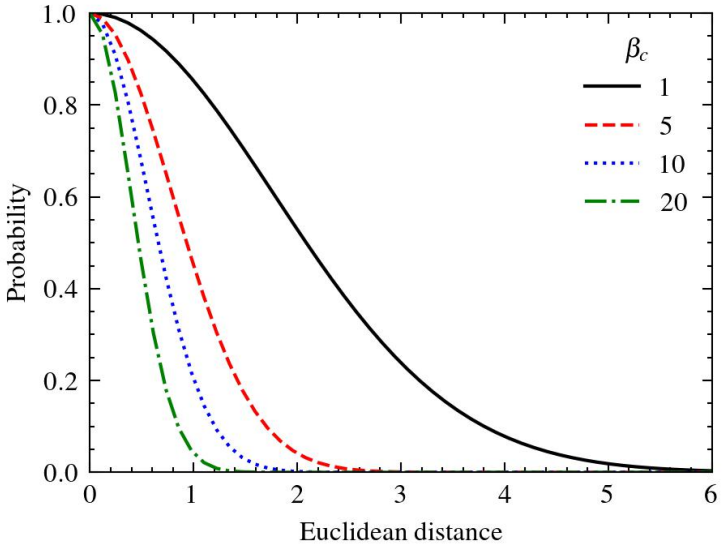


Figure 5.5: The Gaussian kernel as the covariance function with different values of β_c.

BN-based formulation can be flexible and extend to more general cases of N redundant modules. However, the autonomous vehicle application is usually more applicable with "1+1" redundancy implementation as suggested in [265]. In comparison to human driving, the classifier proposed in (5.5) can be viewed as two rookie drivers (perception modules) monitoring the driving situation and comparing their observations before deciding on whether the condition is safe enough for them or a professional driver (human driver) is needed to take-over.

5.3.3 Discussion

The ODD monitoring without map matching can be modeled with the flexible probabilistic approach that can incorporate the offline knowledge from validation to encode the ODD attributes in the map directly. Additionally, the observations from the redundancy module can be included to help the online monitoring model to classify the driving situation, not solely by offline experience.

The ODD monitoring strategy can be further extended in the case I formulation by exploring the atom scenarios. In this example, only the geometric and weather aspects of the driving situation are considered. If the probability table obtained from the offline validation step include the components such as connectivity (Layer 6 in Fig. 3.2), the ODD of applications such as CAV can be included in the map encoding as well. In the case II formulation, even though an additional redundant perception result is available, the ground truth referencing is still missing. The perception results matching between the two modules do not necessarily guarantee a correct perception (safe driving situation). For example, there could be cases where the two modules perform poorly and do not perceive the object in front of the vehicle. In that case, the correlation between the observations $k \to 0$, but it is unsafe to trust the current state estimations. Introducing the redundant perception module in the ODD monitoring classifier proposed in (5.5) utilizes the additional evidence and provides stricter rules compared to the case I formulation. However, the problem arises when one of the perception modules is way better than the other. The overall monitoring system will generate many more false alarms when one module makes the proper observation but the other keeps making mistakes. This brings requirements when designing redundancy systems. The two perception modules should avoid the case that one completely covers the capability over the other.

One of the significant drawbacks of the proposed approach is the number of atom scenarios that need to be tested during the offline validation step. Every edge of the BN represents a dependency between the connected random variables and should be defined with a probability distribution. However, the probability distribution between the nodes can be defined with more sophisticated ones since the perception module failure and observation matching may depend on the parameters such as distance, field of view, etc. Furthermore, the ODD monitor may also consider attributes such as the object distance and road curvature to refine the classification step. These attempts will increase the model's complexity and will be investigated in future work.

5.4 ODD Monitoring with HD Map

HD map is one of the core technologies that support autonomous driving applications, especially in constraint scenarios where the map can be pre-sampled. The HD map provides critical semantic (drivable areas, signage) and geometric (lane attributes) information about the driving environment. This information provided by the HD map can be used as a pseudo-ground-truth reference to detect violations of ODD conditions.

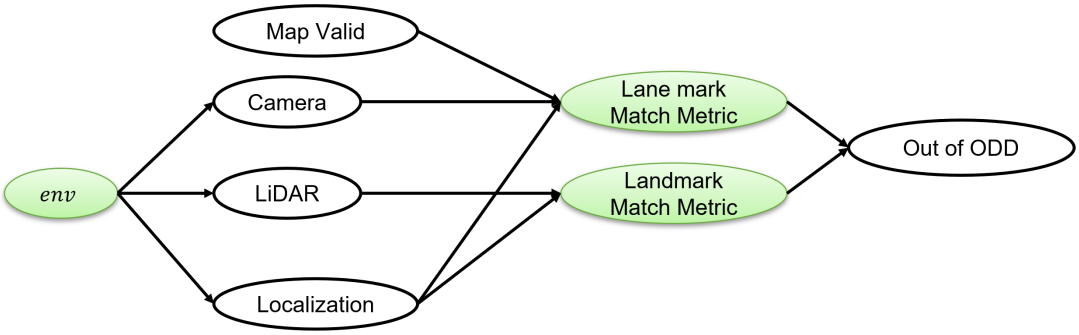


Figure 5.6: Formulation of the ODD monitoring based on the map checking and stacked perception modules.

5.4.1 General Architecture

Consider the autonomous vehicle equipped with a front camera to observe the lane markings and LiDAR to observe the landmarks. This section considers that separate perception modules do not return the object map in the same format, which is different from the case II formulation. As depicted in the causal map in Fig. 5.6, the environment will influence camera and LiDAR-based modules differently. For example, the road signs and signal boards are retro-reflective and can be robustly detected using LiDAR intensity measures. However, it is not the case for camera-based perception. It is also assumed that all sensors are well-calibrated and share the same pose with the vehicle, and the distortion of the sensor has been removed [266].

The coordinate system is shown in Fig. 5.7 where $\{\mathcal{W}\}$ defines the world frame that is used for HD map querying. The vehicle reference frame with X^{vc} pointing to the vehicle heading is defined as $\{\mathcal{VC}\}$. The 2-D coordinate is accepted here for simplicity, which can be extended further to a 3-D world map. The HD map database contains the actual world

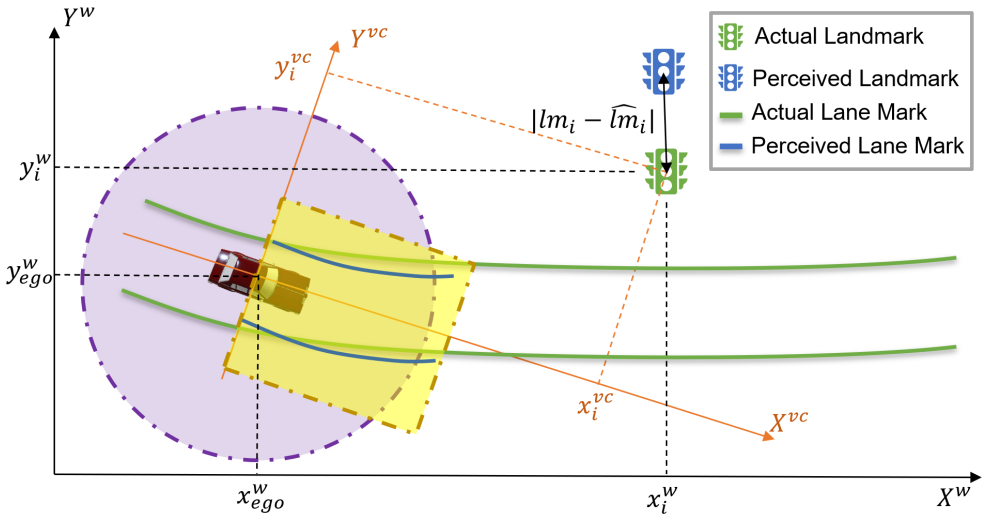


Figure 5.7: A demonstration of the landmark and lane marking observation and matching in the world frame and vehicle reference frame (based on ISO 8855 [10]). The measurements in the vehicle frame can be transformed to the world frame and vice versa based on the pose and calibration.

position of the landmarks and lane markings. The location of the i-th landmark in the HD map is denoted as a single point $p^{\mathcal{W}} \in \mathbb{R}^2$. Note that in reality, a landmark usually consists of a set of points; however, landmark location $p^{\mathcal{W}}$ can be selected at the centroid of the point set.

The vehicle pose at the world frame can be directly obtained through an onboard localization algorithm. Since it is assumed that the same pose for the vehicle and sensor, then rigid body transformation from the vehicle reference frame to the global frame at time t can be described as

$$T^{\mathcal{VC}\to\mathcal{W}}(p_t^{\mathcal{W}}) = R(p_t^{\mathcal{W}}) + p_{ego}^{\mathcal{W}}, \quad R = \begin{bmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{bmatrix} \tag{5.6}$$

where θ_t corresponds to the vehicle heading (rotations about the origin) at time t.

The basic map features are extracted within a pre-defined range around the vehicle. For lane markings, a forward-looking rectangular area originating from the vehicle center (the yellow area in Fig. 5.7) is used to search the markings of the vehicle's current lane. Similarly, the related landmarks of the ego-vehicle are filtered out using a circle centered at the vehicle's origin (the purple area shown in Fig. 5.7). These pseudo-ground-truth labels

extracted from the HD map are then used for the online estimation of perception system performance. The goal is, given the HD map database $\mathcal{D}_{map}$, the ego vehicle pose at world frame p_{ego}^{W}, perceived lane markings and landmarks $\{lm_1, lm_2, ..., lm_n\} \in \mathcal{OM}$, decide if the current perception system in this architecture still in its ODD.

5.4.2 Lane Marking Measurements

The lane markings can be extracted from the front-view RGB camera image using various techniques, such as Inverse Perspective Mapping (IPM) based lane detection [267], weighted regression [268, 269], and semantic segmentation [270]. Overall, the perception system is expected to output the estimation of lanes to implement in-lane localization, lane keeping, etc. The lane estimation in the vehicle reference frame is expected at least to help these ADS functions. The details of the implementation can be referred to in Appendix A.5.

In our work, the perceived lane markings at time t are projected to the vehicle reference frame such that the left and right adjacent lane lines are obtained in parabola model [271]

$$y_{l,t} = a_{l,t} + b_{l,t}x_t + c_{l,t}x_t^2 \tag{5.7}$$

$$y_{r,t} = a_{r,t} + b_{r,t}x_t + c_{r,t}x_t^2 \tag{5.8}$$

where the subscript l and r denote left and right adjacent lane marking, respectively. The values $[a,\ b,\ c]$ are the coefficient of the parabola model. The barrier and guardrail can be estimated using the same model to represent the lane boundaries. The lane markings here are only used to define the road boundaries. Similarly, the actual lane markings can be extracted from $\mathcal{D}_{map}$ based on the vehicle pose at the discrete time t and a given region of interest. The actual left and right lane lines $\{\hat{y}_{l,t}, \hat{y}_{r,t}\}$ can be obtained in the similar format as (5.7).

Two sets of parabola equations can be obtained from the $\mathcal{VC}$ frame from the previous step. Both sets can vary in size and range of the line. Consider in the region that the actual left and right lane lines exist; however, if fewer than two parabola lines are detected in the perception module, the event that "lane missing" will be triggered, and the lane mark match metric will be "not match". In reality, perception modules will produce estimation on more than two lanes [268], only the nearest lane markings that define the current driving lane of the vehicle are considered. If the reference lane markings are unavailable (typically in the intersection as shown in Fig. 5.8), the lane marking estimation is ignored from the perception module. The other case is that the lane markings get shorter when a vehicle

approaches the intersection (Fig. 5.8). In this case, an additional constraint on x_t can be applied,

$$\hat{y}_{l,t} = a_{l,t} + b_{l,t}x_t + c_{l,t}x_t^2, \ x_t \in [0, d] \tag{5.9}$$
$$\hat{y}_{r,t} = a_{r,t} + b_{r,t}x_t + c_{r,t}x_t^2, \ x_t \in [0, d]. \tag{5.10}$$

Figure 5.8: Sample front images from real (top row) and virtual (bottom row) driving environments. The left corresponds to the cases where the reference lane markings are unavailable, whereas the right two images correspond to the case with shorter lane information approaching the intersection.

To verify the land markings between the HD map database and those perceived in the current time frame, The following features are recorded for further analysis.

1. Averaged deviation error: A consistent number of M points were sampled from each pair of parabolic, and each point can have the point-wise error $\|\hat{y} - y\|$. The averaged deviation error for each frame of observation can then be calculated as

$$e_{ade} = \frac{1}{M} \sum \max(\|\hat{y}_l(x) - y_l(x)\|, \|\hat{y}_r(x) - y_r(x)\|), \ x \in [0, d]. \tag{5.11}$$

The maximum point-wise error in the adjacent lanes is selected in the $[0, d]$ forward-looking range. Overall, the expression in (5.11) evaluates the average lateral displacement error.

2. Weighted deviation error: It is worth noting that the perception system may have a larger error when estimating the lanes far away from the current position. That is, a large error of the lane displacement at 30m away does not necessarily have the same impact on safety when the same error at the current location. By applying the weight α^x along the sampling points, the weighted deviation error can be calculated as

$$e_{wde} = \frac{1}{M}\sum_{i=1}^{M} \alpha^i \max(\|\hat{y}_l(x) - y_l(x)\|, \|\hat{y}_r(x) - y_r(x)\|),\ x \in [0, d],\ \alpha \in (0, 1]. \quad (5.12)$$

It should be noticed that these metrics are sensitive to localization (and calibration) since it provides the translation between the coordinates and querying evidence for extracting the pseudo-ground truth. These effects will be further analyzed in Sec. 5.5.

5.4.3 Landmark Measurement

Various landmark features can be used to verify the driving condition, such as the light poles [272], road signs [273], and even buildings [76]. The sensor implementation for the landmark perception could vary from the camera, LiDAR, or fusion to provide both the semantic and geometric information [76]. In this study, the problem is simplified to measure the landmark's location only, and the signage's relative position can be directly measured based on the LiDAR intensity (filtering the point cloud with a threshold). The centroid of i-th cluster $lm^{vc}_{i,t} = [x_{i,t}, y_{i,t}]$ is then considered the landmark measurement in the vehicle reference frame at time t. In reality, additional filtering and association steps are required for advanced landmark measurement [274, 275]. It is assumed that these refined steps are already implemented in the perception module, and LiDAR-based $\mathcal{PM}$ directly output the detected landmarks in this thesis.

Given the ego-vehicle localization at discrete time t, the surrounded landmark of interest can be obtained as a list of positions $\hat{lm}^w_1, ..., \hat{lm}^w_k$ confined in the circle around the vehicle (the purple area in Fig. 5.7). It is assumed that the k number of landmarks is sparse in the vehicle frame to avoid the overlap between the reference landmarks. The positions of the perceived landmarks are transformed into the world frame based on (5.6) to compare with

the actual landmark locations. In reality, the perception module may miss the detection, so it should be considered in the error calculation when the length of actual landmarks of interest and the perceived list. To simplify this problem, here a constant length of the list is used, e.g., $k = 4$, and the landmarks are the ones in the nearest feature in each quadrant of the vehicle frame. Thus the error metric used here for landmark matching is

$$e_{lm} = \frac{1}{4} \sum \|\hat{lm}_{i,t} - lm_{i,t}\| \tag{5.13}$$

There could be a case when the observation or the ground truth only has less than four reference landmarks. In that case, the corresponding field of that quadrant will be filled with the ego-vehicle's location so that it will not affect the error estimation in (5.13).

5.4.4 Out of ODD Classification

The online monitoring for ODD boils down to the classification of whether the system can handle the current driving situation. Otherwise, it should prompt a warning or alarm to the user to prepare the backup plans. The probabilistic graph is arranged into a BN shown in Fig. 5.9.

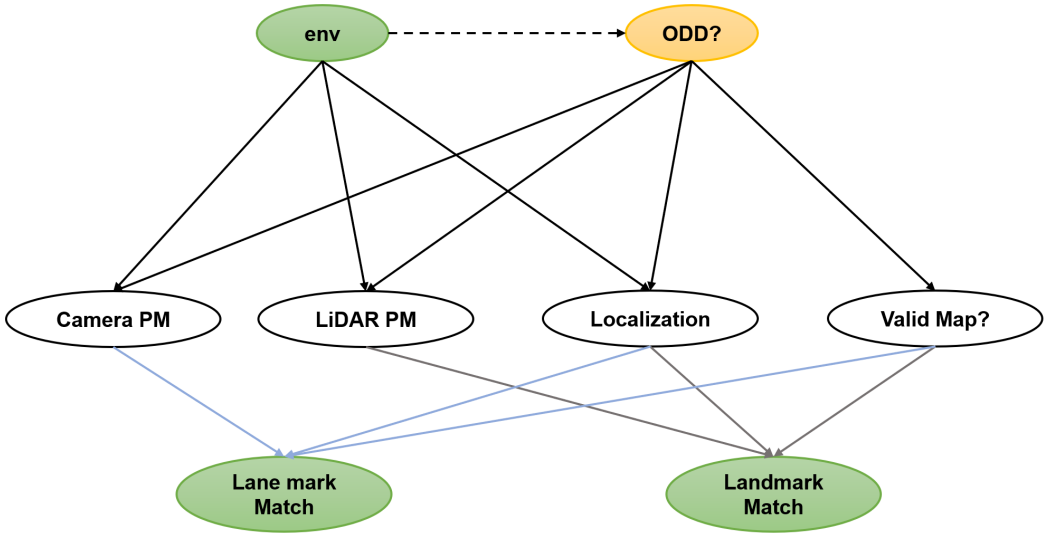


Figure 5.9: The BN diagram of the ODD monitoring based on map checking. The green nodes are observable nodes that can be obtained in real-time as evidence.

The environment is a set of discrete conditions that do not necessarily fully represent the system's ODD. This is different from the case of Sec. 5.3 that an ODD defined based on the environment condition can eventually be checked with the table of the driving situations obtained through offline validation. The environmental condition that can be observed are discrete and cannot formally represent the actual operating condition. It is assumed that the observed environment condition is correlated with the event of whether the vehicle is in the ODD or not (yellow node in Fig. 5.9). The observed environment and the actual "out of ODD" event will directly relate to the perception modules. For example, an observed snowstorm will impact both the camera and LiDAR module perception, and it can be implied as out of ODD, given the conditions in the validation step. On the other hand, if the observed environment does not include in any of the previous validation, or the case does not correctly reflect the inability of the current ADS function, then the perception system could be degrading in performance and might trigger the out-of-ODD event even if the environment is observed to be a normal one.

Directly observing if the vehicle's perception module is out of its ODD is hard. However, the evidence mismatching may cause by the invalid map, wrong localization, and degraded performance of the camera and LiDAR. More importantly, these factors can be related to the observable measure from matching lane marking and landmarks between the knowledge base and the real-time measurement.

Out-of-ODD classifier is then formulated by knowing the evidence on the three nodes in Fig. 5.9 and aims to find the following distribution

$$P(ODD|env, match\_landmark, match\_lanemarking), \tag{5.14}$$

where the random variable of whether find a landmark or lane mark match can be derived based on the measured metrics in (5.11), (5.12). The knowledge of the network includes both online observation and offline experience. The online observation includes the environment condition (queried online) and the match metrics of landmark and lane marking observations. The environment condition has three discrete values here, namely $\{normal, unusual, harsh\}$. The atom scenarios-alike (as Table 5.1) driving situations can eventually be categorized into these three types. Even if the observed environmental conditions are reported as "normal", it does not necessarily mean that the system usually operates in the environment and within its ODD. Similarly, under the reported "harsh" weather conditions, the ADS may still be in its ODD due to the simplicity of the driving task undertaking. The unusual cases will cover those driving situations that have not been seen yet or have not been validated scenarios.

In the BN diagram depicted in Fig. 5.9, the match metrics are events that record a match between observed features and the actual features recorded in the map. Here a simple cut-off function that maps the error metrics in Boolean is used, in which 0 means to correct match and 1 corresponds to an erroneous one. This can be done by simply compare e_{ade}, e_{wde} and e_{lm} with the constant boundary ϵ. Another choice is to set the bottom layer node of Fig. 5.9 with a continuous distribution with the measured error as the dependent variable. For example, let the error $e_{wde} \in [0, \infty)$ be the dependent variable, and the probability distribution for the event of finding a correct match can be formulated as an exponential distribution, where the more prominent the error is, the lower the possibility that a match is found.

5.5 Experiment on Virtual Environment

This section presents the software tools and the parameter settings for the experiment conducted to highlight how the online ODD monitoring strategy performs based on various strategies. Furthermore, the quantitative performance and the critical issues raised from the source of error are investigated. To this end, the open-source tool CARLA [189] is chosen as the primary simulation tool in our experiment. Carla provides a versatile simulation API that one can control the overall sensor noise level and environment setting, which will be helpful for the ground truth generation in our simulation. The PGM and corresponding solvers are implemented using the pyAGrum package [276] for its high-level interface and fast-speed inference.

5.5.1 Simulation Setup

Driving Route Configuration and Ground Truth ODD

The CARLA-Town05 map is chosen as an integrated scenario to test the ODD monitoring module. The environment is divided into four different sections as shown in Fig. 5.10.

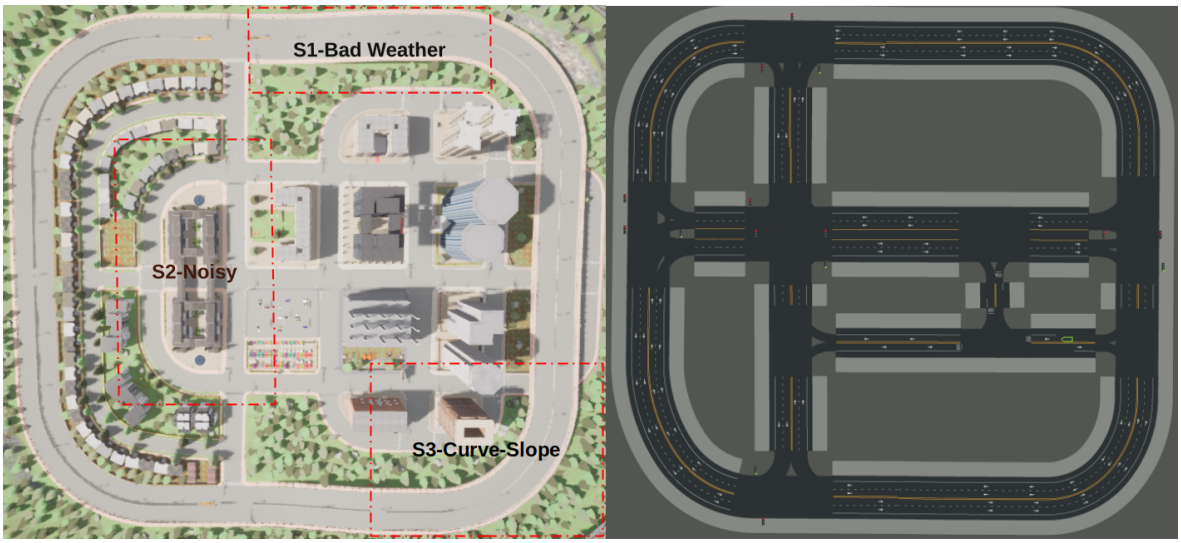


Figure 5.10: The scenario region setting in Town05 map using CARLA simulator, the selected regions are set to be out of ego vehicle's ODD.

The red rectangular covered area is considered the ground truth out of ODD regions with different settings. The "S1-Bad Weather" scenario covers one segment of straight highway on the Town05 map, and it is set to have heavy rain and large fog density in this region. Once the vehicle enters the region, the weather's configuration will change to harsh weather, and the RGB input for the perception system will be distorted using the Unreal Engine. In the "S2-Noisy" region, the sensor noise is increased to a level that violates the safety requirement for autonomous driving (details in Table 5.2). The "S3-Curve-Slope" region represents the potential map error since the original Town05 map in that region has a curvy slope road which violates our assumption in the 2-D geometry (Fig. 5.7) to compute the feature matching. In other regions of the map, the weather is set to sunny at noon with no additional sensor noise.

	Localization (m)	Weather Setting	Post-Processed Measurement Noise
S1	lat: 0.2 ± 0.1(m) lon: 0.2 ± 0.1 (m) heading: 3 ± 1 (°)	foggy & rainy	none
S2	lat: 0.5 ± 0.2 (m) lon: 0.5 ± 0.2 (m) heading: 8 ± 4 (°)	normal	landmark displacement, 0.8 ± 0.4 (m)
S3	0.2 ± 0.1(m) lon: 0.2 ± 0.1 (m) heading: 3 ± 1 (°)	normal	landmark displacement, 0.8 ± 0.4 (m)
Other	lat: 0.2 ± 0.1(m) lon: 0.2 ± 0.1 (m) heading: 3 ± 1 (°)	normal	none

Table 5.2: The sensor noise, weather configuration in simulation and post-noise adjustment setting in our experiment.

The CARLA server is hosted on the local machine, and the SUT vehicle is initiated in the configured driving environment. The autopilot (automatic follow-the-way points) is used to ensure the vehicle walks through the pre-defined regions with the front camera-based perception modules to estimate the lane markings. The vehicle sensor noise level and the weather configuration are changed during the trip based on Table 5.2. The noise in the LiDAR-based perception is simulated by adding the post-processed measurement noise to the ground-truth landmark positions (traffic lights and fixed points). These displacements are then added to the recorded data after each successful travel on the map.

Classifier Configuration

In this experiment, the following three types of classifier implementation for ODD monitoring tasks are considered.

1. **Geo-fencing Classifier:** The ODD table checker (CASE I in Sec. 5.3) follows the geo-fencing technique that monitors if the vehicle enters a pre-defined zone for an ODD warning. In this experiment, S3-Curve-Slope is encoded as prior knowledge of the ODD warning zone. This mimics the real case where part of the map is not available or well-annotated, then the ODD warnings are encoded directly using geofencing. Even though the other two regions are considered ground truth, they cannot be directly observed in CASE I setting.

2. **Rule-based checker:** This is based on the redundancy checker (CASE II in Sec. 5.3) which utilize the additional "confidence score" as evidence from the redundant modules, such as estimated covariance or the output from the activation layer of a neural network.

3. The BN classifier with prior knowledge is stored in the conditional probability table shown in Fig. 5.11. This prior knowledge takes potential localization and map errors into account. The exact value of the conditional probability table can be obtained by the exhausted validation in the offline step. The conditional probability table is created in a similar way to [277] to construct the BN classifier.

Notice that the conditional probability table initialized for the BN classifier on each node and links can be customized for various assumptions. For example, the parameters in Fig. 5.11 can be set such that other variables are independent of the map to achieve the perfect map assumption.

5.5.2 Evaluation Metrics

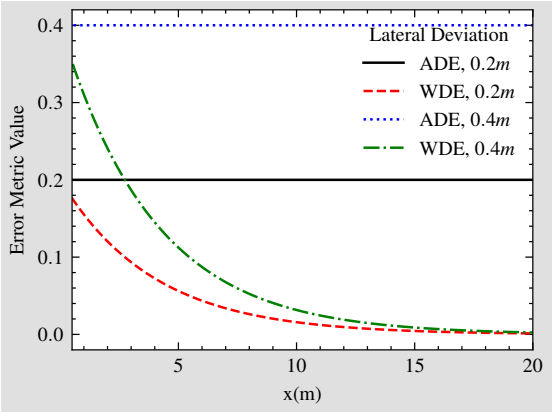
To assess the performance of each method and corresponding parameter setup, the detection and classifier metrics used in this experiment are presented. The detection metrics evaluate the correlation between observations and the existing map knowledge, as addressed in Sec. 5.4. The classifier metrics primarily focus on the performance of the ODD monitoring strategies laid out in the previous part.

Figure 5.11: The example conditional probability table for BN classifier.

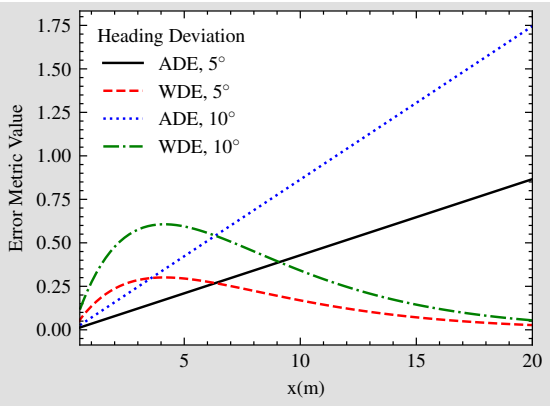
Detection Metrics

The detection metrics are used to classify whether a landmark or lane markings matching event is available in the frame for the bottom two nodes of the BN classifier. It is proposed to use error measures such as e_{ade}, e_{wde}, e_{κ} and e_{lm} in Sec. 5.4.

In the case of the intersection or other regions with no proper ground truth lane markings, the landmark features are compared in the map to estimate how "good" the current perception system works. Similar to the condition of lane marking match events, the bounded requirement $e_{lm} \leq \epsilon_{lm}$ is set for the landmark matches. The lane markings matches are found in the most straightforward way by limiting $e_{ade} \leq \epsilon_{ade}$ or $e_{wde} \leq \epsilon_{wde}$. The boundary ϵ is set based on the localization requirement for autonomous vehicles proposed in [278]. The sample of the lane estimation error for perturbed localization measure at straight lane can be found in Fig. 5.12. The error metrics depend on the look-forward distance and the real deviations of vehicle position. The heading deviation will generate a considerable bias on the e_{ade} with a larger look forward distance since the error will accumulate much more at those points far away from the ego-vehicle. Typically the e_{ade} provides a more consistent error measure at curve roads or requires a more comprehensive look forward distance. In the following analysis, both error metrics e_{ade} and e_{wde} are considered for the detection matching.



(a) Lateral Deviation



(b) Heading Deviation

Figure 5.12: The sample error measures with respect to lateral deviation, heading error as well as the choice of looking forward distance at straight lane.

It should be noted that these lane marking error measures can only be calculated with a lane estimation first. However, in many cases, especially when the perception system is out of its ODD, there is no proper lane estimation (often returns none in lane parameter estimation). The NAN values are filled to the previously available detected parameters in this case.

Classifier Metrics

Since the ODD monitoring problem is formulated as a binary classification task in this experiment, the contingency table can be used to evaluate the performance. The probability of declaring the current situation is "out of ODD" when at the ground truth high noise and harsh weather area is called sensitivity, or True Positive Rate (TPR). On the other hand, the rate of warning the driver of "out of ODD" when it is actually safe to drive is called False Positive Rate (FPR). The TPR and FPR can be collected by computing the following from the contingency table

$$\text{TPR} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}, \tag{5.15}$$

$$\text{FPR} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}. \tag{5.16}$$

The TPR and FPR can also be derived from the PGM-based classifier by computing

$$\text{TPR} = P[\zeta(u;\lambda) = 1|GT = 1], \quad \text{FPR} = P[\zeta(u;\lambda) = 1|GT = 0]. \tag{5.17}$$

The value of each classifier $\zeta(u;\lambda)$ is set to 1 when the posterior distribution of out-of-ODD detection has a value higher than the threshold λ.

By setting different values of λ, different classifiers with pairs of TPR and FPR can be obtained that describe the classifier performance. For example, setting the $\lambda = 1$ always corresponds to categorizing the vehicle to be in a safe driving condition in the predefined ODD. On the other hand, if $\lambda = 0$, then consider all the samples as "out of ODD". The various classification thresholds λ yield a receiver operating characteristic (ROC) curve that provides an aggregate measure of the classifier performance.

5.5.3 Qualitative Analysis

The qualitative analysis of the error measure in various driving scenarios defined in Table 5.2 based on their ground truth ODD settings are presented here.

Normal Driving Evaluation

Multiple driving routes are collected in the given map and select one typical normal driving performance (in ODD) shown in Fig. 5.13. It corresponds to the typical driving trajectory on the straight lane with minor weather changes in the back end of the simulator. It can be seen that even in the straight lane, the lane parameter estimation is sensitive to the weather change that happened at the time 10s after the beginning of the roll-out. The heading angle and curvature are volatile to the perturbations and usually have abrupt changes between consecutive frames. The differences between the e_{kappa} are still limited in the pre-defined boundary. Even though there are multiple degrees of discrete measures in the lane parameter errors, the combined error (the bottom of Fig. 5.13) provides us much a straightforward measurement of lane estimation. Both the projected errors reflect the average bias in lane estimation in meters that in this case lower than the required localization error proposed in [278].

In Fig. 5.14, the recordings that reflect the case of ego-vehicle driving at the curvature road with safe operating conditions are demonstrated. It can be found that the trend in heading angle and the curvature detection change reflect the actual movement of the vehicle entering the curve road. The entering of the curve road starts at 5.5 seconds based on the observation of the ground truth lane parameters. Even though evaluating in normal weather where the lane detector is trained, the neural network predictions have a more significant error in the curved lanes than on the straight road. The detected lane parameter relies on the lane points extraction from the neural network and then curve fitting of the selected lane points, where the overall process introduces more noise in curve lane fitting. Additionally, the bottom of Fig. 5.14 showed that the curvature could be examined on both the proposed ADE and WDE metrics. It can be seen that the ADE is more sensitive to the geometrical change of the lanes due to the lane detector generating more error at a longer look ahead distance. Notice that the vehicle is driving in average weather, and the only change is the road geometry. The lane estimation offset is still within the acceptable range [278]. The WDE error is preferred, which provides a more consistent estimation of the lane marking matching.

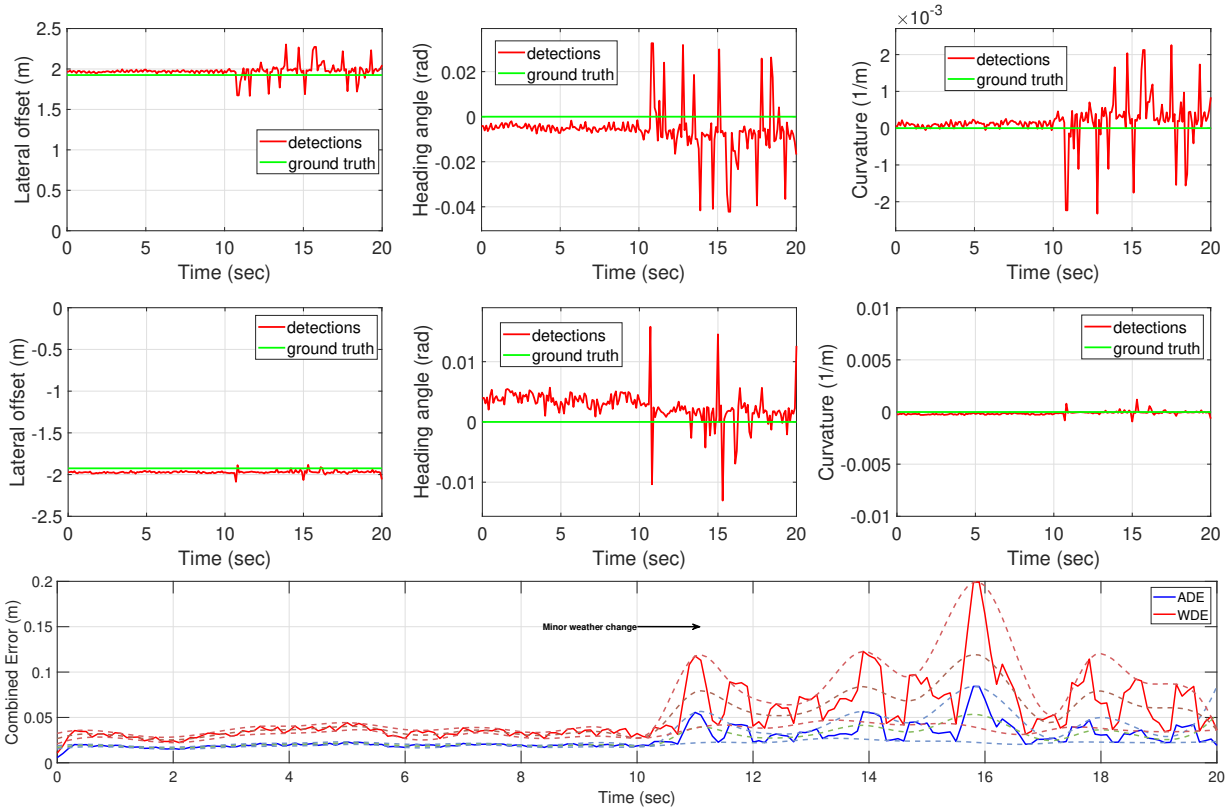


Figure 5.13: The lane estimation parameters and ground truth values at the straight lane with minor weather changes (top: left lane parameters; middle: right lane parameters; bottom: projected error).

Influence of the Weather-induced Perception Error

One trajectory recording in S1 is presented in Fig. 5.15. The vehicle turns into S1 with an alarming weather setting illustrated in Fig. 5.10. In our implementation, the misdetection of the lanes is handled by using a queue structure that reads the last frame detection and utilizes the stored information for the misdetection frame. With this implementation, the segments of abrupt changes reported in Fig. 5.15 of both left and right lane parameter detections correspond to the misdetection of the perception model. For example, the estimated left lane offset, heading angle, and curvature have a constant value of around $9 - 15$ s since there is no proper lane detection reported from the perception module due to low confidence in lane point selection and curve fitting. This low confidence level

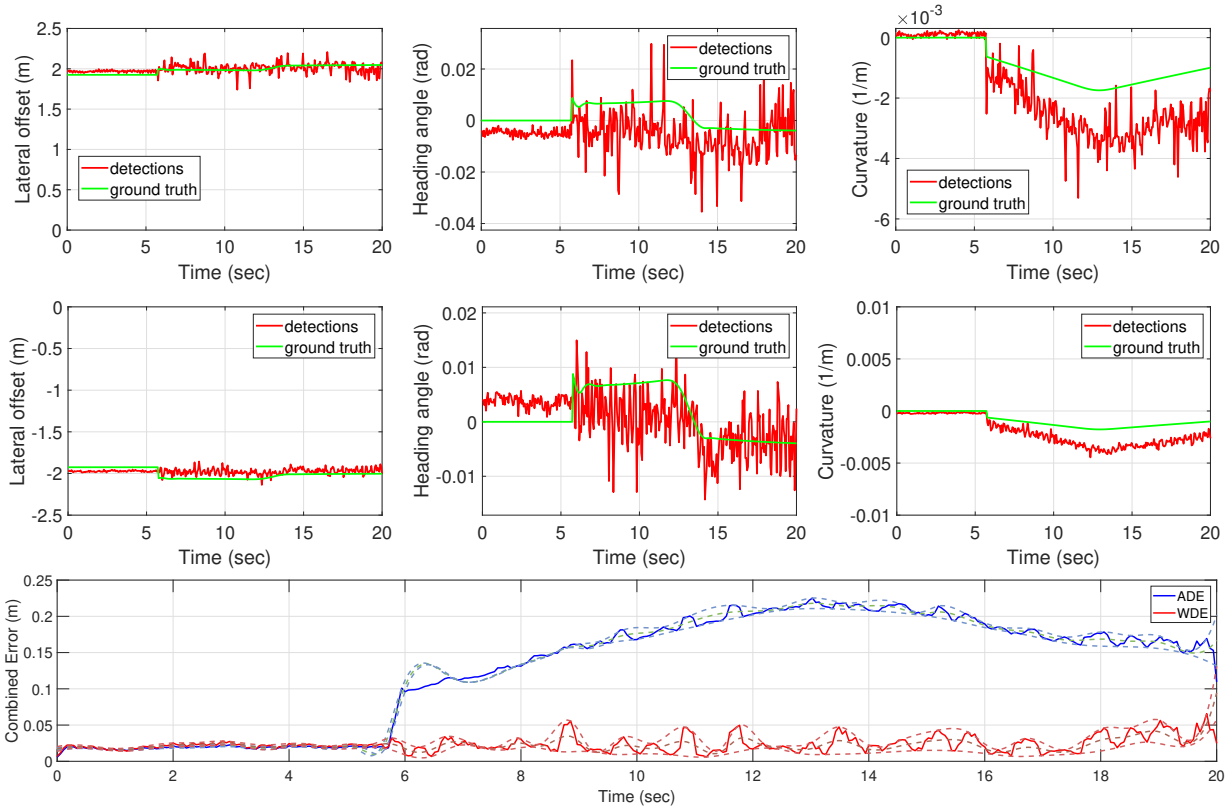


Figure 5.14: The lane estimation parameters and ground truth values at the road curve within ODD (top: left lane parameters; middle: right lane parameters; bottom: projected error).

and misdetection can be formulated as a binary signal sent from the perception module. However, to make the projected lane error consistent, the lane parameter in memory is accepted to ensure there will always be a lane estimation result.

For the first eight seconds of the roll-out, the ego vehicle is not yet driven into S1, but there are a few frames (at 4 second in Fig. 5.15) of incomplete lane detection due to the occlusion of other actor vehicles driving by. When the ego-vehicle is driving into the S1-harsh weather, the lane estimation parameters are mostly unreliable since the distorted input is not out of the training distribution from the perception model. It can be seen in the bottom of Fig. 5.15 that both ADE and WDE correctly reflect the occurrence of lane not matching events during this short period. It should be noted that the occlusion that happened for the first four seconds impacts the projected error, and relying solely on the

lane estimation error for ODD classification may raise more false alarms. It can be found that projected ADE is more sensitive to the obstruction and lane geometry change since it has constant weights on the forward-looking points of the lane. In contrast, the proposed WDE is not particularly sensitive to small fluctuations of lane parameter estimation of longer distance estimation and thus helps to reduce the impact of sudden misdetection due to the obstructions.

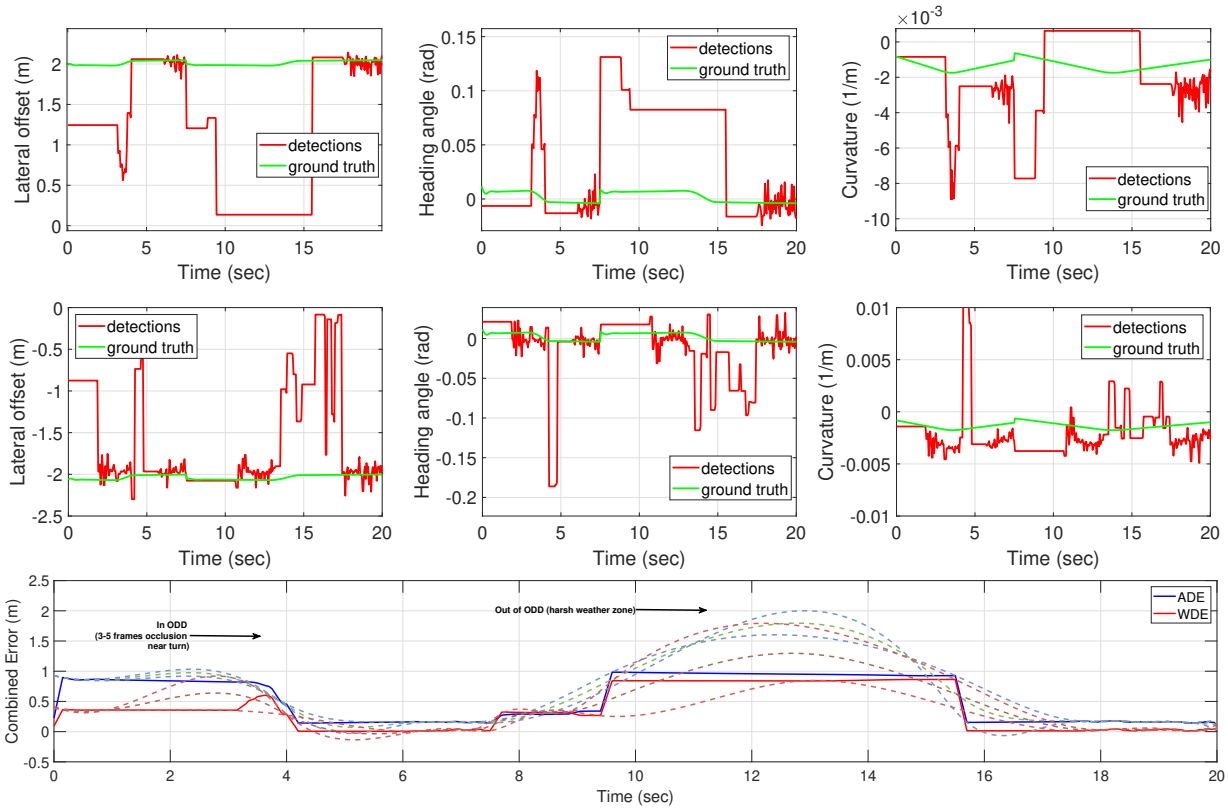


Figure 5.15: The lane estimation parameters and ground truth values when ego-vehicle driving from road curve into S1. (top: left lane parameters; middle: right lane parameters; bottom: projected error).

The projected error is calculated with window filtering by looking back in ten frames of detection results to make the generated results smoother and not be affected by the perception flickering in brief periods. Based on the observation in Fig. 5.15, it is clear that the projected error remains low when the perception model can correctly generate lane parameter predictions in consecutive frames. If there is a continuous loss track of lane

points, the lane error will be reflected quantitatively rather than reporting not available in our implementation.

Overall, the challenging weather will induce lane metric matching errors due to the perception module's miss detections that are stochastic and dependent on the specific version of the neural network. The proposed projected lane error metric can be used to obtain a quantitative measure for both false and miss detection cases for lane parameters. This projected measure is reflected as one input feature in the Bayesian classifier for the out-of-ODD warning generation.

Influence of the Localization and Map Error

The lane parameter in the vehicle frame depends on both the perception model and localization performance. In S2, Gaussian noise is applied to the vehicle localization, and the estimated lane parameters and corresponding ground truth in one roll-out are presented in Fig. 5.18. In the first five seconds, the vehicle is operating in S1, and the corresponding lane estimations are consistent. Afterward, the vehicle operates in S2, where the localization is injected with larger noise in both positioning and heading. The lane parameter estimation is based on the perception module; hence the predicted lanes are consistent with the ground truth since the perception is running in the normal condition. However, the queried reference based on the localization result differs from detection and ground truth after five seconds.

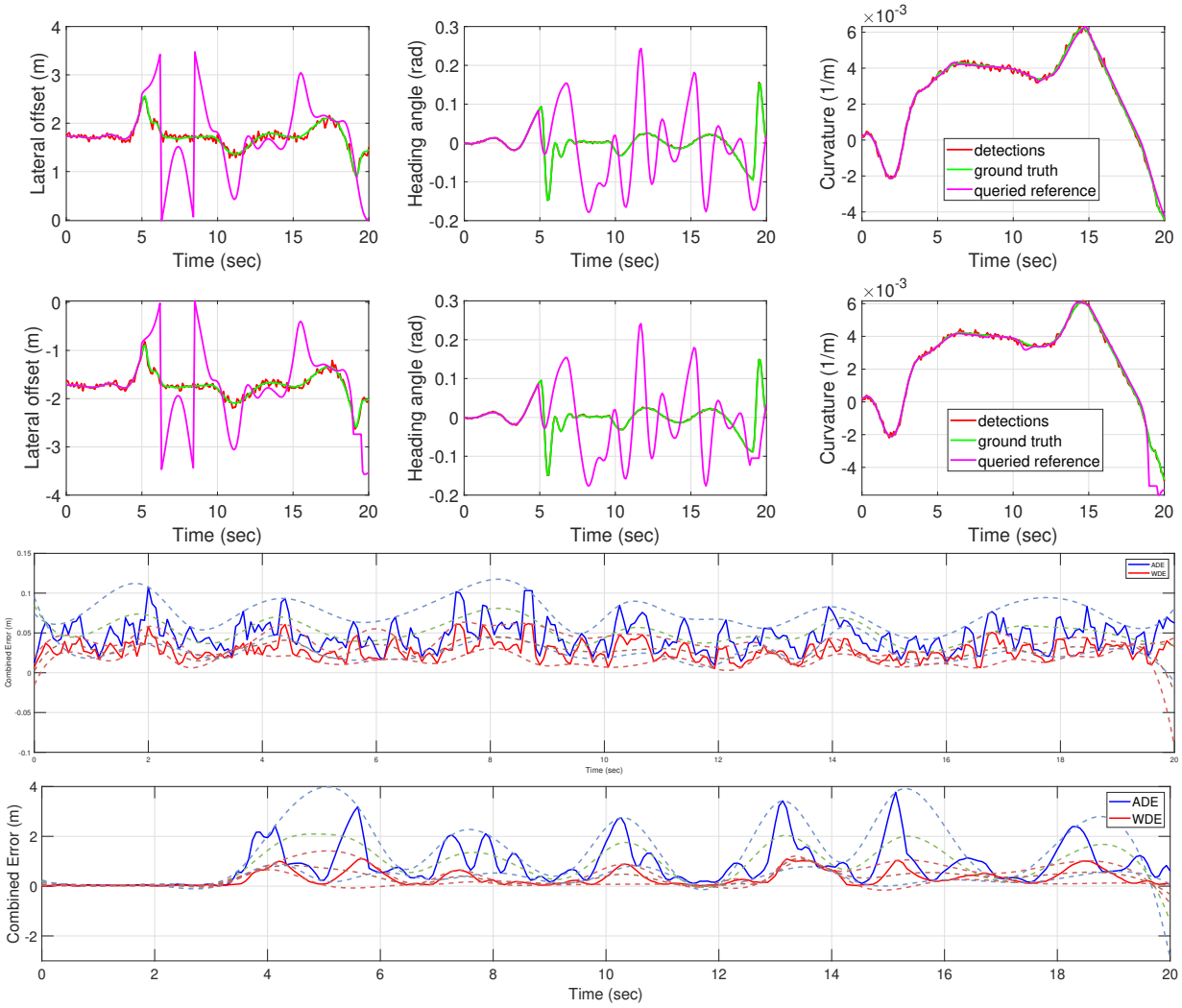


Figure 5.16: The lane estimation parameters and ground truth values when ego-vehicle driving from S1 into S2. (top: left lane parameters; middle: right lane parameters; bottom: projected error compared with ground truth and with queried reference).

Notice that the queried reference can be obtained in real-time using the coordinate transform [279]. The difference between the queried lane parameter and the ground truth is due to the localization error, which can be projected to the error in the extrinsic parameters, e.g., camera position or rotation. The two projected errors were computed in comparison with ground truth and with queried reference in Fig. 5.18. The combined error remains low with the ground truth, reflecting that the lane identification perception worked well. However, the ground truth lane parameters cannot be accessed in real-time when operating. Comparing the detected lane parameters with the queried reference extracted from the localization and map information, the combined error goes up to 2-3 m in the $[5, 20]$ period. In real-time operation, the queried reference is used such that the combined error shown in Fig. 5.18 can reflect the impact of the localization error.

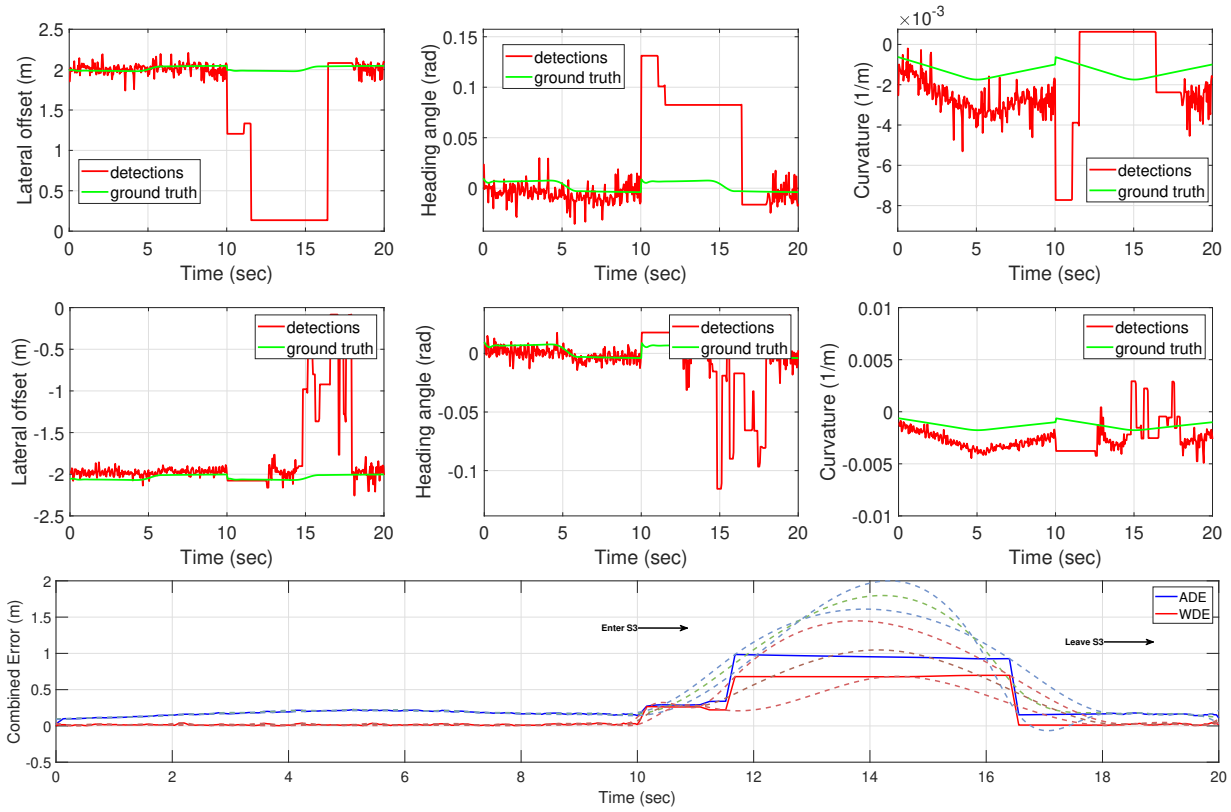


Figure 5.17: The lane estimation parameters and ground truth values when ego-vehicle driving in S3. (top: left lane parameters; middle: right lane parameters; bottom: projected error).

The accuracy of the queried reference is associated with localization and map accuracy since one acts as a key and the other as a dictionary. In the CARLA simulation, there is a consistent lane parameter error since only 2D information about the lane is used, whereas the CARLA map also has a height difference. This height difference will impact the S3 area with curve and slope road where the height difference goes up to 3.3 meters, which is two times more than the vertical alert limit mentioned in [278]. The lane estimation and the ground truth are plotted in Fig. 5.17. It can be found that during the $[10, 16.5]$ seconds period, the lane estimation is inconsistent with the ground truth. The source of such error comes from two sources, one is the misdetection due to the scenario change, especially with the curved road; the other is the projection error due to the two-dimensional model. Based on the combined error plot in Fig. 5.17, it shows that the map error combined with the perception misdetections can be well represented in both error metrics.

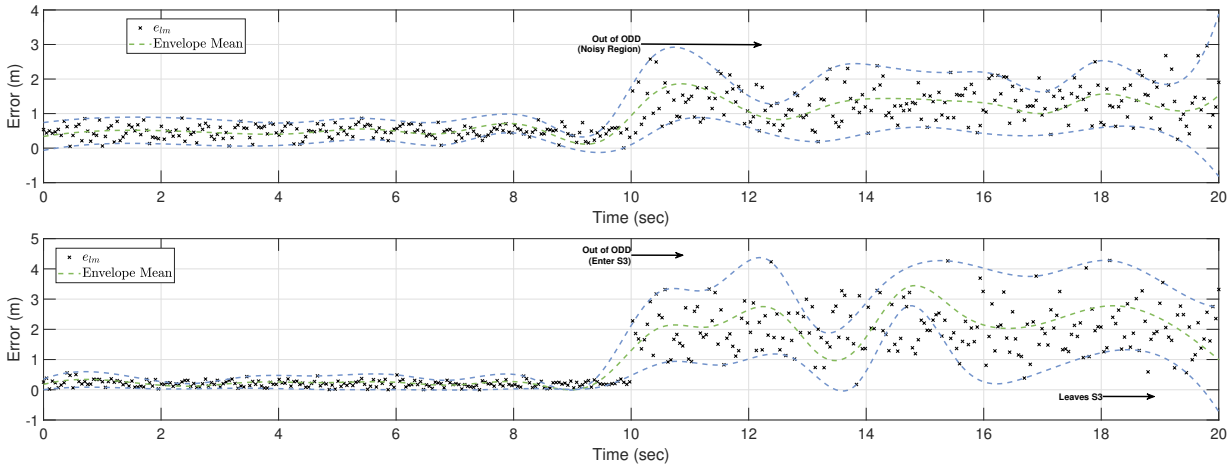


Figure 5.18: The landmark error when ego-vehicle exiting its ODD. (top: S2-noisy configuration; bottom: S3-map error).

The localization and map error also reflects in the landmark measurements. In Fig. 5.18, the landmark error is plotted for both S2 and S3 regions. In the noisy region, the localization and landmark displacement are the major sources of the landmark error since the error accumulates with the ego-landmark distance error. The measured error mean in the noisy region is about 1 meter which correctly reflects the simulation setting for the localization and landmark displacements in S2. The bottom plot in Fig. 5.18 depicts the landmark error changing from normal condition to S3 with simulation setting of the landmark displacement of 0.8 ± 0.4 m. In S3, the slope also includes the map error in such

a region, resulting in higher landmark errors even with lower localization errors than in S2. The error varies from 1-4 meters with a mean of 2 meters in the S3 scenario.

Additionally, it can be observed the time-lag effect in the landmark error comparing with the lane parameter estimation error when combining Fig. 5.17 and 5.18. At 18 seconds, the ego-vehicle is driving out of the S3 region, where the combined error in lane estimation in Fig. 5.17 is already approaching zero, reflecting that the vehicle is driving at normal conditions. However, since the landmark error is computed with surrounding fixed landmarks in a given region, those landmarks with larger displacement errors will impact the landmark error after the vehicle just leaves S3.

5.5.4 Classifier Performance Analysis

Based on Section 5.5.1, three forms of classifiers are considered: the geo-fencing checker, the rule-based redundancy checker, and the BN classifier. The classifier maps the measure values onto the set for the warning signals to tell the current state of the ego-vehicle is $\{\textbf{in ODD}, \textbf{out of ODD}\}$.

Classifier Definition

The geo-fencing and rule-based checker are straightforward and well-defined in Section 5.5.1. Two forms of the BN classifier are illustrated here. The first one is the "Static Bayesian Net" where the posterior distribution of the ODD valid is computed based on the Conditional Probability Table (CPT) that propagates with the probabilistic graphical model illustrated in Fig. 5.9. The CPT used for propagating the probability in the static Bayesian Net model is chosen in the same form from Fig. 5.11. Notice that in this setting, the CPT is predefined parameters and not related to any real-time measurements. The conditional probabilities can be extracted from the offline testing procedure mentioned in Chapter 4.

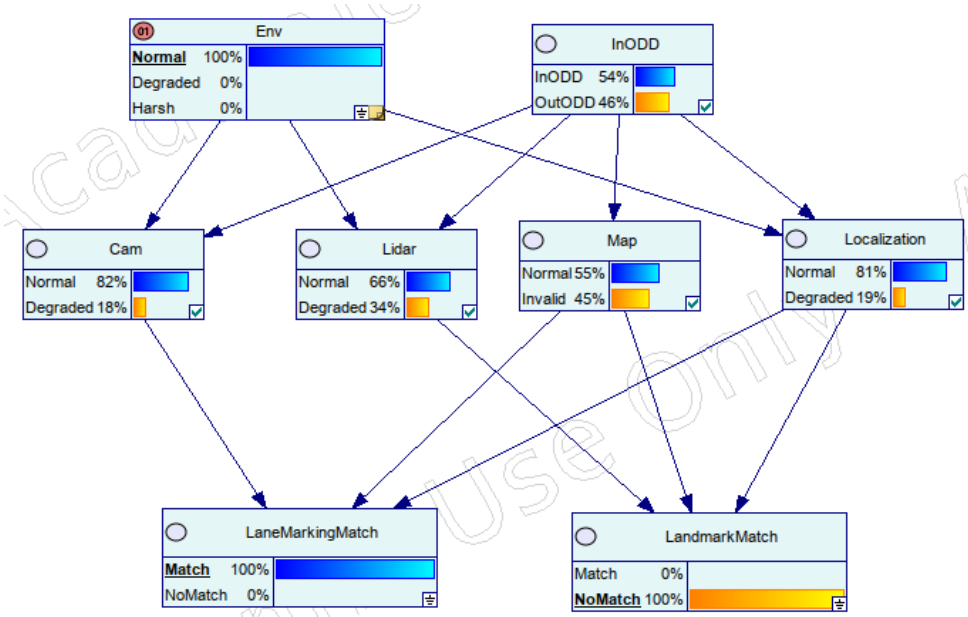


Figure 5.19: The example Bayesian Network with fixed evidence.

The other form of the BN classifier shares the same structure as the previous one, but the confidence score from perception and the localization module is accepted as visible evidence in the BN. The other difference is that the last layer of the evidence node is modeled with discrete probability rather than the Boolean label. The example BN network is presented in Fig. 5.19, where the only evidence available is the basic environment information, "Lane Marking Match" and "Landmark Match". Notice that whether there is a match is dependent on the measured error. The mapping from the measurement error to the probability of the evidence node to be matched can be modeled as a piece-wise or sigmoid-shaped function as (5.18), (5.19) listed below. The piece-wise function (5.18) maps the error to the probability of one if the error is in the lower bound $[0, a]$, and maps to probability zero if the error is greater than the upper bound b.

$$y = \begin{cases} 1, & x \in [0, a] \\ \frac{1}{a-b}(x-1), & x \in (a, b] \\ 0, & x > b. \end{cases} \tag{5.18}$$

Notice that if $a = b$ in (5.18), then the evidence for matching is a Boolean value with either 0 or 1. The sigmoid function form (5.19) can map the positive number to $[0, 1]$ with the shape controlled by k and i where the former controls the steepness of the function and the latter controlled where the mapping goes to 0.5.

$$y = \frac{1}{1 + e^{k(x-i)}}, \; x \geq 0. \tag{5.19}$$

Example function mapping is depicted in Fig. 5.20 where 0.3 m is chosen as the lower bound and 0.7 m as the upper bound based on the qualitative analysis in Sec. 5.5.3 and the requirements mentioned in Appendix A. In this research, the "Parameterized Bayesian Net" is using the sigmoid-like function as the mapping between error and probability.

Effect of Threshold Parameter and Error Metrics

The driving data collected from the CARLA virtual environment in Sec. 5.5.1 are split into 1s segments with labels $\{\textbf{in ODD}, \textbf{out of ODD}\}$ based on its running scene and the sensor noise setting. In this section, all driving segments in S1-3 are set with the **out of ODD** label, and the combined error is computed as the average of the twenty frames in one second. In order to evaluate the classifier with proper balance, two sets

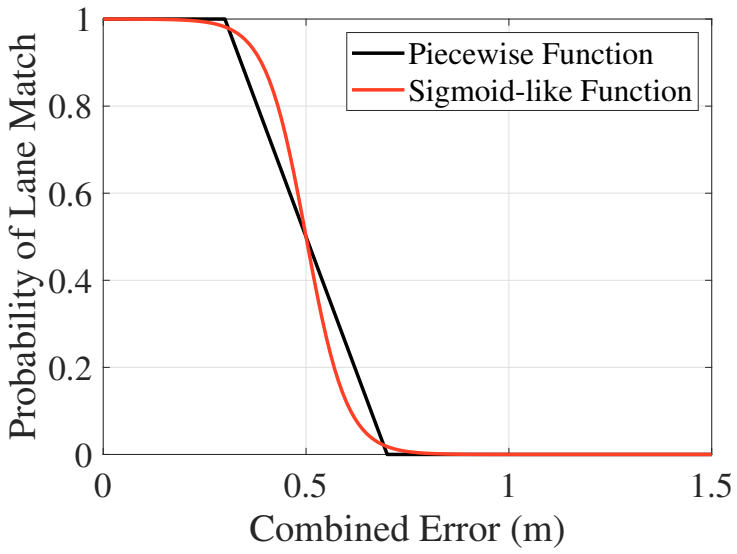


Figure 5.20: The example mapping function from measured error to the probability of a correct pattern match, ($a = 0.3,\ b = 0.7,\ k = 20, i = 0.5$).

total of one hundred segments with half of them **in ODD** and another half of them out as ground truth are selected.

For the geo-fencing classifier, it will categorize all the segments in the pre-specified region S3. Thus the classifier performance evaluated depends on how many cases from S1 and S2 are randomly chosen in the evaluation set since they will contribute to the False Negatives. The performance of the geo-fencing classifier is plotted in Fig. 5.21 as a single point since the threshold parameter does not control it. The geo-fencing classifier achieved zero false positive rates since it will never predict outside ODD other than the predefined region.

The rule-based checker aims to check the neural network uncertainty and the localization variance is greater than the predefined bound. In this specific test case, it is assumed that no localization covariance is provided, and the last layer activation from the neural network is used as the uncertainty indicator. Thus, the rule-based checker will generate a warning whenever the perception system has low confidence ($\leq 70\%$) of the lane prediction. The neural network "uncertainty" drops not only in S1 but also in some segments in S3 when the lane marking is not evident in the image. This behavior explains the 60% and 50% true positive rate; and 36% and 40% false positive rate depicted in Fig. 5.21 (yellow cross mark).

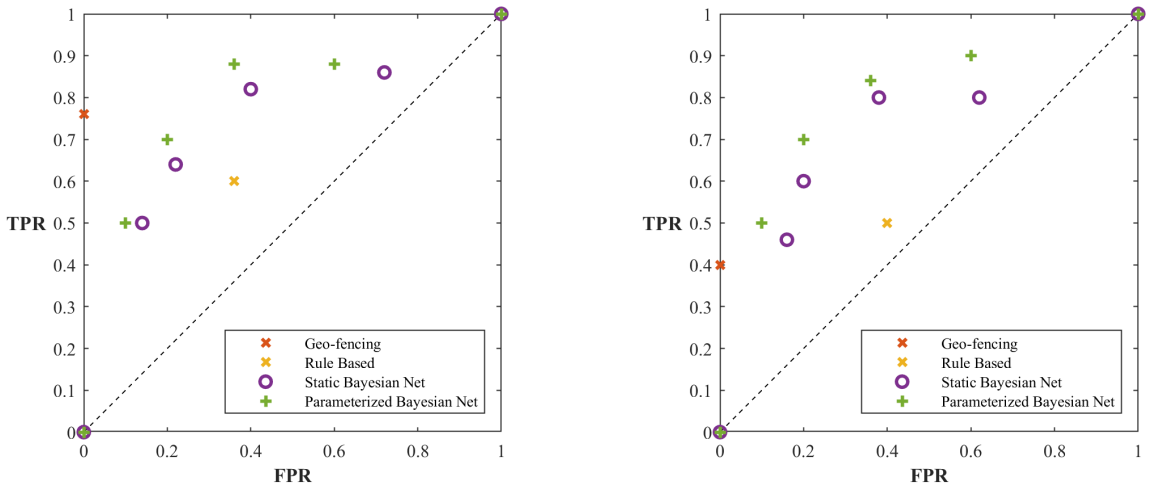


Figure 5.21: The ROC curve over different values of the threshold parameter $\lambda = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ in the CARLA testing scenario. (left: test set with more segments in S3, right: test set with more segments in S1-2)

The performance of the two BN classifiers is also presented in Fig. 5.21. The TPR and FPR decrease step-wise with the increasing threshold parameter value λ. Based on the observation, it can be observed that the parameterized Bayesian Net outperforms the static Bayesian Net with all the λ settings and benefit from the virtual probability evidence. The proposed BN classifier can outperform the geo-fencing and rule-based binary classifiers with multiple choices of the threshold parameter, and the model is relatively insensitive to both λ and the dataset chosen.

5.6 Experiment on Real Operation Data

The experiment was conducted using the rosbag collected using the WatonoBus platform on the ring road in the Waterloo region. The details of the sensor and software setups can be further found in Appendix A.6. The center camera images are collected at 20 frames per second. The localization results are obtained with lower than 0.1 m error which corresponds to very accurate results and is used as our reference results. The ring road reference lanes are pre-collected and stored such that the map is assumed to be perfect in this experiment.

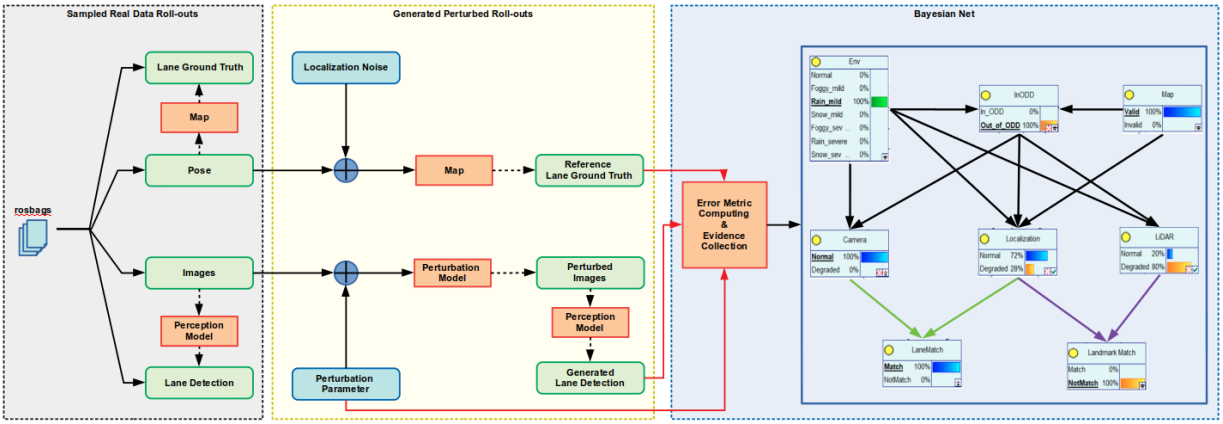


Figure 5.22: The data processing pipeline for the shuttle bus driving data.

5.6.1 Data Roll-outs Configuration and Perturbation

In the experiment, the real driving data collected in autonomous mode from the WatonoBus platform is processed using the procedure illustrated in Fig. 5.22. Every five consecutive frames and corresponding vehicle ground truth pose are extracted as one single reference roll-out. The reference roll-out shares the ground truth lane detection from the pose-map query. These anchor roll-outs are classified as **in ODD** if there is a clear lane mark, otherwise, **out of ODD**. Sample image examples are presented in Fig. 5.23. Once the region of these data has been marked, the geofencing-based classifier is defined to identify all the roll-outs in those areas without a clear lane mark as **out of ODD**.

The perturbed roll-outs are generated by randomly applying localization noise on the pose results for each anchor roll-out data and adding the perturbation to the images.

Figure 5.23: The example center camera image is classified as in/out of ODD based on geofencing.

The localization noise added will eventually reflect in the erroneous "reference lane ground truth". At the same time, the image perturbation model is the same one used and evaluated in Sec. 4.2. The lane detector extracts the estimated lane parameters from the perturbed image to mimic the detections in difficult foggy, rainy situations. The ground truth label for **in ODD** or not of these generated roll-outs is decided by the localization noise and the perturbation parameter. The generated roll-outs are set as **out of ODD** if the norm of localization noise is greater than 0.3 m or with the perturbation level over one. Otherwise, the ground truth label is then **in ODD**. The error metrics are computed for each roll-out, and the perturbation level is considered evidence to feed into the PGM. Additionally, the confidence score from the perception algorithm is used as an additional factor for the parameterized Bayesian Net classifier.

5.6.2 Qualitative Analysis

The sample images and corresponding lane detections are presented in Fig. 5.24 and 5.25. The anchor data corresponding to the raw image data is collected from the rosbag. The lane detection algorithm (Appendix A.5) detects the lane features, and the detected lane points are also drawn in corresponding images. Both left, center and right lane features are detected; however, only the green points (center lane) features are used for the next step of lane parameter regression [280]. The green text on each image shows the "confidence score" from the neural network for the lane classification task. It should be noted that the confidence is always greater than 50% for those images that can obtain some lane estimation, even the erroneous one, since the last layer activation predicted that those are classified as lane features. This indicates that the neural network's "confidence score" is not reliable for evaluating the actual perception error.

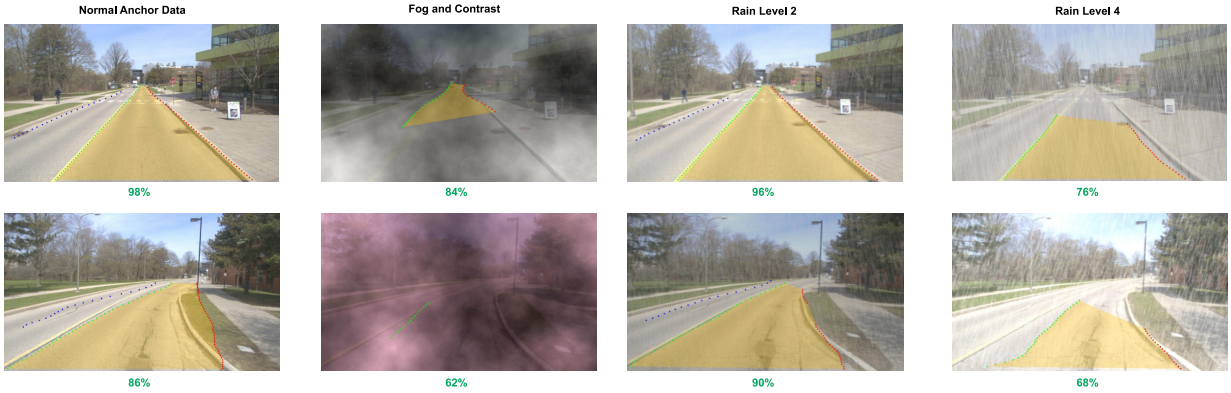


Figure 5.24: The sample reference roll-out data and the corresponding augmentation and raw lane detection results (fog, contrast, and rain). The green text below reflects the last layer sigmoid function output from the neural network for lane classification for each image.

Fig. 5.25 presents the sample images for snow type perturbation and the corresponding lane detection results. The final lane estimation will utilize the extracted green center lane points, project them to the vehicle coordinate, and fit the lane equation (see Appendix A.3). The top right of Fig. 5.25 has the center lane classification score is only 12% that resulted in the miss detection. Even though the confidence score is not a reliable source for evaluating the overall perception performance, this redundant evidence still can be used to construct the image-based perception module's relative performance as a conditional probability, e.g., $P(\text{PM}|\text{environment})$.

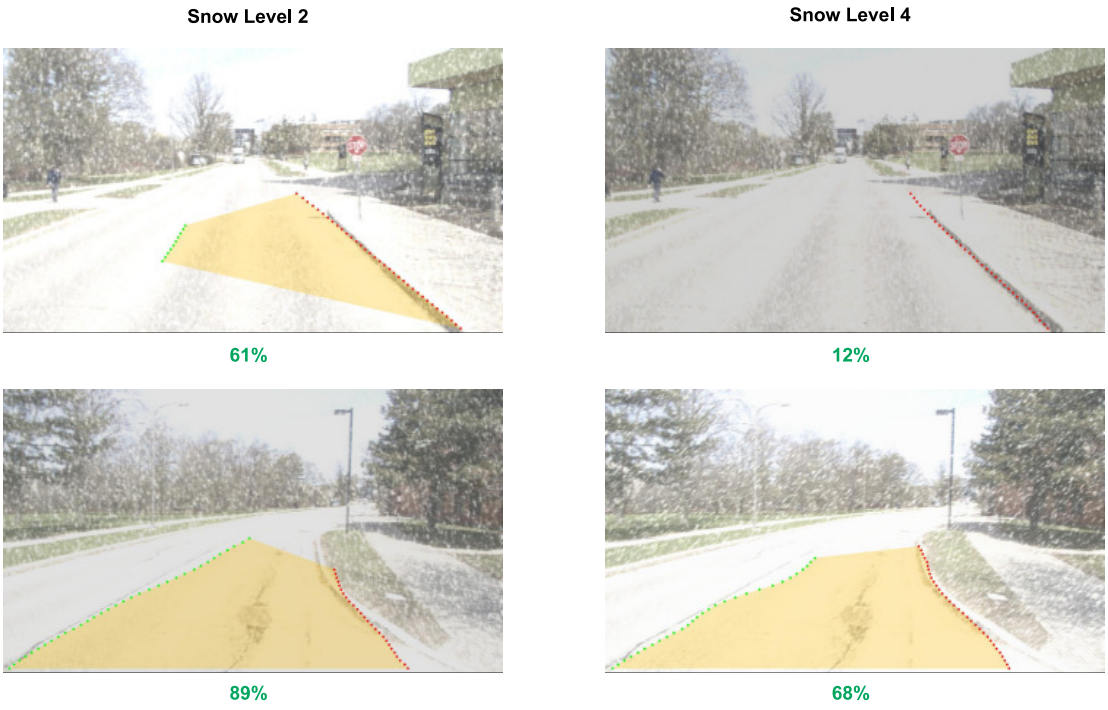


Figure 5.25: The sample snow augmentation and raw lane detection results. The green text below reflects the last layer sigmoid function output from the neural network for lane classification for each image.

5.6.3 Classifier Evaluation

The classifier examined in this part for the driving data collected from the WATonoBus platform is similar to those evaluated in Sec. 5.5.4. The BN structure is slightly different, as shown in Fig. 5.22 where the image-based perception module has the conditional probability table set as Table. 5.3. The conditional probabilities are priors obtained through the offline testing process mentioned in Chapter 4. The operating conditions impact the perception algorithm accuracy, which reflects on the conditional probabilities. The confidence score from the neural network is used as a multiplier here in Table. 5.3 since it reflects the current uncertainty that the perception module estimates when comparing the current frame to the training frame distribution.

Additionally, the conditional probability table transit from the camera and localization nodes to the lane feature matching is presented in Table. 5.4. The lane matching is only

Table 5.3: Example conditional probabilities of image-based perception module.

Variable Node/ Probability	**Environment**	Normal		Foggy		Rain		Snow	
	In ODD	True	False	True	False	True	False	True	False
Camera Perception	Match	0.95	0.5	0.9	0.5	0.6	0.1	0.45	0.02
	Not Match	0.05	0.5	0.1	0.5	0.4	0.9	0.55	0.98

associated with localization and camera module under the assumption of a perfect map.

Table 5.4: Example conditional probabilities of lane marking match.

Variable Node/Probability	**Camera**	Normal		Degraded	
	Localization	Normal	Degraded	Normal	Degraded
Lane Match	Match	0.95	0.2	0.3	0.01
	Not Match	0.05	0.8	0.7	0.99

The classifiers' performances are plotted on the ROC curve shown in Fig. 5.26, with the different discrete settings of the threshold parameter. The geofencing classifier achieved around 43% TPR and 24% FPR. The geofencing classifier's low TPR represents the ratio of possible perception failures in the predefined geofencing region, whereas the FPR represents cases where the predefined region cannot represent well due to ODD or localization noise. The further detailed cause of the error is presented in Table 5.5 where the recall (TPR) remains the same for both localization degradation and perturbation of the images. There is a drop in precision in the subset where the longitudinal noise is generated on the second data column of Table 5.5. Because the geo-fencing classifier relies on the predefined region and localization accuracy, precision over the subset of longitudinal localization displacement is reduced.

In Fig. 5.26, the rule-checking classifier obtains slightly better results than the geo-fencing classifier, which reports about 26% FPR. The TPR of the rule checking classifier is 56%, where half the cases when the system is **out of ODD** missed due to the over-trust on the "confidence score". It is not surprising that the rule-checking classifier performed consistently in subsets perturbed with perception degradation. This is because the "confidence score" not always reliable, as discussed earlier. It is worth mentioning that the classifier should be considerably sensitive to the heading error noise in actual operation data compared to the generated degradation since the perturbation in localization here does not change the raw image data observed. However, as illustrated in the virtual data,

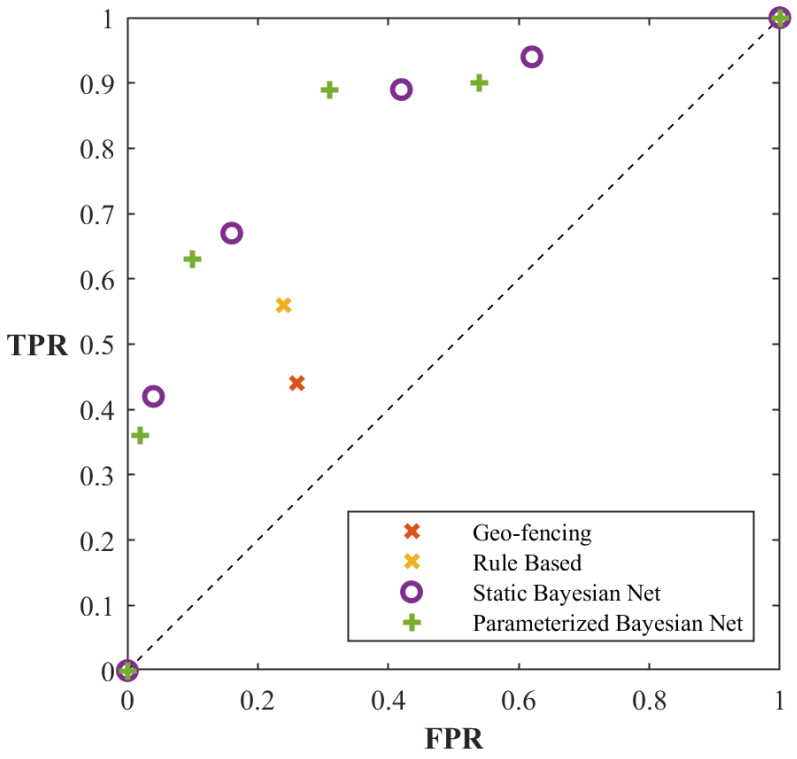


Figure 5.26: The ROC curve over different values of the threshold parameter $\lambda = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ (from top right to left bottom).

the heading error or lateral displacements will sometimes result in image observations with no proper lane features.

Table 5.5: Precision and Recall on the subsets of roll-outs with specific perturbation.

Precision/Recall	Localization Degradation			Perception Degradation			
	Lat. Displacement	Long. Displacement	Heading Error	Fog-Lv1	Fog-Lv2	Rain-Lv1	Rain-Lv2
Geo-fencing	0.31/0.43	0.28/0.43	0.31/0.43	0.31/0.43	0.31/0.43	0.31/0.43	0.31/0.43
Rule-Checking	0.46/0.58	0.48/0.4	0.39/0.48	0.46/0.58	0.41/0.6	0.59/0.6	0.44/0.43
Static Bayesian Net	0.64/0.68	0.71/0.68	0.78/0.72	**0.67/0.73**	0.78/0.51	0.74/0.78	0.82/0.76
Parameterized Bayesian Net	0.64/0.71	0.71/0.7	0.78/0.77	0.66/0.69	**0.86/0.89**	**0.77/0.84**	**0.91/0.90**

By employing offline testing knowledge, the PGM-based classifiers can better determine if the system operates in safe conditions. The discrete ROC curve for the two Bayesian net methods is presented in Fig. 5.26. The static Bayesian Net usually gets a lower FPR, whereas the parameterized Bayesian classifier gets a higher TPR. The higher TPR results from the additional multiplier from the "confidence score" that makes more samples detected as **out of ODD**. For the same reason, the FPR is relatively higher in the parameterized Bayesian case. For a proper comparison with the other two methods, the threshold $\lambda = 0.6$ are selected. Classifiers are applied to the subsets of different perturbations to examine the performance, as shown in Table 5.5. The two PGM-based classifiers achieve similar performance in the generated roll-out subsets under the localization degradation condition. There is a slight improvement in recall in the parameterized Bayesian classifier, where the confidence score in those typical frames is low when the roll-out sample is injected with more significant localization noise. The major difference is when perception degradation reflects the weather change at the deployment stage, which the environmental evidence might not directly identify. Both PGM-based methods get about a 30-50% performance boost compared to conventional rule-checking methods. The real-time measurement of the lane mark matching error can significantly help the classifier understand the current environment. The parameterized Bayesian net classifier gets the highest precision and recall performance in three out of four augmentation subsets.

In summary, the effectiveness of the PGM-based classifier is shown in both virtual and real data in evaluating the system's reliability, especially in scenarios where the reference evidence can be measured. Offline knowledge of conditional dependency can improve such prediction performance even further.

5.7 Discussion

Relying solely on a rule-based approach to classify the system's reliability often results in less accurate predictions and may lead to overlooking potential hazards in real operations. The proposed solution is to make a reference ground truth by encoding offline knowledge and real-time measurements into a model. The PGM can be effectively used for this ODD monitoring purpose, especially for generating soft decision boundaries compared to the commonly used geofencing and rule-based checkers. Even though the embedded expert knowledge on the prior probability dependency on different nodes may require additional efforts, the process is well aligned with the offline testing procedure mentioned in previous chapters. Furthermore, the proposed models utilize evidence from vastly heterogeneous sources (sensors, weather changes) and fuse it into one Bayesian network. Additionally, the posterior distribution for all sources in the model is provided, which can be further used for fault detection and isolation from different sources.

Some extra work is still needed for the industrial design of such an ODD monitoring system. For the prior estimation part, proper calibration is needed for the reference redundancy module and the monitored module. This thesis assumes a strong correlation between the perception modules using the same hardware set. However, a detailed calibration would improve the model's accuracy further in the conditional distribution design as priors. Additionally, online learning techniques could further improve this prior, which can update while operating. Another potential improvement is the classifier fusion since the goal is to have a higher TPR and lower FPR with more importance on the low FPR for the "out of ODD" warning. The fusion between geo-fencing and the PGM-based method might improve the robustness of the predictions.

Chapter 6

Summary and Outlook

We summarize the content of the previous chapters and state our main contributions in the following. The conclusions for ODD testing and monitoring are presented based on our experimental results. Finally, we discuss the open issues and potential future directions for extending our work.

6.1 Summary and Contributions

The first chapter presents the background and motivation for investigating the ODD of self-driving vehicles. The ODD is critical to improving overall driving safety and comfort for both ADS and the driver. Following this motivation, the detailed research challenges related to the clear and unified ODD definition and the procedure to extract and monitor such ODD for the ADS are the focus of this dissertation. Based on our description of related background information in Chapter 2, the following contributions to this thesis are listed.

6.1.1 Contributions

The presented work identified research challenges in the three major building blocks related to ODD (Extraction, Augmentation, and Real-time Monitoring) of the ADS in the development and deployment cycle. Our proposed contributions to resolving them, as well as the conclusions reached, are organized accordingly.

Extracting the Operational Design Domain for Autonomous Driving Systems

This dissertation addresses the strategy to extract and enhance the ODD for an ADS as a continuous engineering problem. The six-layer environmental model is proposed to describe the ODD in a unified and quantifiable way. The proof of concept has shown promising directions for future work on ODD monitoring and the applications in iterative development for ADS. In addition to the proposed framework, an implementation of the proposed approach is examined with two learning-based agents to demonstrate its feasibility. This work is majored organized in Chapter 3 and has already been published as a journal paper in [281].

Safety Requirements and Validation Methods for Autonomous Driving Systems

The verification and validation are significant in ensuring the safety of automated vehicles. The published review [282] investigates the various techniques implemented in the decision-making and planning for autonomous vehicles in terms of safety, comfort, and energy optimization. The approaches based on scenario-based testing, fault injection testing, and formal verification are reviewed, and six criteria are proposed to compare and evaluate the characteristics of the V&V approaches. The decision-making module in the ADS is broken down into a module hierarchy, and each module's functional requirements are deduced.

In Chapter 4 of this dissertation, the validation and improvement process for the perception module is laid out. The robust training methods proposed to handle the new data distribution for perception modules of the ADS. The results are under preparation for publication.

Real-time Monitoring of the Operational Design Domain

Estimating the real-time ODD for the perception modules embedded with data-driven functions is challenging, primarily due to the lack of labelled data in real-time operations. Following our study of related work regarding existing concepts of geo-fencing and rule-based checking of the operating conditions, the model-based method incorporating the map and localization information to generate a real-time reference can be considered helpful in real-time prediction of the perception module's performance. So, the PGM-based strategy is proposed to use offline knowledge and measurements of landmarks and lane

markings during operation time, as described in Chapter 5. The PGM stores the offline knowledge of causal dependencies in a graph model and uses the real-time measurement from the redundancy module to estimate the performance of the module.

Additionally, the experiments are conducted with both virtual and real driving data that demonstrate the effectiveness of the proposed solution. The effects of different perturbations and metric choices on the performance of the ODD monitor are showcased in Chapter 5. A parameterized PGM-based classifier has been shown to improve performance. The results are under preparation for publication.

6.2 Outlook

The contributions and results presented in this thesis motivate further research on the ongoing formulation and monitoring of the ODD in terms of the safety aspects of self-driving vehicles with data-driven modules. Even though the capabilities of the autonomous vehicle have steadily increased over the years, there will still be limits and boundaries to confine the vehicle's behaviour and warn drivers to take over in "extreme" driving situations. The extraction, augmentation, and real-time monitoring for the ODD of the self-driving vehicle functions presented in this dissertation help to solve such issues to ensure a high level of safety.

Letter of Copyright Permission

IEEE, as the publisher of the two manuscripts fully or partly adopted in Chapter 2 and Chapter 3, allows the reuse of published papers in the thesis without formal permissions. Thus, the waivers of copyright from IEEE are achieved by the following statement:

Policy Regarding Thesis/Dissertation Reuse, from IEEE Copyright Clearance Centre

"The IEEE does not require individuals working on a thesis to obtain a formal reuse license; however, you may print out this statement to be used as a permission grant".

References

[1] R. Palin, D. Ward, I. Habli, and R. Rivett, "Iso 26262 safety cases: Compliance and assurance," 2011.

[2] J. M. Scanlon, R. Sherony, and H. C. Gabler, "Preliminary potential crash prevention estimates for an intersection advanced driver assistance system in straight crossing path crashes," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1135–1140, IEEE, 2016.

[3] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3234–3243, 2016.

[4] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16, 2017.

[5] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4340–4349, 2016.

[6] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and service robotics*, pp. 621–635, Springer, 2018.

[7] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 172–189, 2018.

[8] D. Temel, G. Kwon, M. Prabhushankar, and G. AlRegib, "CURE-TSR: Challenging unreal and real environments for traffic sign recognition," in *Neural Information Processing Systems (NeurIPS) Workshop on Machine Learning for Intelligent Transportation Systems*, 2017.

[9] D. Temel, G. Kwon, M. Prabhushankar, and G. AlRegib, "CURE-TSR: Challenging unreal and real environments for traffic sign recognition," in *Neural Information Processing Systems (NeurIPS) Workshop on Machine Learning for Intelligent Transportation Systems*, 2017.

[10] M. Sayers, "Standard terminology for vehicle dynamics simulations," *The University of Michigan Transportation Research Institute (UMTRI), Tech. Rep*, 1996.

[11] D. Cualain, C. Hughes, M. Glavin, and E. Jones, "Automotive standards-grade lane departure warning system," *IET Intelligent Transport Systems*, vol. 6, no. 1, pp. 44–57, 2012.

[12] S. Standard, "J3016," *Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems*, vol. 4, pp. 593–598, 2014.

[13] F. W. Geels, "The dynamics of transitions in socio-technical systems: a multi-level analysis of the transition pathway from horse-drawn carriages to automobiles (1860–1930)," *Technology analysis & strategic management*, vol. 17, no. 4, pp. 445–476, 2005.

[14] M. Maurer, J. C. Gerdes, B. Lenz, H. Winner, *et al.*, *Autonomous driving*. Springer, 2016.

[15] F. Kröger, "Automated driving in its social, historical and cultural contexts," in *Autonomous Driving*, pp. 41–68, Springer, 2016.

[16] J. Hammond and E. Purington, "A history of some foundations of modern radio-electronic technology," *Proceedings of the IRE*, vol. 45, no. 9, pp. 1191–1208, 1957.

[17] G. Motors, "An automatically guided automobile cruised along a one-mile check road at general motors technical center today," *Pressemitteilung, USA*, 1953.

[18] L. Jackel, H. Graf, W. Hubbard, J. Denker, D. Henderson, and I. Guyon, "An application of neural net chips: Handwritten digit recognition," in *ICNN proceeding*, pp. 11–10745, 1988.

[19] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition: neural networks vs. hidden markov models vs. hidden markov models," in *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on*, pp. 107–110, IEEE, 1988.

[20] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in neural information processing systems*, pp. 305–313, 1989.

[21] M. Buehler, K. Iagnemma, and S. Singh, *The DARPA urban challenge: autonomous vehicles in city traffic*, vol. 56. springer, 2009.

[22] A. Broggi, P. Cerri, S. Debattisti, M. C. Laghi, P. Medici, D. Molinari, M. Panciroli, and A. Prioletti, "Proud—public road urban driverless-car test," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3508–3519, 2015.

[23] S. Gibbs, "Google sibling waymo launches fully autonomous ride-hailing service," *The Guardian*, vol. 7, 2017.

[24] S. Ingle and M. Phute, "Tesla autopilot: semi autonomous driving, an uptick for future autonomy," *International Research Journal of Engineering and Technology*, vol. 3, no. 9, 2016.

[25] D. A. Hennessy and D. L. Wiesenthal, "Traffic congestion, driver stress, and driver aggression," *Aggressive Behavior: Official Journal of the International Society for Research on Aggression*, vol. 25, no. 6, pp. 409–423, 1999.

[26] M. Berk, O. Schubert, H.-M. Kroll, B. Buschardt, and D. Straub, "Assessing the safety of environment perception in automated driving vehicles," *SAE International Journal of Transportation Safety*, vol. 8, no. 09-08-01-0004, 2020.

[27] P. Koopman and M. Wagner, "Toward a framework for highly automated vehicle safety validation," tech. rep., SAE Technical Paper, 2018.

[28] S. O.-R. A. D. Committee *et al.*, "Sae j3016. taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," tech. rep., tech. rep., SAE International, 2016.

[29] J. Vokrinek, M. Schaefer, and D. Pinotti, "Multi-agent traffic simulation for human-in-the-loop cooperative drive systems testing," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pp. 1691–1692, International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[30] T. M. Gasser, A. Seeck, and B. W. Smith, "Framework conditions for the development of driver assistance systems," 2019.

[31] P. Lin, "Why ethics matters for autonomous cars," in *Autonomous driving*, pp. 69–85, Springer, Berlin, Heidelberg, 2016.

[32] S. Sharma, A. Flores, C. Hobbs, J. Stafford, and S. Fischmeister, "Safety and security analysis of aeb for l4 autonomous vehicle using stpa," in *Workshop on Autonomous Systems Design (ASD 2019)*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[33] B. Mirchevska, C. Pek, M. Werling, M. Althoff, and J. Boedecker, "High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2156–2162, IEEE, 2018.

[34] M. Gharib, P. Lollini, M. Botta, E. Amparore, S. Donatelli, and A. Bondavalli, "On the safety of automotive systems incorporating machine learning based components: a position paper," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pp. 271–274, IEEE, 2018.

[35] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2722–2730, 2015.

[36] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2018.

[37] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[38] G. Wesiak, C. M. Steiner, A. Moore, D. Dagger, G. Power, M. Berthold, D. Albert, and O. Conlan, "Iterative augmentation of a medical training simulator: Effects

of affective metacognitive scaffolding," *Computers & Education*, vol. 76, pp. 13–29, 2014.

[39] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pp. 982–988, IEEE, 2015.

[40] J. Chen, P. Zhao, H. Liang, and T. Mei, "A multiple attribute-based decision making model for autonomous vehicle in urban environment," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pp. 480–485, IEEE, 2014.

[41] G. J. Hains, A. Jakobsson, and Y. Khmelevsky, "Formal methods and software engineering for dl. security, safety and productivity for dl systems development," *arXiv preprint arXiv:1901.11334*, 2019.

[42] NHTSA, "Automated driving systems 2.0: A vision for safety," 2017.

[43] T. Canada, "Canada's safety framework for automated and connected vehicles," 2019.

[44] Waymo., "On the road to fully self-driving: Waymo safety report," 2017.

[45] Tesla, "Tesla vehicle safety report," 2017.

[46] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a formal model of safe and scalable self-driving cars," *arXiv preprint arXiv:1708.06374*, 2017.

[47] R. Palin, D. Ward, I. Habli, and R. Rivett, "Iso 26262 safety cases: Compliance and assurance," 2011.

[48] R. Palin, D. Ward, I. Habli, and R. Rivett, "Iso pas 21448: Road vehicles - safety of the intended functionality," 2019.

[49] D. D. Ward and S. E. Crozier, "The uses and abuses of asil decomposition in iso 26262," in *7th IET International Conference on System Safety, incorporating the Cyber Security Conference 2012*, pp. 1–6, IET, 2012.

[50] S. Shammas, *The Future of Driving: A Look Into the Technology Behind Autonomous Vehicles*. PhD thesis, 2017.

[51] B. Templeton, "Ntsb report on tesla autopilot accident shows what's inside and it's not pretty for fsd," Sep 2019.

[52] A. Lubben, "Self-driving uber killed a pedestrian as human safety driver watched," Mar 2018.

[53] Z. Cao and A. A. Ceder, "Autonomous shuttle bus service timetabling and vehicle scheduling using skip-stop tactic," *Transportation Research Part C: Emerging Technologies*, vol. 102, pp. 370–395, 2019.

[54] H. Hungar, F. Köster, and J. Mazzega, "Test specifications for highly automated driving functions: Highway pilot," 2017.

[55] P. Irvine, X. Zhang, S. Khastgir, E. Schwalb, and P. Jennings, "A two-level abstraction odd definition language: Part i," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2614–2621, IEEE, 2021.

[56] B. Spanfelner, D. Richter, S. Ebel, U. Wilhelm, W. Branz, and C. Patz, "Challenges in applying the iso 26262 for driver assistance systems," *Tagung Fahrerassistenz, München*, vol. 15, no. 16, p. 2012, 2012.

[57] M. Thomason and R. Gonzalez, "Data structures and databases in digital scene analysis," in *Advances in Information Systems Science*, pp. 1–47, Springer, 1985.

[58] M. Maurer, "Ems-vision: knowledge representation for flexible automation of land vehicles," in *Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No. 00TH8511)*, pp. 575–580, IEEE, 2000.

[59] S. Geyer, M. Baltzer, B. Franz, S. Hakuli, M. Kauer, M. Kienle, S. Meier, T. Weißgerber, K. Bengler, R. Bruder, *et al.*, "Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance," *IET Intelligent Transport Systems*, vol. 8, no. 3, pp. 183–189, 2013.

[60] X. Zhang, S. Khastgir, and P. Jennings, "Scenario description language for automated driving systems: A two level abstraction approach," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 973–980, IEEE, 2020.

[61] P. Koopman and F. Fratrik, "How many operational design domains, objects, and events?," 2019.

[62] P. Koopman, U. Ferrell, F. Fratrik, and M. Wagner, *A Safety Standard Approach for Fully Autonomous Vehicles*. Edge Case Research Pittsburgh PA 15201 USA Carnegie Mellon University Pittsburgh PA 15213 USA; The MITRE Corporation McLean VA 22102 USA; Edge Case Research Pittsburgh PA 15201 USA.

[63] M. Maiouak and T. Taleb, "Dynamic maps for automated driving and uav geofencing," *IEEE Wireless Communications*, vol. 26, no. 4, pp. 54–59, 2019.

[64] I. 34503:2021(E), "Road vehicles — Taxonomy for operational design domain for automated driving systems," standard, International Organization for Standardization, Geneva, CH, Mar. 2021.

[65] British-Standard-Institution, "Operational design domain (odd) taxonomy for an automated driving system (ads) - specification," 2021.

[66] P. Koopman, "An overview of draft ul 4600:'standard for safety for the evaluation of autonomous products.'," *Edge Case Research*, 2019.

[67] J.-R. Xue, J.-W. Fang, and P. Zhang, "A survey of scene understanding by event reasoning in autonomous driving," *International Journal of Automation and Computing*, vol. 15, no. 3, pp. 249–266, 2018.

[68] D. Wang, X. Hou, J. Xu, S. Yue, and C.-L. Liu, "Traffic sign detection using a cascade method with fast feature extraction and saliency test," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 12, pp. 3290–3302, 2017.

[69] T. Deng, K. Yang, Y. Li, and H. Yan, "Where does the driver look? top-down-based saliency detection in a traffic driving environment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 7, pp. 2051–2062, 2016.

[70] A. Zang, Z. Li, D. Doria, and G. Trajcevski, "Accurate vehicle self-localization in high definition map dataset," in *Proceedings of the 1st ACM SIGSPATIAL Workshop on High-Precision Maps and Intelligent Applications for Autonomous Vehicles*, p. 2, ACM, 2017.

[71] S. K. Jenson and J. O. Domingue, "Extracting topographic structure from digital elevation data for geographic information system analysis," *Photogrammetric engineering and remote sensing*, vol. 54, no. 11, pp. 1593–1600, 1988.

[72] K. Johnston, J. M. Ver Hoef, K. Krivoruchko, and N. Lucas, *Using ArcGIS geostatistical analyst*, vol. 380. Esri Redlands, 2001.

[73] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, 2008.

[74] A. Ess, T. Mueller, H. Grabner, and L. J. Van Gool, "Segmentation-based urban traffic scene understanding.," in *BMVC*, vol. 1, p. 2, Citeseer, 2009.

[75] A. Geiger, M. Lauer, and R. Urtasun, "A generative model for 3d urban scene understanding from movable platforms," in *CVPR 2011*, pp. 1945–1952, IEEE, 2011.

[76] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: A survey of current trends in autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194–220, 2017.

[77] C.-C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 1, pp. 842–849, IEEE, 2003.

[78] H. Kong, J.-Y. Audibert, and J. Ponce, "Vanishing point detection for road detection," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 96–103, IEEE, 2009.

[79] H. Kong, S. E. Sarma, and F. Tang, "Generalizing laplacian of gaussian filters for vanishing-point detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 408–418, 2012.

[80] J. Shi, J. Wang, and F. Fu, "Fast and robust vanishing point detection for unstructured road following," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 970–979, 2015.

[81] E. Casapietra, T. H. Weisswange, C. Goerick, and F. Kummert, "Enriching a spatial road representation with lanes and driving directions," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1579–1585, IEEE, 2016.

[82] D. Kasper, G. Weidl, T. Dang, G. Breuel, A. Tamke, A. Wedel, and W. Rosenstiel, "Object-oriented bayesian networks for detection of lane change maneuvers," *IEEE Intelligent Transportation Systems Magazine*, vol. 4, no. 3, pp. 19–31, 2012.

[83] J. Nilsson, J. Silvlin, M. Brannstrom, E. Coelingh, and J. Fredriksson, "If, when, and how to perform lane change maneuvers on highways," *IEEE Intelligent Transportation Systems Magazine*, vol. 8, no. 4, pp. 68–78, 2016.

[84] E. I. Vlahogianni and J. C. Golias, "Bayesian modeling of the microscopic traffic characteristics of overtaking in two-lane highways," *Transportation research part F: traffic psychology and behaviour*, vol. 15, no. 3, pp. 348–357, 2012.

[85] G. A. Davis and T. Swenson, "Collective responsibility for freeway rear-ending accidents?: An application of probabilistic causal models," *Accident Analysis & Prevention*, vol. 38, no. 4, pp. 728–736, 2006.

[86] T. Gindele, S. Brechtel, and R. Dillmann, "A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments," in *13th International IEEE Conference on Intelligent Transportation Systems*, pp. 1625–1631, IEEE, 2010.

[87] R. K. Satzoda and M. M. Trivedi, "Overtaking & receding vehicle detection for driver assistance and naturalistic driving studies," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 697–702, IEEE, 2014.

[88] A. Khosroshahi, E. Ohn-Bar, and M. M. Trivedi, "Surround vehicles trajectory analysis with recurrent neural networks," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2267–2272, IEEE, 2016.

[89] S. Busch, T. Schindler, T. Klinger, and C. Brenner, "Analysis of spatio-temporal traffic patterns based on pedestrian trajectories," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences-ISPRS Archives 41 (2016)*, vol. 41, pp. 497–503, 2016.

[90] R. M. Mueid, C. Ahmed, and M. A. R. Ahad, "Pedestrian activity classification using patterns of motion and histogram of oriented gradient," *Journal on Multimodal User Interfaces*, vol. 10, no. 4, pp. 299–305, 2016.

[91] C. G. Keller and D. M. Gavrila, "Will the pedestrian cross? a study on pedestrian path prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 494–506, 2013.

[92] R. Quintero, J. Almeida, D. F. Llorca, and M. Sotelo, "Pedestrian path prediction using body language traits," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pp. 317–323, IEEE, 2014.

[93] A. Tageldin, M. H. Zaki, and T. Sayed, "Examining pedestrian evasive actions as a potential indicator for traffic conflicts," *IET Intelligent Transport Systems*, vol. 11, no. 5, pp. 282–289, 2017.

[94] Y. Hou, P. Edara, and C. Sun, "Modeling mandatory lane changing using bayes classifier and decision trees," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 647–655, 2013.

[95] N. Schneider and D. M. Gavrila, "Pedestrian path prediction with recursive bayesian filters: A comparative study," in *German Conference on Pattern Recognition*, pp. 174–183, Springer, 2013.

[96] M. Goldhammer, M. Gerhard, S. Zernetsch, K. Doll, and U. Brunsmann, "Early prediction of a pedestrian's trajectory at intersections," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pp. 237–242, IEEE, 2013.

[97] W. Song, G. Xiong, and H. Chen, "Intention-aware autonomous driving decision-making in an uncontrolled intersection," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[98] D. J. Phillips, T. A. Wheeler, and M. J. Kochenderfer, "Generalizable intention prediction of human drivers at intersections," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1665–1670, IEEE, 2017.

[99] X.-Y. Lu and A. Skabardonis, "Freeway traffic shockwave analysis: exploring the ngsim trajectory data," in *86th Annual Meeting of the Transportation Research Board, Washington, DC*, 2007.

[100] K. Pei, Y. Cao, J. Yang, and S. Jana, "Deepxplore: Automated whitebox testing of deep learning systems," in *Proceedings of the 26th Symposium on Operating Systems Principles*, pp. 1–18, ACM, 2017.

[101] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid, "Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems," in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, pp. 132–142, ACM, 2018.

[102] X. Xie, L. Ma, F. Juefei-Xu, M. Xue, H. Chen, Y. Liu, J. Zhao, B. Li, J. Yin, and S. See, "Deephunter: a coverage-guided fuzz testing framework for deep neural networks," in *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pp. 146–157, 2019.

[103] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, pp. 1050–1059, 2016.

[104] C. Mao, X. Zhan, J. Chen, J. Chen, and R. Huang, "Adaptive random testing based on flexible partitioning," *IET Software*, vol. 14, no. 5, pp. 493–505, 2020.

[105] H. Pei, B. Yin, M. Xie, and K.-Y. Cai, "Dynamic random testing with test case clustering and distance-based parameter adjustment," *Information and Software Technology*, vol. 131, p. 106470, 2021.

[106] T. Bai, J. Luo, J. Zhao, B. Wen, and Q. Wang, "Recent advances in adversarial training for adversarial robustness," *arXiv preprint arXiv:2102.01356*, 2021.

[107] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou, *et al.*, "Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9710–9719, 2021.

[108] S. He, "Who is liable for the uber self-driving crash? analysis of the liability allocation and the regulatory model for autonomous vehicles," in *Autonomous Vehicles*, pp. 93–111, Springer, 2021.

[109] H. Yu, S. Yang, W. Gu, and S. Zhang, "Baidu driving dataset and end-to-end reactive control model," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 341–346, IEEE, 2017.

[110] P. Liu, Z. Xu, and X. Zhao, "Road tests of self-driving vehicles: Affective and cognitive pathways in acceptance formation," *Transportation research part A: policy and practice*, vol. 124, pp. 354–369, 2019.

[111] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?," *Transportation Research Part A: Policy and Practice*, vol. 94, pp. 182–193, 2016.

[112] S. S. Banerjee, S. Jha, J. Cyriac, Z. T. Kalbarczyk, and R. K. Iyer, "Hands off the wheel in autonomous vehicles?: A systems perspective on over a million miles of field data," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 586–597, IEEE, 2018.

[113] M. Schwall, T. Daniel, T. Victor, F. Favaro, and H. Hohnhold, "Waymo public road safety performance data," *arXiv preprint arXiv:2011.00038*, 2020.

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

[115] R. Guzmán, J.-B. Hayet, and R. Klette, "Towards ubiquitous autonomous driving: The ccsad dataset," in *International Conference on Computer Analysis of Images and Patterns*, pp. 582–593, Springer, 2015.

[116] G. Pandey, J. R. McBride, and R. M. Eustice, "Ford campus vision and lidar data set," *The International Journal of Robotics Research*, vol. 30, no. 13, pp. 1543–1552, 2011.

[117] A. Patil, S. Malla, H. Gang, and Y.-T. Chen, "The h3d dataset for full-surround 3d multi-object detection and tracking in crowded urban scenes," in *International Conference on Robotics and Automation*, 2019.

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

[119] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3213–3223, 2016.

[120] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving video database with scalable annotation tooling," *arXiv preprint arXiv:1805.04687*, 2018.

[121] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: An open dataset benchmark," 2019.

[122] P. Wang, X. Huang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The apolloscape open dataset for autonomous driving and its application," *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[123] V. Ramanishka, Y.-T. Chen, T. Misu, and K. Saenko, "Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[124] A. Narayanan, I. Dwivedi, and B. Dariush, "Dynamic traffic scene classification with space-time coherence," *arXiv preprint arXiv:1905.12708*, 2019.

[125] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88–97, 2009.

[126] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 4, pp. 743–761, 2011.

[127] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The oxford robotcar dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.

[128] J. Jeong, Y. Cho, Y.-S. Shin, H. Roh, and A. Kim, "Complex urban dataset with multi-level sensors from highly diverse urban environments," *The International Journal of Robotics Research*, vol. 38, no. 6, pp. 642–657, 2019.

[129] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet, "Lyft level 5 av dataset 2019." urlhttps://level5.lyft.com/dataset/, 2019.

[130] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, 2017.

[131] F. Codevilla, M. Miiller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–9, IEEE, 2018.

[132] Y. Li, K. Li, Y. Zheng, B. Morys, S. Pan, and J. Wang, "Threat assessment techniques in intelligent vehicles: A comparative survey," *IEEE Intelligent Transportation Systems Magazine*, vol. 13, no. 4, pp. 71–91, 2020.

[133] S. Brunson, E. Kyle, N. Phamdo, and G. Preziotti, "Alert algorithm development program: Nhtsa rear-end collision alert algorithm," tech. rep., 2002.

[134] T. Hiraoka, M. Tanaka, H. Kumamoto, T. Izumi, and K. Hatanaka, "Collision risk evaluation index based on deceleration for collision avoidance (first report)-proposal of a new index to evaluate collision risk against forward obstacles," *Review of Automotive Engineering*, vol. 30, no. 4, pp. 429–437, 2009.

[135] Q. Zhu, Y. Yan, and Z. Xing, "Robot path planning based on artificial potential field approach with simulated annealing," in *Sixth International Conference on Intelligent Systems Design and Applications*, vol. 2, pp. 622–627, IEEE, 2006.

[136] L. Li, J. Gan, X. Ji, X. Qu, and B. Ran, "Dynamic driving risk potential field model under the connected and automated vehicles environment and its application in car-following modeling," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[137] H. Wang, Y. Huang, A. Khajepour, Y. Zhang, Y. Rasekhipour, and D. Cao, "Crash mitigation in motion planning for autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3313–3323, 2019.

[138] S. Kolekar, J. de Winter, and D. Abbink, "Human-like driving behaviour emerges from a risk-based driver model," *Nature communications*, vol. 11, no. 1, pp. 1–13, 2020.

[139] W. V. WINSUM and A. Heino, "Choice of time-headway in car-following and the role of time-to-collision information in braking," *Ergonomics*, vol. 39, no. 4, pp. 579–592, 1996.

[140] M. Makridis, K. Mattas, and B. Ciuffo, "Response time and time headway of an adaptive cruise control. an empirical characterization and potential impacts on road capacity," *IEEE transactions on intelligent transportation systems*, vol. 21, no. 4, pp. 1677–1686, 2019.

[141] J. Chen, Y. Zhou, and H. Liang, "Effects of acc and cacc vehicles on traffic flow based on an improved variable time headway spacing strategy," *IET Intelligent Transport Systems*, vol. 13, no. 9, pp. 1365–1373, 2019.

[142] A. Manglik, X. Weng, E. Ohn-Bar, and K. M. Kitanil, "Forecasting time-to-collision from monocular video: Feasibility, dataset, and challenges," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8081–8088, IEEE, 2019.

[143] R. Chen, R. Sherony, and H. C. Gabler, "Comparison of time to collision and enhanced time to collision at brake application during normal driving," tech. rep., SAE Technical Paper, 2016.

[144] K. D. Kusano, R. Chen, J. Montgomery, and H. C. Gabler, "Population distributions of time to collision at brake application during car following from naturalistic driving data," *Journal of safety research*, vol. 54, pp. 95–e29, 2015.

[145] J. Hillenbrand, A. M. Spieker, and K. Kroschel, "A multilevel collision mitigation approach—its situation assessment, decision making, and performance tradeoffs," *IEEE Transactions on intelligent transportation systems*, vol. 7, no. 4, pp. 528–540, 2006.

[146] Y. Zhang, E. K. Antonsson, and K. Grote, "A new threat assessment measure for collision avoidance systems," in *2006 IEEE Intelligent Transportation Systems Conference*, pp. 968–975, IEEE, 2006.

[147] J. E. Stellet, P. Vogt, J. Schumacher, W. Branz, and J. M. Zöllner, "Analytical derivation of performance bounds of autonomous emergency brake systems," in *2016 ieee intelligent vehicles symposium (iv)*, pp. 220–226, IEEE, 2016.

[148] S. Liu, X. Wang, O. Hassanin, X. Xu, M. Yang, D. Hurwitz, and X. Wu, "Calibration and evaluation of responsibility-sensitive safety (rss) in automated vehicle performance during cut-in scenarios," *Transportation research part C: emerging technologies*, vol. 125, p. 103037, 2021.

[149] E. J. Rossetter and J. C. Gerdes, "Lyapunov based performance guarantees for the potential field lane-keeping assistance system," 2006.

[150] J. Wang, Y. Zheng, X. Li, C. Yu, K. Kodaka, and K. Li, "Driving risk assessment using near-crash database through data mining of tree-based model," *Accident Analysis & Prevention*, vol. 84, pp. 54–64, 2015.

[151] Y. Huang, H. Ding, Y. Zhang, H. Wang, D. Cao, N. Xu, and C. Hu, "A motion planning and tracking framework for autonomous vehicles based on artificial potential field elaborated resistance network approach," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 2, pp. 1376–1386, 2019.

[152] J. Wang, J. Wu, X. Zheng, D. Ni, and K. Li, "Driving safety field theory modeling and its application in pre-collision warning system," *Transportation research part C: emerging technologies*, vol. 72, pp. 306–324, 2016.

[153] M. Althoff and A. Mergel, "Comparison of markov chain abstraction and monte carlo simulation for the safety assessment of autonomous cars," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1237–1247, 2011.

[154] X. Xiong, L. Chen, and J. Liang, "Vehicle driving risk prediction based on markov chain model," *Discrete Dynamics in Nature and Society*, vol. 2018, 2018.

[155] A. Lambert, D. Gruyer, and G. Saint Pierre, "A fast monte carlo algorithm for collision probability estimation," in *2008 10th International Conference on Control, Automation, Robotics and Vision*, pp. 406–411, IEEE, 2008.

[156] A. Murphy and J. Xia, "Risk analysis of animal–vehicle crashes: a hierarchical bayesian approach to spatial modelling," *International Journal of Crashworthiness*, vol. 21, no. 6, pp. 614–626, 2016.

[157] S. Joerer, M. Segata, B. Bloessl, R. L. Cigno, C. Sommer, and F. Dressler, "A vehicular networking perspective on estimating vehicle collision probability at intersections," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 4, pp. 1802–1812, 2013.

[158] D. Lee and H. Yeo, "Real-time rear-end collision-warning system using a multilayer perceptron neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3087–3097, 2016.

[159] D. S. Brown, Y. Cui, and S. Niekum, "Risk-aware active inverse reinforcement learning," in *Conference on Robot Learning*, pp. 362–372, PMLR, 2018.

[160] S. Annell, A. Gratner, and L. Svensson, "Probabilistic collision estimation system for autonomous vehicles," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 473–478, IEEE, 2016.

[161] K. Czarnecki and R. Salay, "Towards a framework to manage perceptual uncertainty for safe automated driving," in *International Conference on Computer Safety, Reliability, and Security*, pp. 439–445, Springer, 2018.

[162] Y. Dong, Z. Hu, K. Uchimura, and N. Murayama, "Driver inattention monitoring system for intelligent vehicles: A review," *IEEE transactions on intelligent transportation systems*, vol. 12, no. 2, pp. 596–614, 2010.

[163] P. Koopman, U. Ferrell, F. Fratrik, and M. Wagner, "A safety standard approach for fully autonomous vehicles," in *International Conference on Computer Safety, Reliability, and Security*, pp. 326–332, Springer, 2019.

[164] P. Cimiano, *Ontologies*. Springer, 2006.

[165] X. Geng, H. Liang, B. Yu, P. Zhao, L. He, and R. Huang, "A scenario-adaptive driving behavior prediction approach to urban autonomous driving," *Applied Sciences*, vol. 7, no. 4, p. 426, 2017.

[166] H. Herre, "General formal ontology (gfo): A foundational ontology for conceptual modelling," in *Theory and applications of ontology: computer applications*, pp. 297–345, Springer, 2010.

[167] K. Czarnecki, "Operational design domain for automated driving systems–taxonomy of basic terms," *Waterloo Intelligent Systems Engineering (WISE) Lab, University of Waterloo, Canada*, 2018.

[168] C. Sun, J. M. U. Vianney, Y. Li, L. Chen, L. Li, F.-Y. Wang, A. Khajepour, and D. Cao, "Proximity based automatic data annotation for autonomous driving," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 395–404, 2020.

[169] H. Winner, K. Lemmer, T. Form, and J. Mazzega, "Pegasus—first steps for the safe introduction of automated driving," in *Road Vehicle Automation 5*, pp. 185–195, Springer, 2019.

[170] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7167–7176, 2017.

[171] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[172] F. Yu, Z. Xu, Y. Wang, C. Liu, and X. Chen, "Towards robust training of neural networks by regularizing adversarial gradients," *arXiv preprint arXiv:1805.09370*, 2018.

[173] H. Wang, C. Xiao, J. Kossaifi, Z. Yu, A. Anandkumar, and Z. Wang, "Augmax: Adversarial composition of random augmentations for robust training," *Advances in neural information processing systems*, vol. 34, pp. 237–250, 2021.

[174] F. Tramèr and D. Boneh, "Adversarial training and robustness for multiple perturbations," *arXiv preprint arXiv:1904.13000*, 2019.

[175] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *International conference on machine learning*, pp. 7472–7482, PMLR, 2019.

[176] H. Wang, X. Shi, and D.-Y. Yeung, "Natural-parameter networks: A class of probabilistic neural networks," *Advances in Neural Information Processing Systems*, vol. 29, pp. 118–126, 2016.

[177] R. Taori, A. Dave, V. Shankar, N. Carlini, B. Recht, and L. Schmidt, "Measuring robustness to natural distribution shifts in image classification," *Advances in Neural Information Processing Systems*, vol. 33, pp. 18583–18599, 2020.

[178] A. Robey, H. Hassani, and G. J. Pappas, "Model-based robust deep learning: Generalizing to natural, out-of-distribution data," *arXiv preprint arXiv:2005.10247*, 2020.

[179] M. Gyllenhammar, R. Johansson, F. Warg, D. Chen, H.-M. Heyn, M. Sanfridson, J. Söderberg, A. Thorsén, D. Chen, and S. Ursing, "Towards an operational design domain that supports the safety argumentation of an automated driving system," in *10th European Congress on Embedded Real Time Systems*, (Toulouse, France), 2020.

[180] I. ISO, "26262-1: 2011 road vehicles–functional safety–part 1: Vocabulary," *Berlin: ISO*, 2011.

[181] J. Dunjó, V. Fthenakis, J. A. Vílchez, and J. Arnaldos, "Hazard and operability (hazop) analysis. a literature review," *Journal of hazardous materials*, vol. 173, no. 1-3, pp. 19–32, 2010.

[182] R. Abdallah, R. Kouta, C. Sarraf, J. Gaber, and M. Wack, "Fault tree analysis for the communication of a fleet formation flight of uavs," in *2017 2nd International Conference on System Reliability and Safety (ICSRS)*, pp. 202–206, IEEE, 2017.

[183] F. Yan, T. Tang, and H. Yan, "Scenario based stpa analysis in automated urban guided transport system," in *2016 IEEE International Conference on Intelligent Rail Transportation (ICIRT)*, pp. 425–431, IEEE, 2016.

[184] Y. Ali, M. M. Haque, Z. Zheng, S. Washington, and M. Yildirimoglu, "A hazard-based duration model to quantify the impact of connected driving environment on safety during mandatory lane-changing," *Transportation research part C: emerging technologies*, vol. 106, pp. 113–131, 2019.

[185] P. Koopman and M. Wagner, "Autonomous vehicle safety: An interdisciplinary challenge," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 1, pp. 90–96, 2017.

[186] A. Schnellbach and G. Griessnig, "Development of the iso 21448," in *European Conference on Software Process Improvement*, pp. 585–593, Springer, 2019.

[187] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2446–2454, 2020.

[188] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11621–11631, 2020.

[189] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," *arXiv preprint arXiv:1711.03938*, 2017.

[190] D. J. Fremont, T. Dreossi, S. Ghosh, X. Yue, A. L. Sangiovanni-Vincentelli, and S. A. Seshia, "Scenic: a language for scenario specification and scene generation,"

in *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 63–78, 2019.

[191] C. Sun, L. Su, S. Gu, J. M. U. Vianney, K. Qin, and D. Cao, "Cross validation for cnn based affordance learning and control for autonomous driving," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 1519–1524, IEEE, 2019.

[192] D. Araiza-Illan, D. Western, A. Pipe, and K. Eder, "Coverage-driven verification—," in *Haifa Verification Conference*, pp. 69–84, Springer, 2015.

[193] G. Bagschik, T. Menzel, and M. Maurer, "Ontology based scene creation for the development of automated vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1813–1820, IEEE, 2018.

[194] H. Weber, J. Bock, J. Klimke, C. Roesener, J. Hiller, R. Krajewski, A. Zlocki, and L. Eckstein, "A framework for definition of logical scenarios for safety assurance of automated driving," *Traffic injury prevention*, vol. 20, no. sup1, pp. S65–S70, 2019.

[195] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[196] M. Althoff, S. Urban, and M. Koschi, "Automatic conversion of road networks from opendrive to lanelets," in *2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, pp. 157–162, IEEE, 2018.

[197] F. Poggenhans, J.-H. Pauls, J. Janosovits, S. Orf, M. Naumann, F. Kuhnt, and M. Mayr, "Lanelet2: A high-definition map framework for the future of automated driving," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1672–1679, IEEE, 2018.

[198] M. Szalai, B. Varga, T. Tettamanti, and V. Tihanyi, "Mixed reality test environment for autonomous cars using unity 3d and sumo," in *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pp. 73–78, IEEE, 2020.

[199] L. Bai, P. Liu, C.-Y. Chan, and Z. Li, "Estimating level of service of mid-block bicycle lanes considering mixed traffic flow," *Transportation research part A: policy and practice*, vol. 101, pp. 203–217, 2017.

[200] G. Volk, S. Müller, A. von Bernuth, D. Hospach, and O. Bringmann, "Towards robust cnn-based object detection through augmentation with synthetic rain variations," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 285–292, IEEE, 2019.

[201] L. Milor and A. L. Sangiovanni-Vincentelli, "Minimizing production test time to detect faults in analog circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 13, no. 6, pp. 796–813, 1994.

[202] A. Segismundo and P. A. C. Miguel, "Failure mode and effects analysis (fmea) in the context of risk management in new product development: A case study in an automotive company," *International Journal of Quality & Reliability Management*, 2008.

[203] R. Mader, E. Armengaud, G. Grießnig, C. Kreiner, C. Steger, and R. Weiß, "Oasis: An automotive analysis and safety engineering instrument," *Reliability engineering & system safety*, vol. 120, pp. 150–162, 2013.

[204] A. P. Kaleeswaran, P. Munk, S. Sarkic, T. Vogel, and A. Nordmann, "A domain specific language to support hazop studies of sysml models," in *International Symposium on Model-Based Safety and Assessment*, pp. 47–62, Springer, 2019.

[205] A. Abdulkhaleq, S. Wagner, D. Lammering, H. Boehmert, and P. Blueher, "Using stpa in compliance with iso 26262 for developing a safe architecture for fully automated vehicles," *arXiv preprint arXiv:1703.03657*, 2017.

[206] B. Li, S. Shang, and Y. Fu, "The application of stpa in the development of autonomous vehicle functional safety," in *2021 International Conference on Intelligent Computing, Automation and Applications (ICAA)*, pp. 863–868, IEEE, 2021.

[207] Y. Xiao, F. Codevilla, A. Gurram, O. Urfalioglu, and A. M. López, "Multimodal end-to-end autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[208] M. Luckcuck, M. Farrell, L. A. Dennis, C. Dixon, and M. Fisher, "Formal specification and verification of autonomous robotic systems: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 5, pp. 1–41, 2019.

[209] C. Hu, W. Dong, Y. Yang, H. Shi, and G. Zhou, "Runtime verification on hierarchical properties of ros-based robot swarms," *IEEE Transactions on Reliability*, 2019.

[210] T. Dreossi, A. Donzé, and S. A. Seshia, "Compositional falsification of cyber-physical systems with machine learning components," *Journal of Automated Reasoning*, vol. 63, no. 4, pp. 1031–1053, 2019.

[211] F. Codevilla, A. M. Lopez, V. Koltun, and A. Dosovitskiy, "On offline evaluation of vision-based driving models," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 236–251, 2018.

[212] S. Mitsch, K. Ghorbal, D. Vogelbacher, and A. Platzer, "Formal verification of obstacle avoidance and navigation of ground robots," *The International Journal of Robotics Research*, vol. 36, no. 12, pp. 1312–1340, 2017.

[213] N. Arechiga, "Specifying safety of autonomous vehicles in signal temporal logic," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 58–63, IEEE, 2019.

[214] M. Hekmatnejad, S. Yaghoubi, A. Dokhanchi, H. B. Amor, A. Shrivastava, L. Karam, and G. Fainekos, "Encoding and monitoring responsibility sensitive safety rules for automated vehicles in signal temporal logic," in *Proceedings of the 17th ACM-IEEE International Conference on Formal Methods and Models for System Design*, pp. 1–11, 2019.

[215] A. Sauer, N. Savinov, and A. Geiger, "Conditional affordance learning for driving in urban environments," in *Conference on Robot Learning (CoRL)*, 2018.

[216] T. I. Cannings, Y. Fan, and R. J. Samworth, "Classification with imperfect training labels," *Biometrika*, vol. 107, no. 2, pp. 311–330, 2020.

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

[218] C. Yu, J. Wang, Y. Chen, and M. Huang, "Transfer learning with dynamic adversarial adaptation network," in *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 778–786, IEEE, 2019.

[219] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[220] M. Zinkevich, M. Weimer, L. Li, and A. Smola, "Parallelized stochastic gradient descent," *Advances in neural information processing systems*, vol. 23, 2010.

[221] Z. Zhang, "Improved adam optimizer for deep neural networks," in *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, pp. 1–2, IEEE, 2018.

[222] A. Najafi, S.-i. Maeda, M. Koyama, and T. Miyato, "Robustness to adversarial perturbations in learning from incomplete data," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[223] A. Fawzi, O. Fawzi, and P. Frossard, "Analysis of classifiers' robustness to adversarial perturbations," *Machine learning*, vol. 107, no. 3, pp. 481–508, 2018.

[224] S. S. Halder, J.-F. Lalonde, and R. d. Charette, "Physics-based rendering for improving robustness to rain," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10203–10212, 2019.

[225] W. Qiu and A. Yuille, "Unrealcv: Connecting computer vision to unreal engine," in *European Conference on Computer Vision*, pp. 909–916, Springer, 2016.

[226] J. Lin, Y. Xia, T. Qin, Z. Chen, and T.-Y. Liu, "Conditional image-to-image translation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5524–5532, 2018.

[227] C. Sakaridis, D. Dai, and L. Van Gool, "Semantic foggy scene understanding with synthetic data," *International Journal of Computer Vision*, vol. 126, no. 9, pp. 973–992, 2018.

[228] M. Tremblay, S. S. Halder, R. de Charette, and J.-F. Lalonde, "Rain rendering for evaluating and improving robustness to bad weather," *International Journal of Computer Vision*, 2020.

[229] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

[230] A. B. Jung, K. Wada, J. Crall, S. Tanaka, J. Graving, C. Reinders, S. Yadav, J. Banerjee, G. Vecsei, A. Kraft, Z. Rui, J. Borovec, C. Vallentin, S. Zhydenko, K. Pfeiffer, B. Cook, I. Fernández, F.-M. De Rainville, C.-H. Weng, A. Ayala-Acevedo, R. Meudec, M. Laporte, *et al.*, "imgaug." https://github.com/aleju/imgaug, 2020. Online; accessed 01-Feb-2020.

[231] A. S. Mohammed, A. Amamou, F. K. Ayevide, S. Kelouwani, K. Agbossou, and N. Zioui, "The perception system of intelligent ground vehicles in all weather conditions: a systematic literature review," *Sensors*, vol. 20, no. 22, p. 6532, 2020.

[232] S. Hong, M. Kim, and M. G. Kang, "Single image dehazing via atmospheric scattering model-based image fusion," *Signal Processing*, vol. 178, p. 107798, 2021.

[233] M. Hansard, "Fast synthesis of atmospheric image effects," in *European Conference on Visual Media Production*, pp. 1–10, 2019.

[234] P. F. Proença and Y. Gao, "Deep learning for spacecraft pose estimation from photorealistic rendering," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6007–6013, IEEE, 2020.

[235] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 12, pp. 2341–2353, 2010.

[236] F. Guo, J. Tang, and X. Xiao, "Foggy scene rendering based on transmission map estimation," *International Journal of Computer Games Technology*, vol. 2014, 2014.

[237] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 1253–1260, IEEE, 2010.

[238] C. Sakaridis, D. Dai, S. Hecker, and L. Van Gool, "Model adaptation with synthetic and real data for semantic dense foggy scene understanding," in *Proceedings of the european conference on computer vision (ECCV)*, pp. 687–704, 2018.

[239] R. De Charette, R. Tamburo, P. C. Barnum, A. Rowe, T. Kanade, and S. G. Narasimhan, "Fast reactive control for illumination through rain and snow," in *2012 IEEE International Conference on Computational Photography (ICCP)*, pp. 1–10, IEEE, 2012.

[240] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[241] Y. Alharbi, N. Smith, and P. Wonka, "Latent filter scaling for multimodal unsupervised image-to-image translation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1458–1466, 2019.

[242] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

[243] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, E. Boise, G. Uhm, M. Gerow, S. Mehta, *et al.*, "Lgsvl simulator: A high fidelity simulator for autonomous driving," in *2020 IEEE 23rd International conference on intelligent transportation systems (ITSC)*, pp. 1–6, IEEE, 2020.

[244] P. Kaur, S. Taghavi, Z. Tian, and W. Shi, "A survey on simulators for testing self-driving cars," in *2021 Fourth International Conference on Connected and Autonomous Driving (MetroCAD)*, pp. 62–70, IEEE, 2021.

[245] R. M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik, "Sgd: General analysis and improved rates," in *International Conference on Machine Learning*, pp. 5200–5209, PMLR, 2019.

[246] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," *arXiv preprint arXiv:1712.04621*, 2017.

[247] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[248] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "AugMix: A simple data processing method to improve robustness and uncertainty," *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[249] D. Temel, M.-H. Chen, and G. AlRegib, "Traffic sign detection under challenging conditions: A deeper look into performance variations and spectral characteristics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3663–3673, 2019.

[250] R. Padilla, S. L. Netto, and E. A. Da Silva, "A survey on performance metrics for object-detection algorithms," in *2020 international conference on systems, signals and image processing (IWSSIP)*, pp. 237–242, IEEE, 2020.

[251] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[252] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry, "Exploring the landscape of spatial robustness," in *International conference on machine learning*, pp. 1802–1811, PMLR, 2019.

[253] R. U. Khan, X. Zhang, and R. Kumar, "Analysis of resnet and googlenet models for malware detection," *Journal of Computer Virology and Hacking Techniques*, vol. 15, no. 1, pp. 29–37, 2019.

[254] L. Bottou, "Stochastic gradient descent tricks," in *Neural networks: Tricks of the trade*, pp. 421–436, Springer, 2012.

[255] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30, IEEE, 2017.

[256] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 969–977, 2018.

[257] M. Hörwick and K.-H. Siedersberger, "Strategy and architecture of a safety concept for fully automatic and autonomous driving assistance systems," in *2010 IEEE Intelligent Vehicles Symposium*, pp. 955–960, IEEE, 2010.

[258] I. Colwell, B. Phan, S. Saleem, R. Salay, and K. Czarnecki, "An automated vehicle safety concept based on runtime restriction of the operational design domain," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1910–1917, IEEE, 2018.

[259] A. Reschka, G. Bagschik, S. Ulbrich, M. Nolte, and M. Maurer, "Ability and skill graphs for system modeling, online monitoring, and decision support for vehicle guidance systems," in *2015 Ieee intelligent vehicles symposium (Iv)*, pp. 933–939, IEEE, 2015.

[260] C. W. Lee, N. Nayeer, D. E. Garcia, A. Agrawal, and B. Liu, "Identifying the operational design domain for an automated driving system through assessed risk," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1317–1322, IEEE, 2020.

[261] C. Sun, J. M. U. Vianney, Y. Li, L. Chen, L. Li, F. Wang, A. Khajepour, and D. Cao, "Proximity based automatic data annotation for autonomous driving," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 395–404, 2020.

[262] S. H. Chen and C. A. Pollino, "Good practice in bayesian network modelling," *Environmental Modelling & Software*, vol. 37, pp. 134–145, 2012.

[263] N. Gosala, A. Bühler, M. Prajapat, C. Ehmke, M. Gupta, R. Sivanesan, A. Gawel, M. Pfeiffer, M. Bürki, I. Sa, *et al.*, "Redundant perception and state estimation for reliable autonomous racing," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6561–6567, IEEE, 2019.

[264] S. S. Keerthi and C.-J. Lin, "Asymptotic behaviors of support vector machines with gaussian kernel," *Neural computation*, vol. 15, no. 7, pp. 1667–1689, 2003.

[265] B. Liu, V. P. Betancourt, Y. Zhu, and J. Becker, "Towards an on-demand redundancy concept for autonomous vehicle functions using microservice architecture," in *2020 IEEE International Symposium on Systems Engineering (ISSE)*, pp. 1–5, IEEE, 2020.

[266] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[267] J. Wang, T. Mei, B. Kong, and H. Wei, "An approach of lane detection based on inverse perspective mapping," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 35–38, IEEE, 2014.

[268] J. Tang, S. Li, and P. Liu, "A review of lane detection methods based on deep learning," *Pattern Recognition*, vol. 111, p. 107623, 2021.

[269] L. Tabelini, R. Berriel, T. M. Paixao, C. Badue, A. F. De Souza, and T. Oliveira-Santos, "Polylanenet: Lane estimation via deep polynomial regression," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 6150–6156, IEEE, 2021.

[270] D. Neven, B. De Brabandere, S. Georgoulis, M. Proesmans, and L. Van Gool, "Towards end-to-end lane detection: an instance segmentation approach," in *2018 IEEE intelligent vehicles symposium (IV)*, pp. 286–291, IEEE, 2018.

[271] C. R. Jung and C. R. Kelber, "A robust linear-parabolic model for lane following," in *Proceedings. 17th Brazilian Symposium on Computer Graphics and Image Processing*, pp. 72–79, IEEE, 2004.

[272] J.-H. Pauls, B. Schmidt, and C. Stiller, "Automatic mapping of tailored landmark representations for automated driving and map learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6725–6731, IEEE, 2021.

[273] K. Choi, J. K. Suhr, and H. G. Jung, "Map-matching-based cascade landmark detection and vehicle localization," *IEEE Access*, vol. 7, pp. 127874–127894, 2019.

[274] M. Rapp, K. Dietmayer, M. Hahn, F. Schuster, J. Lombacher, and J. Dickmann, "Fscd and basd: Robust landmark detection and description on radar-based grids," in *2016 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, pp. 1–4, IEEE, 2016.

[275] D. Wilbers, C. Merfels, and C. Stachniss, "Localization with sliding window factor graphs on third-party maps for automated driving," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5951–5957, IEEE, 2019.

[276] G. Ducamp, C. Gonzales, and P.-H. Wuillemin, "aGrUM/pyAgrum : a Toolbox to Build Models and Algorithms for Probabilistic Graphical Models in Python," in *10th International Conference on Probabilistic Graphical Models*, vol. 138 of *Proceedings of Machine Learning Research*, (Skørping, Denmark), pp. 609–612, Sept. 2020.

[277] A. Fabris, L. Parolini, S. Schneider, and A. Cenedese, "Use of probabilistic graphical methods for online map validation," in *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*, pp. 43–48, IEEE, 2021.

[278] T. G. Reid, S. E. Houts, R. Cammarata, G. Mills, S. Agarwal, A. Vora, and G. Pandey, "Localization requirements for autonomous vehicles," *arXiv preprint arXiv:1906.01061*, 2019.

[279] Y. Yu, H. Zhao, F. Davoine, J. Cui, and H. Zha, "Monocular visual localization using road structural features," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pp. 693–699, IEEE, 2014.

[280] C. Guo, K. Kidono, J. Meguro, Y. Kojima, M. Ogawa, and T. Naito, "A low-cost solution for automatic lane-level map generation using conventional in-car sensors,"

IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 8, pp. 2355–2366, 2016.

[281] C. Sun, Z. Deng, W. Chu, S. Li, and D. Cao, "Acclimatizing the operational design domain for autonomous driving systems," *IEEE Intelligent Transportation Systems Magazine*, vol. 14, no. 2, pp. 10–24, 2021.

[282] Y. Ma, C. Sun, J. Chen, D. Cao, and L. Xiong, "Verification and validation methods for decision-making and planning of automated vehicles: A review," *IEEE Transactions on Intelligent Vehicles*, pp. 1–20, 2022.

[283] ISO, "Intelligent transport systems–lane departure warning systems–performance requirements and test procedures," 2017.

[284] R. Bell, "Introduction to iec 61508," in *Acm international conference proceeding series*, vol. 162, pp. 3–12, Citeseer, 2006.

[285] X. Xu, J. Zhao, Y. Li, H. Gao, and X. Wang, "Banet: A balanced atrous net improved from ssd for autonomous driving in smart transportation," *IEEE Sensors Journal*, vol. 21, no. 22, pp. 25018–25026, 2020.

[286] W. Shi, M. B. Alawieh, X. Li, and H. Yu, "Algorithm and hardware implementation for visual perception system in autonomous vehicle: A survey," *Integration*, vol. 59, pp. 148–156, 2017.

[287] D. Gruyer, V. Magnier, K. Hamdi, L. Claussmann, O. Orfila, and A. Rakotonirainy, "Perception, information processing and modeling: Critical stages for autonomous driving applications," *Annual Reviews in Control*, vol. 44, pp. 323–341, 2017.

[288] Z. Qin, H. Wang, and X. Li, "Ultra fast structure-aware deep lane detection," in *European Conference on Computer Vision*, pp. 276–291, Springer, 2020.

[289] Z. Qin, P. Zhang, and X. Li, "Ultra fast deep lane detection with hybrid anchor driven ordinal classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[290] P. Xingang, S. Jianping, L. Ping, W. Xiaogang, and T. Xiaoou, "Spatial as deep: Spatial cnn for traffic scene understanding," in *AAAI Conference on Artificial Intelligence (AAAI)*, February 2018.

[291] X. Zhang, J. Chen, Q. Wang, W. Xiong, X. Chen, and H. Yang, "Estimation of extrinsic parameters with trifocal tensor for intelligent vehicle-mounted cameras," *IEEE/ASME Transactions on Mechatronics*, 2022.

APPENDICES

Appendix A

A.1 Automotive standards on Lane Departure Warning

The ISO standard (ISO 17361:2007) on lane departure warning [11] legitimately addressed that the earliest and latest warning line is 0.75 m and 0.3 m inside and outside the lane boundary for passenger vehicles, as shown in Fig. A.1. The ISO standard suggests the LDW system should only operate when a vehicle travels at a speed greater than 72 km/h with a radius greater than 500 m. With a lower speed of 61 km/h, the radius of the road should then be greater than 250 m.

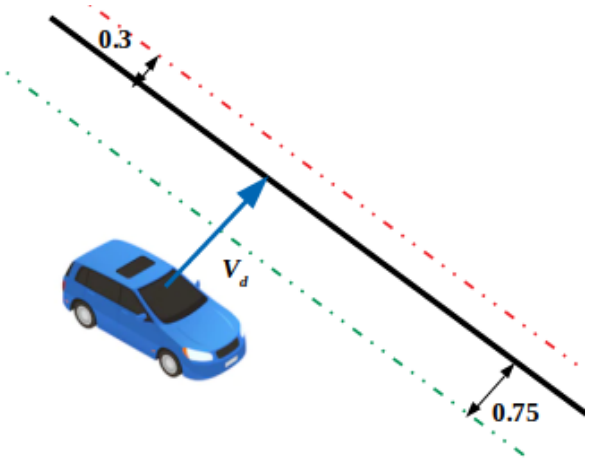


Figure A.1: The illustration of warning thresholds for lane departure referenced from [11]. The rate of lane departure V_d should smaller than 0.5 m/s.

ISO 17361:2017 [283] also states that the earliest warning threshold depends on the rate of departure V_d of the vehicle. If the rate of lane departure is greater than 0.5 m/s, then the earliest warning threshold should be calculated based on $1.5 \times V_d \times t_c$, where the t_c is the estimated time until the vehicle crosses the lane boundary.

A.2 Localization Requirements for Autonomous Vehicles

The high-level autonomous vehicle localization module aims to keep the vehicle in its respective lane during operation. The requirement relates to the vehicle's physical dimensions and road geometry.

In [278], the authors explore the horizontal, vertical, and update frequency requirements for autonomous vehicles. In our study, we only focus on the horizontal requirement, which leads to the lateral, longitudinal, and heading requirements illustrated in Fig. A.2.

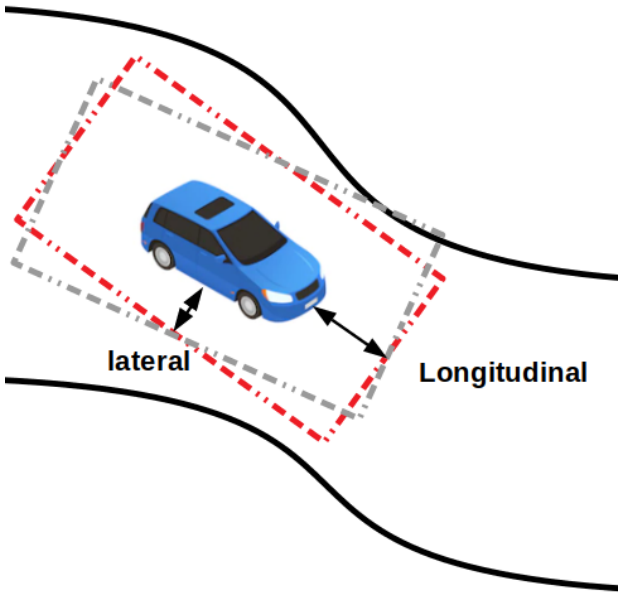


Figure A.2: The illustration of the bounding box required for localization by lateral and longitudinal and heading angle components.

Based on the analysis given in [278], the design equation for a vehicle stay within the desired lane on the 2-D plane are follows

$$\epsilon_{lat} + (\epsilon_{lon} + \tfrac{l_v}{2})\epsilon_\psi \leq \text{Lateral Alert Limit} \tag{A.1}$$

$$\epsilon_{lon} + (\epsilon_{lat} + \tfrac{w_v}{2})\epsilon_\psi \leq \text{Longitudinal Alert Limit} \tag{A.2}$$

where l_v, w_v correspond to the vehicle length and width. Based on the warning thresholds mentioned in Sec. A.1, given a 5.2 m long passenger vehicle, and the lateral alert limit set at 0.5 m for the freeway operation, then the acceptable orientation must be less than 0.15 radians (8.6°) since the $\frac{l_v}{2}\epsilon_\psi$ term alone will quickly contribute to 0.4 m. According to [278], a more reasonable choice of the heading error is around 2 degrees, leading to a contribution of 0.2 meters when increasing the l_v. Similarly, the lateral accuracy would require 0.3 meters for highway operation and 0.2 meters for local street operation. The longitudinal requirement is more forgiving than the lateral requirement, with a 0.4 meters range in highway operation and 0.2 in the urban road.

A.3 Perception Requirements for Autonomous Vehicles

In the analysis of the perception system requirement, we will examine the acceptable integrity risk of 10^{-7} failures per hour of operation corresponding to the goal standard from ISO 26262 [1] which 100 failures in one billion hours of operation. This probability level aligns with the ASIL-D, the highest automobile system standard. Other lower level use 10^{-6} and 10^{-5} probability of failure per hour in ASIL-B/C and ASIL-A in ISO 26262, which is equivalent to the standards in SIL-2 and SIL-1 in IEC-61508 [284]. Notice that the requirement mentioned above is an analysis of the perception system, which may include LiDAR, camera, radar, and most interested objects filtering. Given a perception system with an update frequency of 20 Hz, the total number of frames for an hour of operation is 72000. The number of prediction that the perception system make is more than 10^6 since there is more than one interested object in each frame. However, the 10^{-7} failure/hour failure rate does not implicate we require a neural network model with classification or bounding box detection accuracy over 99.99999% since the failure detection defined in terms of neural network training is different from the requirement of perception system as a whole. For example, the perception system of an autonomous vehicle driving on an urban road may detect many pedestrians on the pavement and vehicles surrounding the ego vehicle; only those semantically related vehicles detections. Even if there are miss detections or bounding box shifts for pedestrians on the pavement with no potential to interact with ego-vehicle, the perception system should not be considered a failure case.

There is no literature have addressed the formal requirement for perception system in perception systems in terms of the machine learning sub-components yet. We need to

Table A.1: Example performance requirement details for perception system

Perception Requirement	Description
R1. Pedestrian Detection Requirement	The system shall detect pedestrians within 20 meters range
R2. Vehicle Detection Requirement	The system shall detect vehicles within 50 meters range
R3. State Estimation Requirement	The system shall estimate the positions and velocities of objects over 80% accuracy
R4. Update Frequency Requirement	The system shall acquire raw data from all sensors at 5 Hz or higher

acknowledge that even the state-of-the-art Single Shot Detector (SSD) algorithm for object detection can optimally achieve 70-80% average precision in experiments in normal weather conditions [285]. Luckily, the accuracy requirement for the machine component in the perception system is much loose than the overall failure rate requirement. In many studies [286, 287], the perception requirements are in the table form based on an experience similar to Table A.1. The R1 - R3 corresponds to the machine learning component prediction results, which are usually decoupled in detection and classification, whereas the R4 corresponds to the updating frequency. Even though R1-R3 can be properly examined in post-analysis with additional annotation efforts, however, it is quite challenging to evaluate in real-time operation, also not directly related to the perception performance, e.g. accuracy, recall. Considering the continued development of perception modules, we select the current best performance under the non-perturbed dataset as the regular performance metric. For real-time monitoring purposes, the requirement for the perception system is that it should predict the detection and classification results with no significant performance drop. In our experiment, we find that the average classification accuracy over ... and lane estimation error suits the designed monitoring purpose.

A.4 Training Details for Vision Tasks

In this appendix, the details concerning the implementation of the robust learning algorithms are provided. All the experiments described in Chapter 4 were run on NVIDIA RTX 3060 GPU. The baseline classifiers are trained with the gradient descent method with a step size of 0.01. The number of iterations of gradient ascent per batch used during training is 20. The learning rate is set to linearly decay from 0.05 to 0.001 over 50 epochs. The batch size is set to 32 as it requires the GPU to store multiple copies of the image during the iteration time.

A.5 Lane Estimation Software Setup

The lane estimation software (as a redundancy model) is composed of three parts, including the CNN lane inference module, and two post-processing modules that map the points to the vehicle or camera frame and get the lane equation by regression. The CNN inference module is based on the Ultra Fast Lane Detection in [288, 289]. The CNN baseline used a lightweight ResNet-18 version which trained on CULane and CurveLanes datasets [290]. The inference speed can achieve about 200 fps. The lane regression is based on polynomial fitting and the coordinate transform is based on the external calibration transform mentioned in the previous section.

A.6 Shuttle Bus Platform and Sensor Setup

In our real test driving data collection with the WATonoBus platform, the sensor frames and transforms are illustrated in Fig. A.3. The global points (X_i^w, Y_i^w, Z_i^w) can be mapped to the image correspondences (X_i^c, Y_i^c) using the homography transformation (A.3) under the assumption that the points are on the single plane.

$$\begin{bmatrix} X_i^c \\ Y_i^c \\ 1 \end{bmatrix} = s \cdot \mathbf{K} \cdot \mathbf{M} \cdot \begin{bmatrix} X_i^w \\ Y_i^w \\ Z_i^w \\ 1 \end{bmatrix}. \tag{A.3}$$

The camera intrinsic matrix $\mathbf{K}$ is known by the camera calibration in advance. Additionally, the extrinsic matrix $\mathbf{M}$ and the scaling factor s can be estimated with the fixed setup shown in Fig. A.3 with few sampled data using various process [291]. The coordinate at the vehicle frame can be transformed onto the image plane backward with the projection matrix $\mathbf{K}[\mathbf{R}_{cv}^T \| -\mathbf{R}_{cv}^T \mathbf{t}_{cv}]$ where $\mathbf{R}_{cv}$ and $\mathbf{t}_{cv}$ is the rotation matrix and translation vector of the camera mounted in the vehicle coordinate frame.

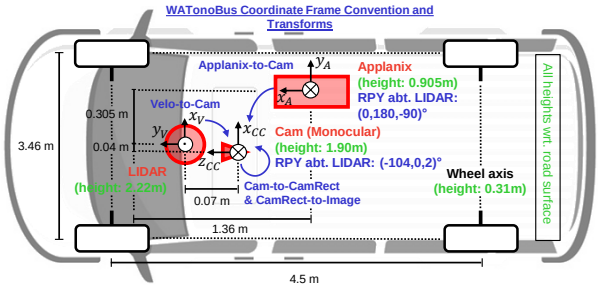


Figure A.3: The WATonoBus platform coordinates frame convention and transforms.

The lcoal map reference of the lane and the shuttle bus location in the ring road region is presented in Fig. A.4. Sample camera image and Lidar point clouds collected are presented in Fig. A.5. In our experiment, only the center camera image is used for the lane estimation.

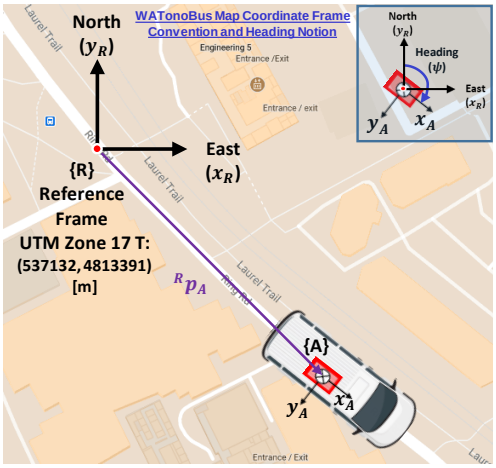


Figure A.4: The WATonoBus local map reference and vehicle heading notion.

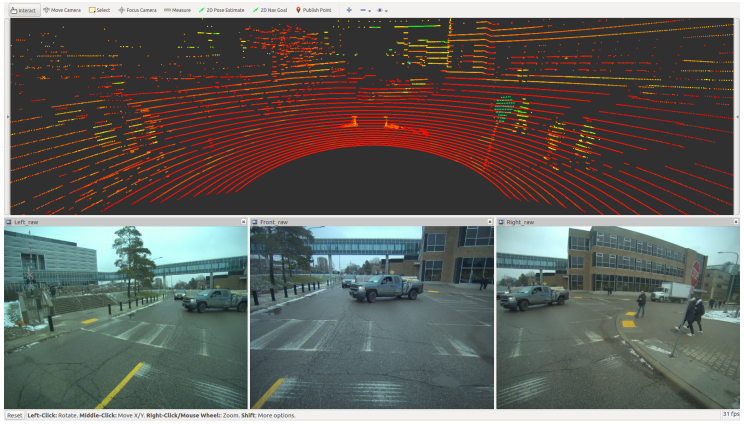


Figure A.5: The sample data collected from the WATonoBus experiment platform.