TUM School of Computation, Information and Technology
Technical University of Munich

Master's Thesis in Robotics, Cognition, Intelligence

Supervised 3D Perception on Roadside LiDARs Under Different Weather Situations

Überwachte 3D-Wahrnehmung auf Infrastruktur-LiDARen unter verschiedenen Wetterbedingungen

Supervisor Prof. Dr.-Ing. habil. Alois C. Knoll

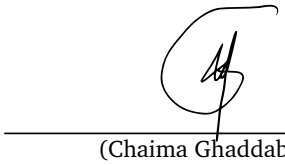
Advisor Walter Zimmer, M.Sc., Xingcheng Zhou, M.Sc

Author Chaima Ghaddab, B.Sc.

Date July 15, 2024 in Munich

Disclaimer

I confirm that this Master's Thesis is my own work and I have documented all sources and material used.



Munich, July 15, 2024 (Chaima Ghaddab, B.Sc.)

Acknowledgements

I am deeply grateful for all the support I received from my advisor **Walter Zimmer**. Completing this study could not have been such a rich experience without your dedication, involvement, and full support. Thank you for all your time and effort.

I am also thankful for **Xingcheng Zhou** for his support in solving all the technical and hardware-related issues so quickly and thoroughly. Thank you for your time and patience.

I would like to express my deep and sincere gratitude to my parents, who set me off on the road to be part of such a challenging degree and never hesitated to encourage me to go forward, who taught me to be ambitious yet patient at the same time. I love you deeply, and I am grateful for you.

I would also like to thank my grandmother, whose prayers and encouragements accompanied me all along the road.

I would like to thank my sisters, Salma, Chahd, and Sarah, for brightening my life with their existence, support, and always being there for me.

I am also as much grateful for all my close friends who were there along this journey to listen to me and cheer me up whenever needed. I would also like to direct special thanks to my dear friend **Bilel Besrour** for proofreading the thesis and providing constructive feedback.

I want to direct special thanks to my team members and coworkers at Predium Technology GmbH, especially my manager, **Mohamed Ali Razouane**, for being patient and accepting of my reduced work schedule along my thesis work-time and showing full support for my studies.

I would also like to thank my co-founders at OrthoCoPilot, **Rafael Vartian**, **Johannes Pawelczyk**, **Marco Rupp**, and my tech team in the project for being considerate of my study schedule and very understanding of my priorities.

Finally, I would like to thank my fantastic choir team, **Arabica**, for being the best source of joy and sense of community for me in Germany and for cheering me up along this journey.

Abstract

LiDAR technology is necessary for applications such as autonomous driving. However, adverse weather conditions like rain, snow, and fog significantly impact the quality and reliability of state-of-the-art LiDAR 3D detectors. This thesis investigates the enhancement of LiDAR-based 3D object detection under such conditions by employing data augmentation and point cloud concatenation techniques to improve the performance of the baseline PointPillars model used in the AUTOtech.agil project [Kra22]. Realistic rain and snow effects are introduced into the point clouds using the LISA library [Kil+21], enhancing the model's resilience to weather-induced data distortions. Combined point cloud concatenation and point cloud filtering techniques are further used to improve LiDAR performance by increasing the density of data points. These methods fill gaps in the data, ensuring detailed environmental models even in adverse weather conditions. The experimental results demonstrate significant improvements in the PointPillars model's accuracy and robustness. While the point cloud concatenation method only increased the mean AP of the model by $\approx 1\%$, data augmentation resulted in an average precision (AP) increase from 63.01% to 66.49%. We also show that the augmentation improves the model's resilience to unseen adverse weather, such as fog. By enhancing LiDAR's resilience to environmental factors, this research contributes to safer and more reliable autonomous systems capable of operating in various conditions. The code for our implementation can be found under https://gitlab.lrz.de/providentiaplusplus/toolchain.

Zusammenfassung

Die LiDAR-Technologie ist für Anwendungen wie das autonome Fahren sehr erforderlich. Allerdings beeinträchtigen ungünstige Wetterbedingungen wie Regen, Schnee und Nebel die Qualität und Zuverlässigkeit von modernen LiDAR-3D-Detektoren erheblich. Diese Arbeit untersucht die Verbesserung der LiDAR-basierten 3D-Objekterkennung unter solchen Bedingungen durch den Einsatz von Datenerweiterungs- und Punktwolkenverkettungstechniken, um die Leistung des PointPillars-Basismodells zu verbessern, das im AUTOtech.agil Projekt [Kra22] verwendet wird. Mit Hilfe der LISA-Bibliothek [Kil+21] werden realistische Regen- und Schneeeffekte in die Punktwolken eingefügt, um die Widerstandsfähigkeit des Modells gegenüber wetterbedingten Datenverzerrungen zu erhöhen. Eine kombinierte Punktwolken-Verkettung und Punktwolken-Filterungstechniken werden weiter eingesetzt, um die LiDAR-Leistung durch Erhöhung der Dichte der Datenpunkte zu verbessern. Mit diesen Methoden werden Datenlücken geschlossen, so dass auch bei ungünstigen Wetterbedingungen detaillierte Umweltmodelle erstellt werden können. Die experimentellen Ergebnisse zeigen signifikante Verbesserungen in der Genauigkeit und Robustheit des PointPillars-Modells. Während die Methode der Punktwolkenverkettung den mittleren AP des Modells nur um etwa 1% erhöhte, führte die Datenerweiterung zu einer Steigerung der durchschnittlichen Genauigkeit (AP) von 63,01% auf 66,49%. Wir zeigen auch, dass die Datenerweiterung die Widerstandsfähigkeit des Modells gegenüber ungünstigen Wet-

terbedingungen wie Nebel verbessert. Durch die Verbesserung der Widerstandsfähigkeit von LiDAR gegenüber Umweltfaktoren trägt diese Forschung zu sichereren und zuverlässigeren autonomen Systemen bei, die unter verschiedenen Bedingungen arbeiten können. Der Code für unsere Implementierung ist unter https://gitlab.lrz.de/providentiaplusplus/toolchain/ zu finden.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Research Gap . . . 1
- 1.3 Contributions . . . 2
- 1.4 Thesis Structure . . . 3

2 Background **5**
- 2.1 LiDAR-Based Object Detection . . . 5
  - 2.1.1 LiDAR Sensors . . . 5
  - 2.1.2 LiDAR Point Cloud Creation . . . 6
  - 2.1.3 LiDAR-based 3D object detection . . . 7
- 2.2 Baseline Model - PointPillars . . . 9
  - 2.2.1 Data Preprocessing Pipeline . . . 9
  - 2.2.2 Data augmentation . . . 11
  - 2.2.3 Voxel Feature Encoding Network (VFE) . . . 11
  - 2.2.4 2D Convolutional Backbone . . . 12
  - 2.2.5 Detection Head . . . 12
  - 2.2.6 Loss Configuration . . . 12
  - 2.2.7 Optimization . . . 13
  - 2.2.8 Performance and Efficiency . . . 13
- 2.3 Dataset and Infrastructure . . . 13
  - 2.3.1 Infrastructure . . . 13
  - 2.3.2 TUMTraf-I Intersection Dataset . . . 14
- 2.4 Physical Effect of Adverse Weather Conditions on Point Cloud Data . . . 16
  - 2.4.1 Rain . . . 16
  - 2.4.2 Snow . . . 16
  - 2.4.3 Fog . . . 17
- 2.5 Evaluation Metrics . . . 18
  - 2.5.1 Precision . . . 18
  - 2.5.2 Recall . . . 18
  - 2.5.3 Average Precision . . . 18
  - 2.5.4 Intersection Over Union: IoU . . . 19
  - 2.5.5 Corruption Error (CE) . . . 19
  - 2.5.6 Resilience Rate (RR) . . . 20
- 2.6 Similarity Calculation Algorithm: ICP . . . 21

3 Related Work **23**
- 3.1 Impact of Adverse Weather Conditions on LiDAR Perception . . . 23
  - 3.1.1 Rain . . . 24
  - 3.1.2 Snow . . . 25
  - 3.1.3 Fog . . . 25

3.2 State-of-the-Art Approaches to Mitigate the Weather's Impact on LiDAR 25
3.2.1 Denoising Point Clouds 26
3.2.2 Energy-based Detection of Adverse Weather Conditions 26
3.2.3 Noise Robustness Loss 27
3.2.4 Densification Methods 28
3.2.5 Data Augmentation Methods 28
3.2.6 Sequence Concatenation in Time 30
3.2.7 Summary 32

4 Methodology **33**
4.1 Data Augmentation 33
4.1.1 Algorithm Description 33
4.1.2 Augmentation Results 37
4.2 Backwards Point Concatenation 40
4.2.1 Algorithm Description 40
4.2.2 Point Filtering Algorithm 41

5 Experiments and Evaluation **45**
5.1 Experimental Setup 45
5.1.1 Sensors 45
5.1.2 Dataset Statistics 45
5.1.3 Evaluation of the Baseline Model 48
5.2 Data Augmentation 50
5.2.1 Experiment 1: Original Test Dataset 52
5.2.2 Experiment 2: Augmented Test Dataset with Unseen Conditions (Fog) . . 55
5.2.3 Experiment 3: Augmented Test Dataset with Seen Conditions (Rain) . . . 59
5.2.4 Runtime Performance 60
5.2.5 mCE and mRR Analysis 62
5.2.6 Conclusions 62
5.3 Backwards Point Concatenation 65
5.3.1 Backwards Concatenation without Filtering 65
5.3.2 Hyperparameter Tuning 65
5.3.3 Performance Evaluation 67

6 Conclusion and Future Work **69**
6.1 Conclusion 69
6.2 Future Work 71
6.2.1 Increasing Dataset Size 71
6.2.2 Improving the Spacial Accuracy 71
6.2.3 Improving the Data Augmentation Pipeline 71
6.2.4 Improving the Point Concatenation Approach 71
6.2.5 Improving Runtime Performance 72
6.2.6 Use of Generative Models 72

A Appendix 1 **73**
A.1 Experiments and Results 73
A.1.1 Experiment 1: Data augmentation - Full original Test Dataset 74
A.1.2 Experiment 2: Data augmentation - Full original Test Dataset with Fog augmentation 78
A.1.3 Experiment 3: Data augmentation - Full original Test Dataset with Seen Conditions (Rain) 79

List of Figures

1.1 Point cloud comparison for a day, clear weather vs. night, rainy weather. 2
1.2 Model's 3D detection comparison for a day, clear weather vs. night, rainy weather. The blue boxes represent the ground truth, and the pink boxes represent the detections. 2

2.1 Examples of mechanical LiDAR sensors. 6
2.2 Visualization of LiDAR point cloud generation in different scenarios. 7
2.3 Overview of the technical steps in LiDAR-based 3D object detection: (1) Point Cloud Preprocessing, (2) Feature Extraction, (3) Object Detection and Classification, and (4) Post-Processing. 8
2.4 Overview of the baseline PointPillars model architecture [Lan+19; Zim+23a; Ngu23] 10
2.5 Setup of the TUMTraf Intersection Dataset infrastructure with two cameras and two LiDAR sensors mounted on a gantry at the intersection of Schleißheimer Straße (B471) and Zeppelinstraße in Garching, near Munich, Germany [Zim+23c]. 14
2.6 Visualization of the scattering effect of rain, snow, and fog particles on the laser pulses. 18

3.1 Comparison of LiDAR scans in normal weather (left) and rainy weather (right). Brighter color corresponds to higher intensity values [Zha+21]. 25

4.1 Pipeline of the Augmentation Process. Note that X *in* [20, 40, 60, 80, 100]% and similarity threshold varies from one adverse weather to another. 36
4.2 Original Point Cloud and RGB Image 37
4.3 Point Cloud and RGB Image with Rain Rate of 2.5 mm/hr 38
4.4 Point Cloud and RGB Image with Rain Rate of 5.6 mm/hr 38
4.5 Point Cloud and RGB Image with Rain Rate of 18.8 mm/hr 38
4.6 Point Cloud and RGB Image with Snow 39
4.7 Point Cloud and RGB Image with Fog 39
4.8 Concatenation Pipeline using point filtering based on minimum and maximum distance threshold to original point cloud 43

5.1 3D point cloud generated by the Ouster OS1-64 LiDAR sensor. 46
5.2 Images captured by the Basler ace camera facing south. 46
5.3 Images captured by the Basler ace camera with projected point clouds using calibration data. 46
5.4 Object class count for train, validation, and test datasets. 47
5.5 Average number of points in the point clouds for sunny and rainy weather conditions in the train, validation, and test datasets. 48
5.6 Examples from the TUMTraf dataset showing LiDAR point clouds under different weather conditions. 49

5.7 False negatives detected in night rainy scenes using the baseline model. Blue boxes represent ground truth, and pink boxes represent the model's detections. . 51
5.8 Detection results of the baseline model (top left) and the five augmentation results at a day vs. night rainy scene. The boxes in blue represent the ground truth, and the ones in pink represent the detections. 54
5.9 Detection results of the baseline model and the five augmentation results at a day foggy scene highlighting the improvement of detection results using adverse weather augmentation for training. Blue boxes represent ground truth, and pink boxes represent the model's detections. A video showing the fog-augmented test scenes with detections by the baseline model and the 60% augmented model can be found under the link:https://rb.gy/55ka8j 58
5.10 Detection results of the baseline model (top left) and the five augmentation results at the rain-simulated scene with rain rates 5.6mm/h (left) and 25mm/h (right), which is the same rain rate used for training. (The rain effect is only visible on the scattered point cloud). The results highlight the augmentation model's ability to detect objects at further ranges and objects with scattered points that the baseline model could not detect. Blue boxes represent ground truth, and pink boxes represent the model's detections. 61
5.11 Comparison of AP - Mean Corruption Error (mCE) and Mean Resilience Rate (mRR). 63
5.12 Heatmaps representing the average precisions of the different augmentation models and the baseline model for the CAR, PEDESTRIAN, BICYCLE, and BUS classes 64
5.13 Evaluation Results of the baseline PointPillars model using 1 to 10 concatenation offset values. The performance of the baseline model with 0 frame concatenation is represented by the dashed horizontal line. 66
5.14 Qualitative Results of the baseline PointPillars model using 1 to 10 concatenation offset values. The first image is the original RGB image. 67
5.15 The results of the evaluation of the baseline PointPillars once without point cloud concatenation (left) and once with a 1-Frame concatenation (right). Blue boxes represent ground truth, and pink boxes represent the model's detections. 67
5.16 Evaluation results of the baseline PointPillars model using minimum and maximum point distance ranging from 0.5 to 1. 68

6.1 Open issues of augmented models: Misdetections and Misclassifications. Blue boxes represent ground truth, and pink boxes represent the model's detections. . 70

List of Tables

2.1 Summary of the TUMTraf Intersection Dataset 15

3.1 Comparison of Mean Average Precision (mAP) Values for "Car" Class on the KITTI Dataset under Simulated Fog and Snow Conditions according to [Kil+21]. The values that show the highest impact are underlined. 24
3.2 The Corruption Error (CE) of 7 detectors on KITTI-C. Bold: Best in column, Red: Worst in row. [Kon+23] 24
3.3 Car 3D AP@.5IoU results on all relevant test splits of the Seeing Through Fog dataset [Bij+20] with easy, moderate, and hard fog conditions [Hah+21]. 29
3.4 A comparison of mAP values of networks retrained on simulated rainy LiDAR poitn clouds using LISA and evaluated on real rainy scenes from the Waymo dataset [Sun+20] for the class “Vehicle”, based on the results from [Kil+21]. . . 31
3.5 Evaluation results for IC with and without Temporal Offset, according to [Kem+23a]. Inference time in seconds per input sequence. (*) indicates use of temporal offset. 31
3.6 Summary of methods and key results. mAP Drop: Drop in performance when tested under adverse weather conditions. mCE: mean Corruption Error. AP Improvement: Improved average precision when trained with augmented data compared to the baseline model. Red: Densification. Blue: Augmentation. Green: Noise Robustness Loss. 32

5.1 Performance Metrics of Baseline PointPillars Model 49
5.2 Runtime Performance of PointPillars Model with registered North and South LiDAR sensors 50
5.3 Performance Metrics and IoU of Baseline PointPillars Model under Night Rainy Scenes 50
5.4 Number of Point Clouds After Augmentation 52
5.5 Summarized Mean IoU, Precision, and Recall metrics for overall, day, and night scenes across various augmentation levels. 53
5.6 Summarized Average Precision (AP) metrics across all scenes, day scenes, and night scenes for various augmentation levels. 55
5.7 Summarized Average Precision (AP) metrics for clear, foggy, and full data (clear + foggy) across various augmentation levels. 56
5.8 Summarized Mean IoU, Precision, and Recall metrics for clear, foggy, and overall scenes across various augmentation levels. 57
5.9 Summarized Mean IoU, Precision, and Recall metrics for overall scenes in Experiments 3.1 and 3.2 across various augmentation levels. 59
5.10 Summarized Average Precision (AP) metrics for Experiments 3.1 and 3.2. 60
5.11 Runtime performance of the baseline PointPillars model and the five augmented models evaluated on the original test dataset using RTX 3090. 60

A.1 The Average Precision (AP) metrics for various augmentation levels across all scenes. It includes AP for each object class (CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY, OTHER) at different distances (0-40m, 40-50m, 50-64m). 74
A.2 Performance (AP) metrics for night scenes, detailing performance for each class at different distances under night conditions. 75
A.3 IoU, Precision, and Recall metrics for overall scenes, comparing spatial accuracy across augmentation levels for each class at various distances 76
A.4 IoU, Precision, and Recall metrics for night scenes, comparing spatial accuracy across augmentation levels for each class at various distances 77
A.5 Performance (AP) metrics for fog-augmented scenes, detailing performance for each class at different distances under foggy conditions. 78
A.6 IoU, Precision, and Recall metrics for Experiment 3.1, comparing spatial accuracy across augmentation levels for each class at various distances. 79
A.7 the Average Precision (AP) metrics for various augmentation levels across all scenes in Experiment 3.1. It includes AP for each object class (CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY, OTHER) at different distances (0-40 m, 40-50 m, 50-64 m). 80
A.8 Average Precision (AP) metrics for various augmentation levels across all scenes in Experiment 3.2. It includes AP for each object class (CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY, OTHER) at different distances (0-40 m, 40-50 m, 50-64 m). 81
A.9 Precision, Recall, and Intersection over Union (IoU) metrics for various augmentation levels across all scenes in Experiment 3.2. It includes precision, recall, and IoU for each object class (CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY, OTHER) for the overall scene. 82

Listings

4.1 Rain simulation using LISA library 34
4.2 Snow simulation using LISA library 34

Chapter 1

Introduction

1.1 Motivation

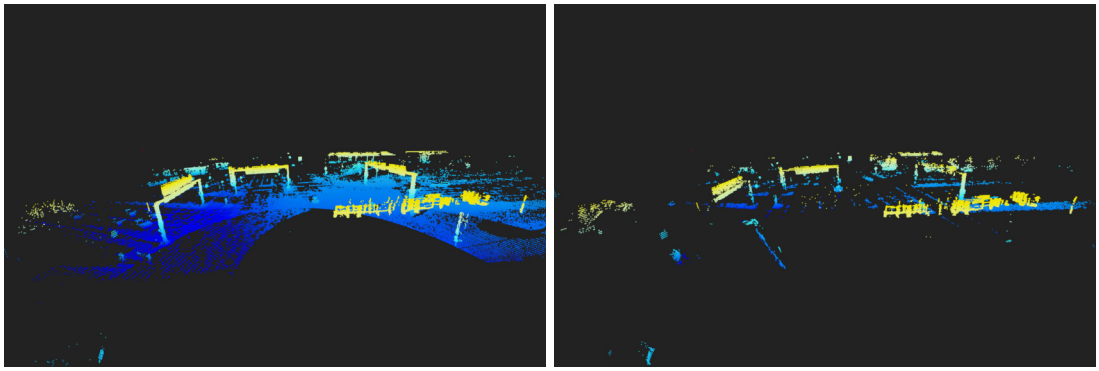
LiDAR (Light Detection and Ranging) technology provides high-resolution, three-dimensional environmental information about the object where a LiDAR sensor is mounted, which makes it essential in applications such as autonomous driving, where accurate perception of the surroundings is necessary for safe navigation. The AUTOtech.agil project [Kem+23b], the successor of the Providentia++ project [Bli+22], uses such technologies to improve roadside infrastructure for traffic monitoring and autonomous vehicle support.

Despite the advances in LiDAR technology, studies [JKP19; Mon+21; Zha+23] have proven a decrease in its performance in adverse weather conditions. Weather conditions like rain, snow, and fog introduce noise and distortions in the LiDAR-captured point clouds [Kut+20], reducing the accuracy and reliability of 3D object detection models. While rain can cause power loss and signal attenuation due to water droplets interacting with the laser beam, snow can obstruct the visibility range of LiDAR sensors, creating false detections and reducing detection distance.

These scattering and absorption of light caused by adverse weather conditions can significantly distort the creation of the point clouds, leading to false and missing detections. These challenges result in sudden braking and lane changes, potentially increasing traffic accidents, making it necessary to use advanced algorithms and robust data augmentation techniques to mitigate these effects. Research has shown that physics-based and empirical models can help simulate adverse weather conditions in LiDAR datasets, improving the robustness of detection algorithms in real-world scenarios [Liu+19].

1.2 Research Gap

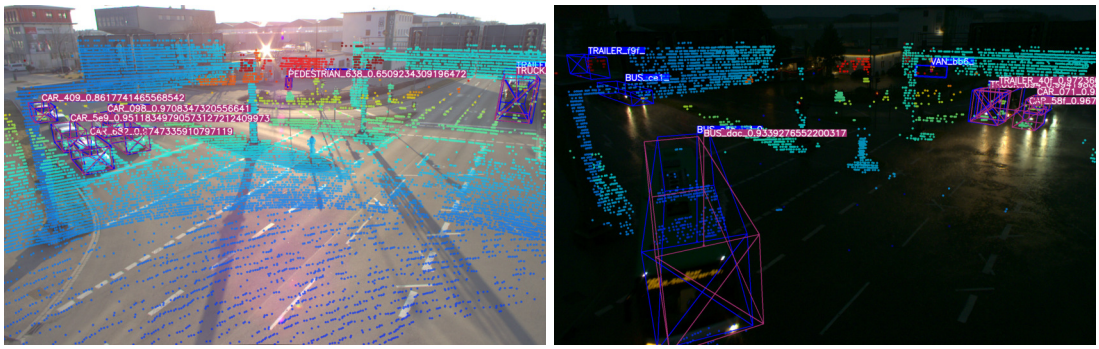
The PointPillars-based 3D detection pipeline developed in [Zim+23a] demonstrates advancements in multi-modal 3D object detection and highlights limitations under adverse weather conditions. [Zim+23a] employs an early fusion of two LiDAR sensors and further incorporates monocular camera detections to improve robustness and detect small objects. Despite the system's proven performance in clear weather conditions, the performance in adverse weather conditions remains challenging, mainly due to the sparse point clouds and increased noise. The evaluation results of the PointPillars-based model used in [Zim+23a] show a drop in performance in mAP by 19.7% when comparing clear weather conditions and rainy night conditions. Figure 1.1 shows the drop in the number of points and the point cloud density when comparing the rainy night point cloud to the clear weather point cloud in the TUMTraf-I dataset [Zim+23c] from the A9 intersection dataset [Zim+23b] used in [Zim+23a].



(a) Point cloud during clear weather conditions. **(b)** Point cloud during rainy night conditions.

Figure 1.1: Point cloud comparison for a day, clear weather vs. night, rainy weather.

Besides, Figure 1.2 shows the drop in 3D detection performance of the PointPillars-based model used in [Zim+23a] in daylight (left) compared to rainy night conditions (right).



(a) Model's 3D detection during clear daylight weather. **(b)** Model's 3D detection during rainy weather conditions.

Figure 1.2: Model's 3D detection comparison for a day, clear weather vs. night, rainy weather. The blue boxes represent the ground truth, and the pink boxes represent the detections.

Hence, the primary challenge addressed in this thesis is the degradation of LiDAR data quality and subsequent reduction in the performance of 3D object detection models under adverse weather conditions. While solutions such as real data collection, manual labeling, and the use of simulation have a positive effect on the simulation of adverse weather conditions [Hei+19], they are often limited in their effectiveness due to instrument-related added errors [Mäy+17] or are computationally expensive. In this thesis, we intend to implement and test innovative approaches to enhance the robustness of LiDAR-based object detection models while maintaining a reasonable computational complexity.

1.3 Contributions

To improve the performance of existing models for LiDAR 3D perception within the AUTOtech.agil project [Kem+23b], this thesis introduces a data augmentation pipeline that serves as a preprocessing step to enhance the dataset's coverage for adverse weather conditions. Data augmentation has been used in several studies [Hah+21; Kil+21; Hah+22] to mitigate the challenges resulting from adverse weather conditions. We also extend the inference pipeline with a point cloud densification step using a point concatenation approach. Furthermore, we extend the evaluation pipeline with task-specific metrics that measure the impact of the adverse weather's

corruption of the point cloud and the model's resilience to these corruptions. The key contributions of this thesis can be summarized as follows:

- The integration of a robust data augmentation framework for simulating adverse weather conditions to the AUTOtech.agil project.
- The implementation of a point cloud concatenation algorithm to enhance the quality and density of LiDAR point clouds.
- A comprehensive experimental validation of the proposed methods.
- Contribution to developing a safer and more reliable autonomous driving system using infrastructure LiDAR data from the Providentia++ Testbed [Bli+22].

1.4 Thesis Structure

The thesis is structured as follows:

- **Chapter 2**: In this chapter, we provide the background information and theoretical foundations necessary to understand the research topic and the approaches implemented to mitigate the adverse weather conditions' impact on the 3D detection task.
- **Chapter 3**: We provide a literature review of existing research on the thesis topic. In this chapter, we explain, analyze, and present the limitations of state-of-the-art approaches to solve the thesis's challenge.
- **Chapter 4**: This chapter explains the approaches and methods used in this thesis, providing the algorithms and frameworks we used.
- **Chapter 5**: This chapter presents the experiments we conducted in the scope of this thesis, the results we obtained, and an analysis of the performance improvements.
- **Chapter 6**: We finally summarize the key findings and contributions in this chapter and suggest directions for future research.

Chapter 2

Background

This chapter provides the background knowledge needed to understand the important topics discussed in this thesis. As the thesis aims to optimize LiDAR 3D sensing, it is essential to understand how LiDAR sensors work and how the 3D detection pipeline is built as a baseline for the optimization approaches.

2.1 LiDAR-Based Object Detection

2.1.1 LiDAR Sensors

A LiDAR sensor is an electronic device that emits many laser pulses per second to measure distances and ranges. The distance from the LiDAR sensor to the objects surrounding it is calculated using the time it takes for the pulses to return to the sensor. According to a review from [Li+22] on solid-state LiDAR and nanophotonics-based LiDAR sensors, modern LiDAR systems can operate at different pulse rates where advanced systems can generate hundreds of thousands of pulses each second, facilitating high-resolution data collection and environmental design. For instance, studies have shown that LiDAR can detect small objects on the road from distances of up to 100 meters with an accuracy deviation as low as 2 centimeters, which is vital for early obstacle detection and collision avoidance [Li+22].

The technology behind LiDAR sensors has evolved with modifications such as solid-state LiDAR that offer cost, size, mechanical robustness, and energy consumption advantages. Solid-state LiDAR systems have no continuous parts and consist of optical phased array (OPA) or Micro-Electro-Based technologies such as mechanical systems (MEMS), which contribute to robustness and compactness [Li+22]. Such sensors include the Velodyne HDL-64E, the Velodyne VLP-16, and the Ouster OS1, all known for their high fluidity, stability, and reliability in 3D mapping.

Datasets Using Velodyne and Ouster LiDAR Sensors

Mechanical LiDAR sensors, especially Velodyne, are widely used in many datasets that have become benchmarks in autonomous driving and robotics. These datasets provide rich and diverse data needed to develop and test algorithms for 3D sensing, object recognition, and autonomous navigation.

KITTI Vision Analysis Suite The KITTI dataset is one of the most used datasets for autonomous driving scenarios. It provides comprehensive statistics on real-world driving situations, including stereo imaging, optical flow, optical odometry, 3D object recognition, and 3D

analysis [GLU12]. The Point cloud frames of the KITTI dataset were recorded with the Velodyne HDL-64E LiDAR sensor shown in Figure 2.1a.

nuScenes dataset The nuScenes dataset generated by nuTonomy provides a rich dataset from urban driving scenarios. It includes camera, radar, and LiDAR sensor data captured under different conditions and labeled with 3D bounding boxes for objects [Cae+20]. These LiDAR point clouds of the nuScenes dataset were captured using the Velodyne HDL-32E LiDAR model shown in Fig. 2.1b. The advantage of the nuScenes dataset is that it contains labeled radar data, which is less affected by adverse weather [Zim+22].

A9 assembly dataset A9 Intersection dataset provides labeled LiDAR point clouds and synchronized camera images from roadside sensors installed on a gantry located at the intersection of Schleißheimer Straße (B471) and Zeppelinstraße in Garching near Munich, Germany [Zim+23b]. This dataset aims to improve the 3D perception of urban areas with its point cloud dataset captured with an Ouster OS1-64 (gen. 2) LiDAR, shown in Fig 2.1c.

(a) Velodyne HDL-64E mechanical LiDAR sensor, used in the KITTI dataset [1].

(b) Velodyne HDL-32E mechanical LiDAR sensor, used in the nuScenes dataset [2].

(c) Ouster OS1-64 (gen. 2), used in the A9 Intersection dataset [3].

Figure 2.1: Examples of mechanical LiDAR sensors.

2.1.2 LiDAR Point Cloud Creation

LiDAR point cloud formation involves the emission of laser pulses from the LiDAR sensor towards the surrounding objects. Upon contact with an object in the LiDAR's environment, a pulse is returned back towards the LiDAR sensor. Each pulse to be returned is measured and used to calculate the distance to the object, resulting in a collection of 3D coordinates known as a point cloud. This point cloud represents the spatial structure of the sensor's environment, capturing detailed geometric information about objects and surfaces, such as vehicles, pedestrians, and infrastructure in the context of autonomous driving. Preprocessing steps such as noise filtering and ground segmentation are applied to the raw point cloud to enhance quality and focus on relevant features for further analysis.

In most scenarios, the LiDAR sensors are mounted on autonomous vehicles or the traffic infrastructure. LiDAR sensors in autonomous vehicles are typically mounted on the roof or sides to provide a 360-degree view of the surrounding environment, which helps the car to detect and respond to surrounding objects in real time. In contrast, infrastructure-mounted LiDAR systems are installed on fixed structures such as traffic lights, bridges, or buildings, as visualized in Figure 2.2.

[1] Source of Subfigure 2.1a: https://www.aeroexpo.online/de/prod/velodyne/product-176220-32081.html
[2] Source of Subfigure 2.1b: https://www.aeroexpo.online/de/prod/velodyne/product-176220-32080.html
[3] Source of Subfigure 2.1c: https://general-laser.at/en/shop-en/ouster-os1-64-lidar-sensor-en

(a) LiDAR mounted on an autonomous vehicle

(b) LiDAR mounted on infrastructure

Figure 2.2: Visualization of LiDAR point cloud generation in different scenarios.

2.1.3 LiDAR-based 3D object detection

LiDAR-based 3D object detection is essential in autonomous driving and other applications where understanding the environment in three dimensions is a fundamental concept [AB22]. This technology uses complex algorithms to process the point clouds generated by LiDAR sensors, identifying and classifying objects within a scene. For example, segmentation algorithms may classify data points corresponding to a single object, effectively isolating cars, pedestrians, and other relevant objects from the surrounding environment. LiDAR 3D perception involves several technical steps, as shown in Figure 2.3.

Point Cloud Preprocessing

The raw data LiDAR sensors collect is a dense cloud of points representing the 3D space. Preprocessing involves filtering noise, down-sampling to reduce data size, and segmenting ground points to focus on objects of interest. Techniques like voxel grid filtering and statistical outlier removal are commonly used to clean the point cloud.

Feature Extraction

Feature extraction transforms the raw point cloud into a set of descriptive features. These features serve at a later point as the input to the 3D detection pipeline. While traditional methods involve features such as geometric shapes and statistical properties, recent advancements use deep learning models to learn features from the point cloud data automatically. Models like PointNet and PointNet++ [Qi+17] directly operate on raw point clouds and capture spatial relationships among points.

Object Detection and Classification

The task of 3D object detection comprises identifying and classifying objects within the processed point cloud. This involves:

- **Region Proposal:** Consists of generating candidate regions in the point cloud where objects might be located. Methods like 3D sliding windows, voxel-based proposals, and point-based proposals are used.
- **Classification:** Consists of assigning a label to each proposed region resulting from the previous step. Deep learning models, like convolutional neural networks (CNNs), play a significant role here. PointNet [Qi+17], PointPillars [Lan+19], and VoxNet [MS15] are examples of architectures used for this task.

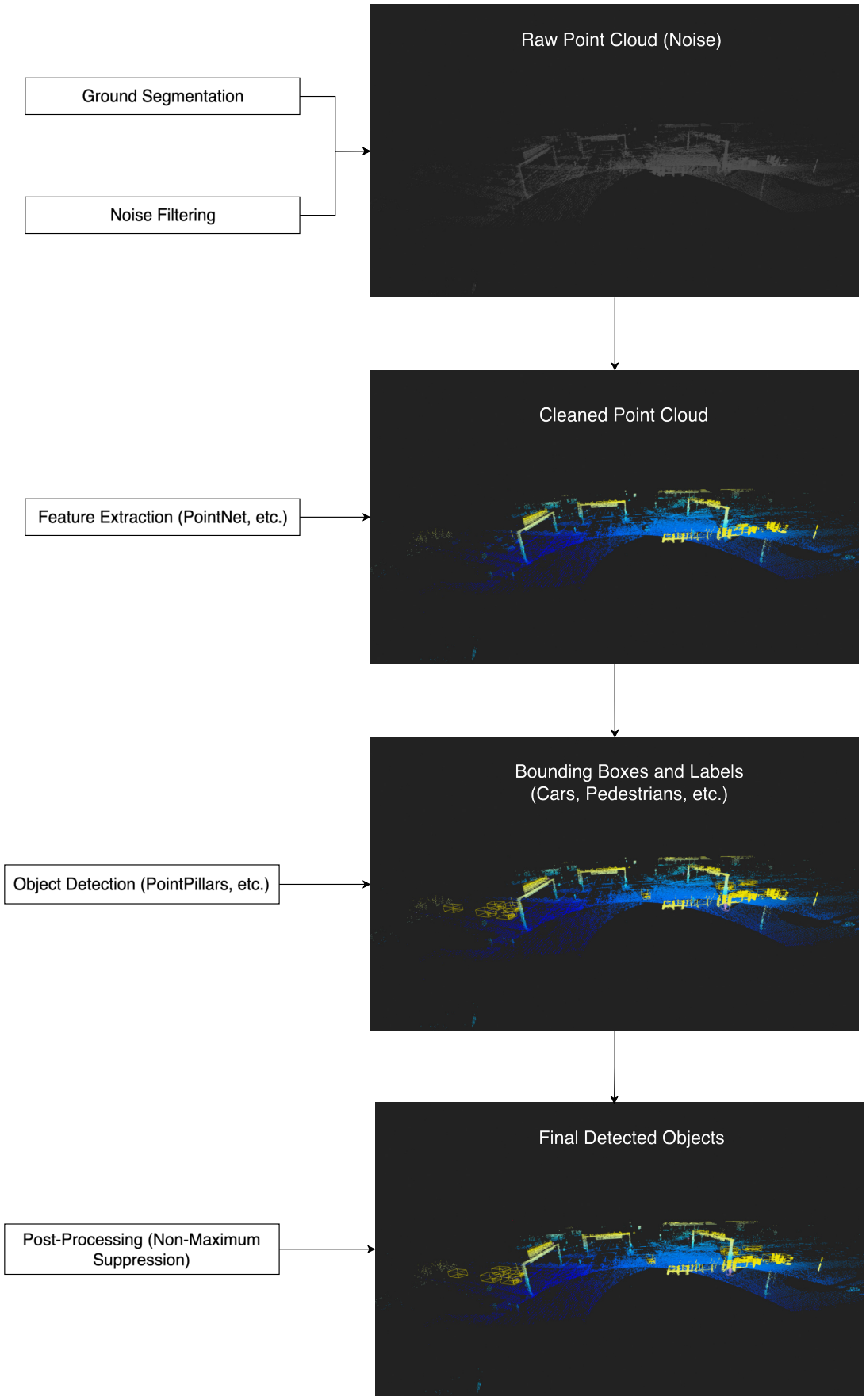


Figure 2.3: Overview of the technical steps in LiDAR-based 3D object detection: (1) Point Cloud Preprocessing, (2) Feature Extraction, (3) Object Detection and Classification, and (4) Post-Processing.

Post-Processing

Post-processing refines the detection results by eliminating redundant proposals and smoothing object boundaries. Non-maximum suppression (NMS), used in [Lan+19], is commonly used to combine overlapping detections into a single coherent detection.

A benchmark study by [ZT17] on various deep learning models, including PointNet [Qi+17] and VoxelNet [MS15], demonstrated a substantial improvement in object recognition rates. These models were trained on extensive labeled datasets, such as the KITTI dataset, and achieved classification accuracies up to 80% for vehicular objects and 60% for pedestrian detection [ZT17].

2.2 Baseline Model - PointPillars

The AUTOtech.agil project uses a PointPillars-based model to achieve efficient and accurate LiDAR-based 3D object detection on the point clouds of the TUMTraf-I dataset [Zim+23c]. First introduced in [Lan+19] in 2019, PointPillars enables end-to-end learning using only 2D convolutional layers, avoiding the complexity and computational burden of 3D convolutions. The architecture of the baseline model that we will use as a basis in our experiments was described in [Lan+19] and consists of three main modules: a feature encoder network, a 2D convolutional backbone, and a detection head, as shown in Figure 2.4. Figure 2.4 illustrates the model's architecture and the training configuration used to obtain the baseline results, to which we will compare our results.

2.2.1 Data Preprocessing Pipeline

The data preprocessing pipeline ensures that the raw point cloud data is effectively transformed and prepared for the neural network. The pipeline includes several steps:

- **Masking Points and Boxes Outside the Defined Range**: To ensure that the data fed into the model is within the spatial boundaries relevant for the detection task, this step involves removing any points and bounding boxes outside the specified point cloud range, which improves the efficiency and focus of the detection process [Lan+19].
- **Shuffling Points for the Training Set**: The points in the training set are shuffled to prevent the model from learning any ordering or patterns in the point cloud data that could lead to overfitting. This sort of randomization helps generalize the model better to unseen data [Lan+19].
- **Voxelization**: The core innovation of PointPillars lies in its ability to convert 3D point clouds into a pseudo-image format through voxelization. Each voxel represents a small partition of the 3D space, with a voxel size of $[0.16, 0.16, 8]$ and contains a maximum of 32 points to maintain computational efficiency [Zim+23a].

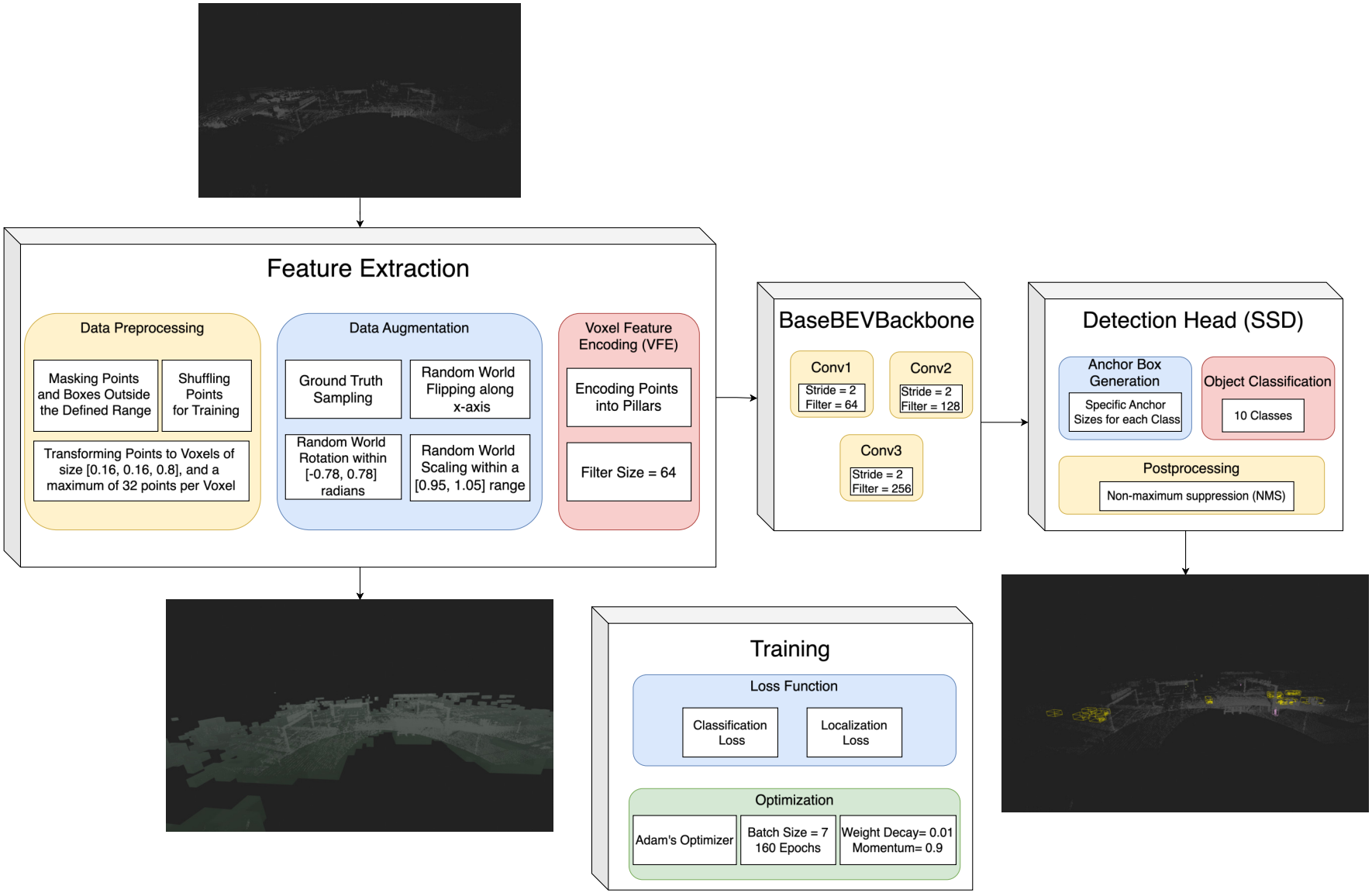


Figure 2.4: Overview of the baseline PointPillars model architecture [Lan+19; Zim+23a; Ngu23]

2.2.2 Data augmentation

Data augmentation plays an important role in enhancing the robustness and generalization capability of the PointPillars model. The baseline pipeline contains an initial augmentation process that includes:

- **Ground Truth Sampling**: This technique involves augmenting the training data with sampled ground truth data, ensuring the model is exposed to various object instances and configurations [Lan+19].
- **Random Flipping**: To simulate different viewing perspectives and improve the model's ability to recognize objects from various angles, the point clouds are randomly flipped along the x-axis. This augmentation helps make the model invariant to left-right orientations [Lan+19].
- **Random Rotation**: The point clouds are randomly rotated within a specified range. This rotation augmentation helps the model to be robust against rotational variations and improves its performance in detecting objects with different orientations [Lan+19].
- **Random Scaling**: The point clouds are randomly scaled to account for variations in object sizes and distances. This scaling augmentation ensures that the model can accurately detect objects of varying sizes and distances [Lan+19].

2.2.3 Voxel Feature Encoding Network (VFE)

The VFE layer, PillarVFE, converts the point cloud into a pseudo-image by encoding points into pillars. This layer utilizes absolute coordinates for the points and applies normalization. The VFE layer is designed to handle a filter size of 64, which helps in capturing the spatial distribution of the points within each voxel effectively. By transforming the sparse 3D point cloud into a dense 2D pseudo-image, the VFE layer allows subsequent neural network layers to process the data using standard 2D convolution operations [Lan+19]. This layer has the following components:

Point Cloud to Pseudo-Image Conversion

- The input to the model is a point cloud, where each point is represented by its coordinates (x, y, z) and intensity i.
- The point cloud is discretized into an evenly spaced grid in the x-y plane, creating a set of vertical columns or "pillars". Each pillar contains points within its grid cell [Lan+19].
- Points in each pillar are augmented with additional features: the offset from the arithmetic mean of the pillar points (x_c, y_c, z_c), and the offset from the pillar center (x_p, y_p). This results in a 9-dimensional vector for each point [Lan+19].

Pillar Tensor Creation

Due to the sparsity of the point cloud, most pillars will be empty or contain very few points. The PointPillars model [Lan+19] limits the number of non-empty pillars per sample and the number of points per pillar, creating a dense tensor of size (D, P, N), where D is the feature dimension, P is the number of pillars, and N is the number of points per pillar. Zero padding is used for pillars with fewer points than N [Lan+19].

PointNet Encoding

- A simplified PointNet architecture processes each point within a pillar. A linear layer followed by BatchNorm and ReLU generates a tensor of size (C, P, N), where C is the number of output channels. A max operation over the channels creates an output tensor of size (C, P) [Lan+19].
- The encoded features are scattered back to their original pillar locations to form a pseudo-image of size (C, H, W), H being the height and W the width of the grid [Lan+19].

2.2.4 2D Convolutional Backbone

The backbone network, BaseBEVBackbone, has three convolutional layers with specific strides and filter sizes that progressively reduce the spatial dimensions of the pseudo-image while increasing the depth of the feature maps [Lan+19]. This network structure allows the model to capture high-level features from the input data. Following the convolutional layers, the feature maps are upsampled and concatenated to form a comprehensive feature representation. This upsampling step ensures the spatial resolution is sufficiently restored for accurate object localization [Lan+19; Zim+23a].

Top-Down Network

This network produces features at increasingly smaller spatial resolutions through a series of convolutional blocks, each characterized by a stride S, number of layers L, and number of output channels F [Lan+19].

Upsampling and Concatenation

Features from the top-down network are upsampled and concatenated. Transposed 2D convolutions are used for upsampling the features to a consistent stride relative to the original input pseudo-image. The upsampled features are concatenated to form the final feature map [Lan+19].

2.2.5 Detection Head

The dense head is responsible for the final detection of objects. This component generates anchor boxes and classifies them into predefined object categories. The detection head uses the Single Shot Detector (SSD) framework to predict 3D bounding boxes for objects using bounding box height and elevation as additional regression targets [Lan+19]. This component also includes direction classification to predict the orientation of objects and an anchor generation configuration for each object class to ensure that the anchors match the expected sizes and shapes of the target objects. Finally, ground truth objects are mapped to the generated anchors, facilitating the learning process [Lan+19; Ngu23].

2.2.6 Loss Configuration

The loss function used in training the PointPillars model used in [Lan+19; Zim+23a; Ngu23] combines several components to ensure balanced training. [Lan+19] describes three losses: a classification loss to measure object classification accuracy, a localization loss to evaluate the precision of bounding box predictions, and a direction loss to ensure that the predicted

orientations of objects are accurate. The loss function assigns weights to these components, typically giving more importance to localization to improve the model's detection performance. This multi-component loss function is essential for training a robust and accurate 3D object detection model [Lan+19]. To refine the 3D detections, the PointPillars model employs non-maximum suppression (NMS) and thresholds for score and recall values to eliminate redundant detections, ensuring that only the most confident predictions are retained [Lan+19; Ngu23]. The baseline model developed in [Ngu23; Zim+23a] was trained and evaluated with a 0.3 score.

2.2.7 Optimization

The optimization strategy for the PointPillars model employs an Adam optimizer with a one-cycle learning rate policy, allowing for dynamic adjustment of the learning rate throughout the training process, starting with a lower learning rate of 0.003. The training schedule in [Zim+23a] includes a specific weight decay of 0.01 and a momentum of 0.9 to ensure stable and practical training.

2.2.8 Performance and Efficiency

- **Speed**: PointPillars performs well on the KITTI dataset for detecting cars, pedestrians, and cyclists. It runs at 62 Hz, significantly faster than previous methods like VoxelNet, which operates at 4.4 Hz. The model balances speed and accuracy, with a faster version achieving up to 105 Hz with minimal loss in performance [Lan+19].
- **Accuracy**: In the BEV detection benchmark, PointPillars achieves a mean average precision (mAP) of 66.19%, with solid performance across all classes and severity levels. The authors of [Zim+23a] also report an mAP score of 62.11% on the full A9-I dataset [Zim+23b] (TUMTraf-I [Zim+23c]).
- In the 3D detection benchmark, PointPillars attains a mAP of 59.20%, consistently outperforming other methods in the moderate severity category. PointPillars excels in average orientation similarity (AOS), with a score of 68.86%, outperforming methods such as SECOND and AVOD-FPN. This indicates a high level of precision in predicting the orientation of detected objects.

2.3 Dataset and Infrastructure

This thesis builds up on the recent study in [Zim+23a; Ngu23] that uses the TUMTraf-I dataset [Zim+23c], a subset of the A9 dataset [Zim+23b] initially captured in the Providentia++ testbed. The Providentia++ testbed is extended over 35km, according to [CBK23]. However, to build a valid measurement of the baseline model's improvement, we will focus on the intersection dataset also used in [Zim+23a; Ngu23]. This section provides more insights into the dataset and the infrastructure where the LiDAR capturing the point clouds was mounted.

2.3.1 Infrastructure

The infrastructure supporting data collection and processing operations of the TUM Traffic Intersection dataset (TUMTraf-I) [Zim+23c] is built on advanced sensor technologies and robust

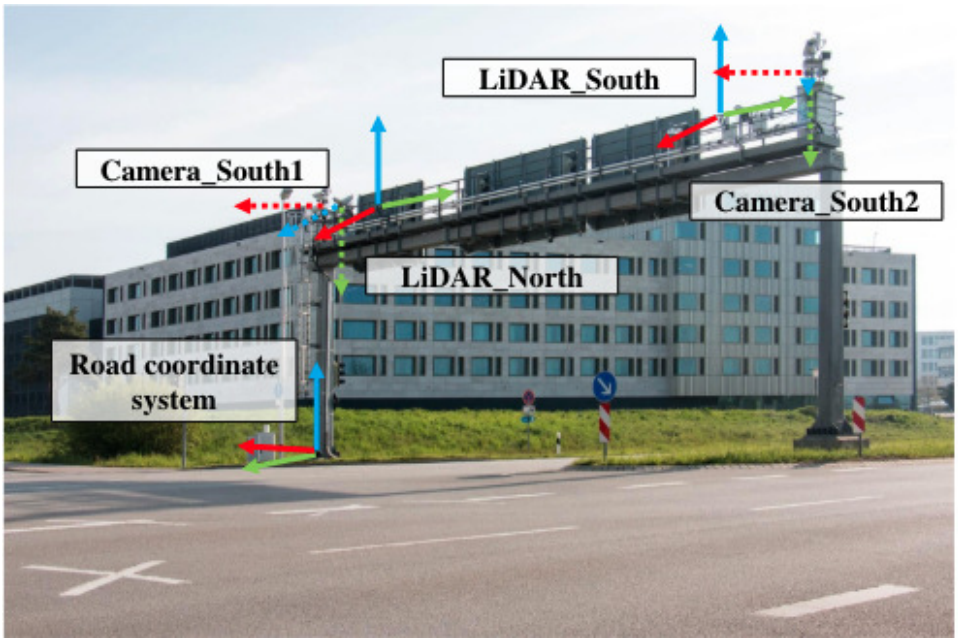


Figure 2.5: Setup of the TUMTraf Intersection Dataset infrastructure with two cameras and two LiDAR sensors mounted on a gantry at the intersection of Schleißheimer Straße (B471) and Zeppelinstraße in Garching, near Munich, Germany [Zim+23c].

data management systems. The setup contains two high-resolution cameras and two LiDAR sensors mounted on gantry bridges to provide comprehensive intersection coverage from elevated vantage points, as shown in Figure 2.5. This strategic placement ensures the capture of detailed, high-quality data necessary for precise 3D perception. The cameras deployed are Basler ace acA1920-50gc models, offering a resolution of 1920×1200 pixels with global shutter and color capabilities. These cameras are equipped with 8 mm lenses and connected via GigE interfaces, ensuring efficient image capture and real-time data transmission [Zim+23c]. The LiDAR sensors used are Ouster OS1-64 (generation 2), featuring 64 vertical layers and a 360-degree field of view. Configured to focus below the horizon, these sensors provide a maximum range of 120 meters with an accuracy of 1.5-10 cm, enabling detailed and accurate 3D mapping of the intersection [Zim+23c].

Data collected from these sensors is processed through a Data Fusion Unit (DFU), which integrates the various data streams into a cohesive digital twin of the traffic environment [Zim+23c]. The DFU also manages the coordinate systems of the individual sensors, aligning them with the road coordinate system for consistent data representation [Zim+23c]. According to the authors of [Zim+23c], temporal synchronization is achieved via a Network Time Protocol (NTP) server, resulting in an average synchronization error of 18.54 ms. At the same time, Spatial calibration between cameras and LiDARs is performed using a targetless automatic calibration method inspired by state-of-the-art research [Zim+23c].

2.3.2 TUMTraf-I Intersection Dataset

The TUMTraf Intersection Dataset is a comprehensive collection of synchronized camera images and LiDAR point clouds curated explicitly for roadside perception tasks. According to [Zim+23c], this dataset was recorded at the intersection of Schleißheimer Straße (B471) and Zeppelinstraße in Garching, near Munich, Germany, as part of the AUTOtech.agil project [Kra22]. It captures many traffic scenarios and conditions, making it an invaluable resource for developing and testing 3D perception algorithms.

Table 2.1: Summary of the TUMTraf Intersection Dataset

Attribute	Value	Description
Number of frames	9,600	Total number of frames
Number of images	4,800	Total number of camera images
Number of LiDAR point clouds	4,800	Total number of LiDAR point clouds
Number of 3D bounding boxes	57,400	Total number of 3D labeled boxes
Number of classes	10	Total number of object classes
Nighttime data	25%	Proportion of data recorded at night
Daytime data	75%	Proportion of data recorded during the day
Maximum range of LiDAR	120 meters	Maximum distance measured by LiDAR
Accuracy of LiDAR	1.5-10 cm	Measurement accuracy of LiDAR
Frame rate (Camera)	25 Hz	Frame rate of camera data collection
Frame rate (LiDAR)	10 Hz	Frame rate of LiDAR data collection

The dataset includes 4.8k frames of camera images and 4.8k LiDAR point clouds labeled by experienced annotators with the 3D bounding box annotation tool [ZRT19], resulting in 57.4k manually labeled 3D bounding boxes representing ten classes of road users: cars, trucks, trailers, vans, motorcycles, buses, pedestrians, bicycles, emergency vehicles, and other objects. These labels are essential for training and evaluating perception algorithms, providing detailed information about each detected object's size, position, and attributes.

The TUMTraf Intersection Dataset is organized into four sequences, each capturing different traffic scenarios and conditions:

- **S1 and S2**: Each sequence is 30 seconds long and contains daytime scenarios of the traffic at the intersection point mentioned above at dusk time.
- **S3**: A 120-second sequence recorded during daytime with sunny conditions.
- **S4**: A 30-second sequence recorded at night with heavy rain.

To facilitate the use and integration of this data, the TUMTraf-I Intersection Dataset provides extrinsic calibration data, allowing a precise mapping between the cameras and LiDARs and accurately projecting 3D labels into camera images. The dataset is structured based on the OpenLABEL standard, where labels and calibration data are stored in *.json* format. Furthermore, a comprehensive TUMTraf-Devkit [Dat24] is available, offering data loading, transformation, splitting, evaluation, and visualization tools.

Key Features and Benefits

The TUMTraf Intersection Dataset offers several key features and benefits that make it an essential resource for research in the field of autonomous driving and a basis for the evaluation and visualization pipeline we will use in this thesis:

- **High-Quality 3D Labels**: The dataset includes 57.4k high-quality manually labeled 3D bounding boxes, ensuring precise and reliable data for training perception algorithms [Zim+23c].
- **Diverse Traffic Scenarios**: The dataset captures many traffic scenarios, including complex maneuvers such as left and right turns, overtaking, and U-turns [Zim+23c], providing a robust foundation for developing robust perception systems.

- **Comprehensive Environmental Coverage**: With 25% of the data recorded at night and in heavy rain, the dataset ensures that algorithms can be trained and tested under various lighting and weather conditions, enhancing their robustness.
- **Synchronized Multi-Sensor Data**: The dataset provides synchronized camera images, LiDAR point clouds, and extrinsic calibration data, enabling accurate data fusion and comprehensive 3D perception.
- **OpenLABEL Standard**: Adhering to the OpenLABEL standard, the dataset ensures compatibility with various data processing and analysis tools, facilitating ease of use and integration into existing workflows.
- **Extensive Devkit**: The TUMTraf-Devkit offers a wide range of tools for data loading, transformation, evaluation, and visualization, streamlining the process of working with the dataset.

2.4 Physical Effect of Adverse Weather Conditions on Point Cloud Data

Adverse weather conditions, such as rain, snow, and fog, significantly impact the quality and reliability of point cloud data collected by LiDAR sensors [Kil+21]. These effects can be attributed to reduced signal-to-noise ratio (SNR) and signal-to-background ratio (SBR) and increased backscattered power from particles in the atmosphere [Kil+21]. This section explores the specific physical effects of each adverse weather condition and explains the underlying light-scattering mechanisms.

2.4.1 Rain

Rain adversely affects LiDAR point clouds through several mechanisms:

1. **Attenuation and Scattering**: As the laser pulse travels through rain, its intensity is exponentially attenuated due to scattering by raindrops. This reduces the returned signal strength, increasing range uncertainty and causing potential mis-detections if the signal falls below the noise floor [Kil+21].
2. **Backscattered Power**: Raindrops, especially those close to the sensor, can scatter significant amounts of laser power back to the sensor, leading to an increase in background noise. This can confuse the detector, resulting in randomly scattered points near the sensor and missing points within the real target [Kil+21].
3. **Loss of Data Points**: Rain can cause the loss of entire data points, particularly those far from the sensor. This phenomenon occurs because the backscattered power from raindrops can overshadow the target's signal, leading to deletions of distant objects in the point cloud data [Kil+21].

2.4.2 Snow

Snow impacts LiDAR data similarly to rain but with some distinct differences:

1. **Increased Scattering**: Snowflakes, being larger and more irregular in shape compared to raindrops, scatter light more effectively, which results in higher attenuation of the laser pulse. This leads to a significant reduction in visibility and increased range uncertainty [Kil+21].

2. **Scattered Points**: Snow's scattering effect is more pronounced, resulting in more scattered points near the sensor. This increased scattering is due to the reflective nature of snowflakes, which can confuse the detector, similar to rain but to a greater extent [Kil+21].

2.4.3 Fog

Fog has a distinct effect on LiDAR data, primarily due to its composition of tiny water droplets:

1. **Dense Scattering Medium**: Fog consists of numerous small droplets that form a dense scattering medium, significantly reducing the intensity of the laser pulse as it travels through the fog. This substantially reduces the signal-to-noise ratio(SNR) and signal-to-background ratio (SBR) [Kil+21], making it difficult for the LiDAR to detect distant objects.

2. **Visibility Reduction**: The dense scattering in fog causes a "fuzzy" effect in the point cloud, degrading the structural fidelity of detected objects. This effect is more pronounced than in snow or rain because the numerous small droplets cause continuous scattering, leading to a significant loss of data points beyond a certain range [Kil+21].

3. **Increased Range Uncertainty**: The presence of fog increases range uncertainty due to the continuous scattering of the laser pulse, making it challenging to accurately determine the distance of objects [Kil+21].

Light scattering in these conditions involves the interaction of laser pulses with particles, causing attenuation and noise in the LiDAR data. In adverse weather conditions, the laser intensity I_T through a scattering medium, for example, a raindrop, can be calculated using the Beer-Lambert law described by the following equation [Kil+21]:

$$I_T = I_0 e^{-\int_0^R \alpha(r)dr} \tag{2.1}$$

where I_0 is the incident intensity, α is the extinction coefficient, and R is the target range. The extinction coefficient α can be calculated using the following equation [Kil+21]:

$$\alpha = \int_0^\infty \sigma_{\text{ext}}(D)N(D)dD \tag{2.2}$$

where σ_{ext} is the extinction cross-section, D is the particle diameter, and $N(D)$ is the particle density function [Kil+21].

Additionally, the range uncertainty σ_R due to small SNR is given in [Kil+21] by:

$$\sigma_R = \frac{\Delta R}{\sqrt{2 \cdot \text{SNR}}} \tag{2.3}$$

where ΔR is the range accuracy determined by the finite bandwidth of the LiDAR system [Kil+21]. The scattering effect on the laser beams is visualized in Figure 2.6.

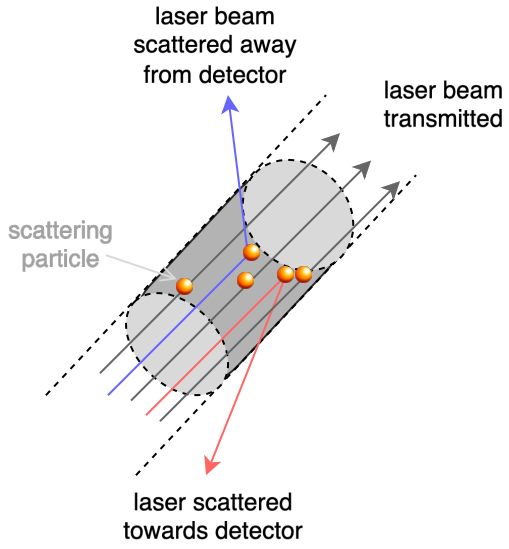


Figure 2.6: Visualization of the scattering effect of rain, snow, and fog particles on the laser pulses.

2.5 Evaluation Metrics

2.5.1 Precision

Precision measures the accuracy of the positive predictions made by the model by calculating the ratio of true positives (TP) to the sum of true positives and false positives (FP):

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.4}$$

Precision ranges from 0 to 1, where 1 indicates that every positive prediction the model makes is a correct detection. High precision is crucial in autonomous driving scenarios since it indicates the detector's false positive rate—a high false positive rate results in many ghost objects that lead to sudden braking and lane changes.

2.5.2 Recall

Recall, also known as sensitivity or true positive rate, is a metric used in 3D detction to measure the model's ability to correctly identify all relevant objects in the dataset. It is defined as the ratio of true positives (TP) to the sum of true positives and false negatives (FN):

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.5}$$

Recall ranges from 0 to 1, where 1 indicates that the model correctly identified all relevant objects.

2.5.3 Average Precision

Average Precision (AP) is a frequently used metric for evaluating the performance of object detection models. This metric measures the precision-recall trade-off by describing the shape

of the Precision-Recall (PR) curve. The PR curve plots the precision against the recall. The AP value represents the area under the curve, capturing the model's ability to balance precision and recall across different thresholds.

Mathematically, AP can be defined as:

$$\text{AP} = \sum_{n} (R_n - R_{n-1})\, P_n \tag{2.6}$$

where P_n and R_n are the precision and recall at the n-th threshold. A higher AP value indicates better performance, with a perfect model achieving an AP of 1.0.

2.5.4 Intersection Over Union: IoU

Intersection Over Union (IoU) is used to evaluate the accuracy of an object detector by comparing the dimensions of a predicted bounding box with its corresponding ground truth bounding box. It is defined by the following equation where B_p is the predicted bounding box and B_{gt} is the ground truth bounding box:

$$\text{IoU} = \frac{|B_p \cap B_{gt}|}{|B_p \cup B_{gt}|} \tag{2.7}$$

The IoU ranges from 0 to 1, where 1 indicates perfect alignment between the predicted and ground truth boxes. A detection is considered a true positive if the IoU exceeds a certain threshold, commonly set at 0.5 or 0.7. High IoU values indicate precise localization of objects by the model, making IoU useful when evaluating whether an object has been detected and how accurately its location has been predicted.

2.5.5 Corruption Error (CE)

Corruption Error (CE), used in [Kon+23], is used to measure the robustness of models against data corruption. It is defined as the relative error between the performance of corrupted data and a baseline model. The CE is calculated as follows:

$$\text{CE} = \frac{\sum_{l=1}^{L}(1 - \text{Acc}_{i,l})}{\sum_{l=1}^{L}(1 - \text{Acc}_{\text{baseline},i,l})} \tag{2.8}$$

where $\text{Acc}_{i,l}$ denotes the accuracy for corruption type i at severity level l.

The mean Corruption Error (mCE) is the average CE over all corruption types:

$$\text{mCE} = \frac{1}{N} \sum_{i=1}^{N} \text{CE}_i \tag{2.9}$$

where N is the total number of corruption types and L is the total number of severity levels.

The values for CE and mCE depend on the range of values for the accuracy metric defined by $\text{Acc}_{i,l}$ and $\text{Acc}_{\text{baseline},i,l}$. Since we will be using precision, recall, AP, and IoU as evaluation metrics, the values for $\text{Acc}_{i,l}$ and $\text{Acc}_{\text{baseline},i,l}$ will be in the range [0,1].

- **Best-case scenario:** If the model's performance is perfect on corrupted data ($\text{Acc}_{i,l} = 1$ for all l), then $1 - \text{Acc}_{i,l} = 1 - 1 = 0$, which results in the lowest mCE value of 0.
- **Worst-case scenario:** If the model's performance is extremely poor on corrupted data compared to the baseline model, then ($\text{Acc}_{i,l} = 0$ for all l), and $1 - \text{Acc}_{i,l} = 1$. Since the baseline model has a better score, then $\text{Acc}_{\text{baseline},i,l} \in (0, 1]$, which results in $1 - \text{Acc}_{\text{baseline},i,l} < 1$ and $mCE > 1$

- **Equal performance to baseline:** If the model's performance on corrupted data matches the baseline ($\text{Acc}_{i,l} = \text{Acc}_{\text{baseline},i,l}$ for all l), then CE would be 1.

Given these considerations, the range of the Corruption Error (CE) is:

$$\text{CE} \in (0, \infty)$$

For the mean Corruption Error (mCE), being an average of the CE values across different corruption types, its range is also:

$$\text{mCE} \in (0, \infty)$$

2.5.6 Resilience Rate (RR)

Resilience Rate (RR), also used in [Kon+23], measures the model's robustness by comparing its accuracy on corrupted data to its accuracy on clean data. The RR is calculated as follows:

$$\text{RR} = \frac{\sum_{l=1}^{L} \text{Acc}_{i,l}}{L \times \text{Acc}_{\text{clean}}} \tag{2.10}$$

where $\text{Acc}_{i,l}$ denotes the accuracy for corruption type i at severity level l and $\text{Acc}_{\text{clean}}$ denotes the accuracy on clean data [Kon+23].

The mean Resilience Rate (mRR) is the average RR over all corruption types:

$$\text{mRR} = \frac{1}{N} \sum_{i=1}^{N} \text{RR}_i \tag{2.11}$$

where N is the total number of corruption types and L is the total number of severity levels.

High RR values indicate better robustness, showing the model maintains high performance even under corrupted conditions.

The values for RR and mRR depend on the accuracies $\text{Acc}_{i,l}$ and $\text{Acc}_{\text{clean}}$, which range from 0 to 1, as mentioned above.

- **Best-case scenario:** If the model's performance is perfect on both corrupted and clean data ($\text{Acc}_{i,l} = 1$ for all l and $\text{Acc}_{\text{clean}} = 1$), then RR would be:

$$\text{RR} = \frac{\sum_{l=1}^{L} 1}{L \times 1} = \frac{L}{L} = 1$$

- **Worst-case scenario:** If the model's performance is extremely poor on corrupted data ($\text{Acc}_{i,l} = 0$ for all l) and it performs well on clean data ($\text{Acc}_{\text{clean}} > 0$), then RR would be:

$$\text{RR} = \frac{\sum_{l=1}^{L} 0}{L \times \text{Acc}_{\text{clean}}} = 0$$

- **Higher than clean performance:** If the model's performance on corrupted data is higher than on clean data (which is unusual but possible in certain scenarios), then RR would be greater than 1.

Given these considerations, the range of the Resilience Rate (RR) is:

$$\text{RR} \in [0, \infty)$$

For the mean Resilience Rate (mRR), being an average of the RR values across different corruption types, its range is also:

$$\text{mRR} \in [0, \infty)$$

2.6 Similarity Calculation Algorithm: ICP

The Iterative Closest Point (ICP) algorithm is a key method for aligning 3D surfaces, widely used in tasks like Simultaneous Localization and Mapping (SLAM) [Bai22]. ICP iteratively refines the transformation needed to minimize the distance between two point clouds as described in Algorithm 1. In SLAM, ICP aligns point clouds from sensors like LiDAR and RGB-D cameras. It aids in pose estimation and loop closure by ensuring accurate alignment of frames or revisited locations [Bai22]. ICP can also compare two point clouds to check their similarity. Applying the ICP algorithm determines the transformation that best aligns the two point clouds. The resulting alignment error provides a quantitative measure of their similarity, where a smaller error indicates a higher degree of similarity between the point clouds.

Algorithm 1 Basic ICP Algorithm [Bai22]

Input: Source point cloud $S = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$, Destination point cloud $D = \{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_m\}$, Convergence threshold θ.
Output: Estimated transformation (rotation $\mathbf{R}$ and translation $\mathbf{t}$).
Initialize $\mathbf{R}$ and $\mathbf{t}$ (e.g., identity matrix and zero vector).
while not converged **do**
    for each $\mathbf{s}_i \in S$ **do**
        Find the closest point $\mathbf{d}_j \in D$.
    end for
    Compute the optimal transformation $(\mathbf{R}, \mathbf{t})$ to minimize the error:

$$\min_{\mathbf{R},\mathbf{t}} \sum_i \|\mathbf{d}_j - (\mathbf{R}\mathbf{s}_i + \mathbf{t})\|^2$$

    Apply the transformation: $S \leftarrow \{\mathbf{R}\mathbf{s}_i + \mathbf{t} \,|\, \mathbf{s}_i \in S\}$.
    Check for convergence: if the change in error is less than θ, break.
end while
return $\mathbf{R}$ and $\mathbf{t}$.

Chapter 3

Related Work

In this section, we conduct a literature review of the relevant existing work related to the thesis's topic. We summarize the results of state-of-the-art approaches and discuss their limitations. We first review existing papers and journal publications highlighting adverse weather conditions' effect on lidar 3D perception. Then, we review the most relevant models and approaches designed to mitigate the adverse weather effects.

3.1 Impact of Adverse Weather Conditions on LiDAR Perception

Adverse weather conditions such as rain, snow, fog, and road spray significantly affect the performance of LiDAR sensors used for 3D perception in autonomous driving scenarios. This section reviews recent research on the degradation of LiDAR performance under these conditions and the methods proposed to mitigate these effects.

The authors of [Kil+21] test the impact of adverse weather conditions like rain, snow and fog on state-of-the-art 3D detection models such as PointPillars [Lan+19], SECOND [YML18], and PV-RCNN [Shi+19], and compute the BEV mAP and the 3D mAP of these models using the augmented KITTI [GLU12] dataset. The results of their experiments are highlighted in Table 3.1. The results show a substantial drop in 3D detection performance for all models under all adverse weather conditions, with strong fog conditions being the most corrupting for these models. For instance, all models show a drop of approximately 40-45% in 3D mAP when tested with snow and moderate fog augmented data.

[Kon+23] also introduces Robo3D, a comprehensive benchmark designed to evaluate the robustness of 3D perception systems under various corruptions encountered in real-world environments. Their benchmark includes eight corruption types divided into three categories: severe weather conditions (fog, rain, snow), external disturbances (motion blur, beam missing), and internal sensor failures (crosstalk, incomplete echo, cross sensor). The Robo3D benchmark was tested on various state-of-the-art 3D perception models using datasets like KITTI [GLU12], SemanticKITTI [Beh+19], nuScenes [Cae+20], and Waymo Open Dataset [Sun+20]. The evaluation metrics included mean Corruption Error (mCE) and Resilience Rate (mRR), both explained in Chapter 2. Higher values of mCE indicate a higher sensitivity to corruption, while lower values indicate a higher resilience of the models. The experiments revealed that current models are significantly affected by the introduced corruption, as shown in Table 3.2. The results in Table 3.2 show very high mCE values overall with models like Part-A2 [Shi+21], and PV-RCNN [Shi+19] more affected by fog conditions, which aligns with the results from [Kil+21] illustrated in Table 3.1. The Part-A2 [Shi+21] model consistently performs better than the other model under all adverse conditions.

Table 3.1: Comparison of Mean Average Precision (mAP) Values for "Car" Class on the KITTI Dataset under Simulated Fog and Snow Conditions according to [Kil+21]. The values that show the highest impact are underlined.

Network	Simulation	BEV mAP	3D mAP
PointPillars	Clear	89.98	83.08
	Snow	63.05	43.96
	Moderate Fog	55.10	43.12
	Strong Fog	**45.24**	**33.35**
SECOND	Clear	90.25	84.12
	Snow	64.12	45.28
	Moderate Fog	57.68	46.33
	Strong Fog	**48.54**	**37.59**
PV-RCNN	Clear	91.73	85.60
	Snow	65.40	47.11
	Moderate Fog	59.44	48.56
	Strong Fog	**49.62**	**38.29**

Table 3.2: The Corruption Error (CE) of 7 detectors on KITTI-C. Bold: Best in column, Red: Worst in row. [Kon+23]

Method	**mCE ↓**	**Fog**	**Wet**	**Snow**
SECOND	95.9	99.7	100.6	87.6
PointPillars	110.7	115.8	106.4	124.9
PartA2-F	**82.2**	**89.4**	**75.8**	**81.3**
PV-RCNN	90.0	95.2	86.6	93.1

3.1.1 Rain

Rain is known to degrade LiDAR sensor performance by introducing noise and reducing the density of the point cloud. According to [Zha+21], rain affects the LiDAR measurements by causing the laser pulses to scatter, leading to a sparser and noisier point cloud compared to clear conditions. Their study proposed a novel anomaly detection method to quantify LiDAR degradation in rainy conditions. Their proposed model learns the distribution of regular LiDAR scans (collected in clear weather) and calculates a degradation score for new scans by measuring their deviation from this distribution. A higher degradation score indicates more significant LiDAR performance degradation due to rain. This method effectively captures the impact of rain on LiDAR by comparing scans from normal and rainy conditions, providing a quantitative measure of degradation [Zha+21]. While they did not report on the average number of points per frame in the dataset they used, they found that rain can reduce the number of LiDAR points per scan by up to 20k points and reduce the detection range by up to 33% (from 60 m to 40 m). The LiDAR scan in Figure 3.1 from rainy weather (right) was sparser and noisier with a lower average intensity than scans taken in clear weather conditions (left).

[Teu+22] also investigates the effect of rain on the 3D detection of the PointPillars model [Lan+19] and reports a decrease of at least 10% in the AP performance of the car, pedestrian, and cyclist classes from the KITTI dataset as can be seen in Figure 3 from [Teu+22].

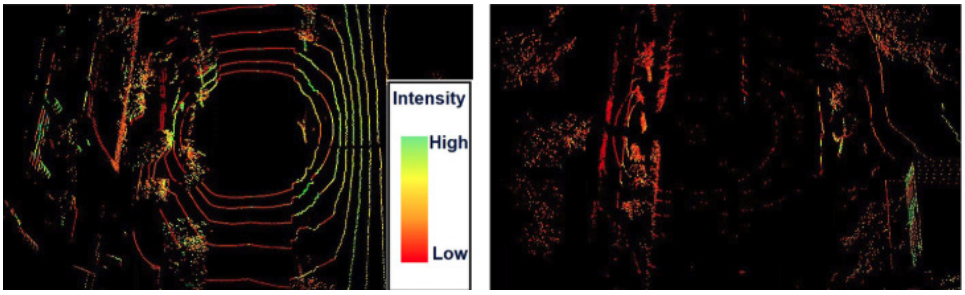


Figure 3.1: Comparison of LiDAR scans in normal weather (left) and rainy weather (right). Brighter color corresponds to higher intensity values [Zha+21].

3.1.2 Snow

Snow introduces significant challenges for LiDAR perception due to its reflective properties. The study by [Teu+22] indicates that snowflakes create unwanted noise in LiDAR measurements, similar to rain but often more intense due to snowflakes' size and reflective nature. Snow can lead to false positives and missed detections as the LiDAR sensor incorrectly identifies snowflakes as objects. In their experiments, they evaluated the 3D detection results of the PointPillars model [Lan+19] on the KITTI dataset using augmented point clouds with several weather condition effects. The results of their experiments show that false positive detections increased significantly in snowy conditions [Teu+22]. While the pedestrian and cyclist classes consistently perform steadily in snowy conditions, the car class shows a 42% drop in AP measures, and the mAP of the model drops from 48% in clear weather to almost 20% in snowy conditions (See Figure 3 in [Teu+22]). The authors of [Lin+22] also conduct a comparative study of the KITTI [GLU12] dataset, containing mostly clear weather point clouds, and the Canadian Adverse Driving Conditions (CADC) dataset [Pit+20], containing point clouds captured in snowy weather conditions. They report that the point clouds generated in snowy conditions are significantly sparser than the ones generated in clear conditions, with an increased noise resulting from the snowflakes' occlusion of the sensor's visibility range [Lin+22].

3.1.3 Fog

[Teu+22] found that fog significantly impacts the detection range and density of LiDAR point clouds. The number of droplets per unit volume in fog is much higher than in rain, leading to severe attenuation of the laser pulses. Their experiments showed that the mean average precision of the most relevant object classes in the KITTI dataset, such as cars, pedestrians, and cyclists, drops significantly, going down to as low as 0.0% for pedestrians and cyclists in very dense fog conditions, resulting in visibility ranges as low as 50 m. The car class shows a slightly better resistance to the fog yet still scores an AP of 20% in dense fog conditions (See Figure 3 in [Teu+22]).

3.2 State-of-the-Art Approaches to Mitigate the Weather's Impact on LiDAR

Recent advancements in LiDAR technology have focused on enhancing its robustness against adverse weather conditions by developing sophisticated computational techniques such as enhanced machine learning models, sensor fusion strategies, and adaptive system designs.

3.2.1 Denoising Point Clouds

[SOT22] introduces a novel deep-learning algorithm to address the noise caused by adverse weather conditions in LiDAR sensor data, significantly degrading the quality of point clouds used in autonomous driving applications. The approach removes points from LiDAR point clouds caused by airborne particles such as rain, fog, and snow. This noise leads to displaced and missing points, hindering tasks like mapping, localization, and object detection. The method involves using adjacent point clouds captured at different times to combine spatial and temporal data that is later fed into the neural network to predict and remove the noise caused by airborne particles.

Neural Network Architecture (4DenoiseNet)

The architecture consists of two branches: spatial and temporal. The *spatial branch* processes the current point cloud's spatial features, capturing the scene's local geometric structure. On the other hand, the *temporal branch* processes the temporal features by comparing the current point cloud with the previous one, helping to differentiate between static objects and moving airborne particles [SOT22]. Both branches use k-nearest neighbors (kNN) convolution to effectively capture local spatial information and motion-guided attention mechanisms to fuse these features [SOT22].

Training Methodology

The network is trained using semi-synthetic data, where adverse weather effects are simulated and added to real point clouds captured in clear weather. This approach creates a large and diverse training set with accurate labels. The training objective includes a combination of cross-entropy and Lovász-Softmax loss to optimize for the Jaccard index, which measures the similarity between the predicted and actual data [SOT22].

Performance and Evaluation

The method is tested on the SnowyKITTI and Canadian Adverse Driving Conditions datasets. The results of the segmentation task show that 4DenoiseNet outperforms previous methods and generalizes well to various adverse weather conditions, reaching an IoU of 0.977 in heavy snow conditions, which is 10% higher than the WeatherNet CNN-based denoising model used in [Hei+20] (0.865 IoU in heavy snow conditions according to [SOT22]). The authors [WL24] propose integrating the 4DenoiseNet model as a data preprocessing step in the 3D detection pipeline to reduce the noise in the dataset resulting from adverse weather conditions. This pipeline extension, however, results in an increased inference time, which faces a challenge to the real-time prediction constraint in the autonomous driving scenario. The authors of [Wan+24] propose using simpler denoising techniques like Bayesian estimation, voxel filtering, and neighborhood filtering to clean noisy point cloud data.

3.2.2 Energy-based Detection of Adverse Weather Conditions

The authors of [Pir+23a] and [Liu+20] propose an energy-based approach for detecting and filtering the point clouds corrupted by adverse weather conditions. Both studies use an energy score that predicts for each point in the point cloud, whether it is a real point reflected from a surrounding object or a result of weather corruption. The [Pir+23a] and [Liu+20] add an outlier removal step to the detection pipeline, such that points with a high energy score are

considered outliers and removed from the point cloud. [Liu+20] have generated a labeled version SemanticSpray [Pir+23b] of the unlabeled RoadSpray dataset [Lin+04] dataset and used the labeled dataset with the energy-based segmentation approach. The authors of [Liu+20] report an increase of up to 18.89% in the area under receiver operating characteristics (AUROC) results when using their model (AWNet), compared to previous point classification models like Particle-UNet [Sta+21], Particle-VoxelNet [Sta+21], and WeatherNet [Hei+20].

3.2.3 Noise Robustness Loss

[Pir+22] addresses the impact of vehicle gas exhaust condensation on LiDAR-based object detectors in cold weather conditions. The authors demonstrate how this corruption compromises the reliability of object detectors by distorting object size and orientation estimates and introducing ghost object detections. Besides, they propose a solution involving data augmentation and a novel training loss term to enhance robustness against such disturbances [Pir+22].

Methodology

Gas Exhaust Data Generation The authors of [Pir+22] developed a method to generate gas exhaust data using 3D surface reconstruction and sampling. This technique allows the creation of large datasets of gas exhaust clouds using a small set of labeled data as input [Pir+22]. The generated gas exhaust data augment point clouds recorded in good weather conditions. This avoids the need for new data collection under adverse conditions.

Noise Robustness Loss [Pir+22] proposes a novel noise robustness loss term to address the noise caused by gas exhaust in LiDAR data. The noise robustness loss L_{noise} penalizes the inclusion of noise points within the predicted bounding boxes [Pir+22]. The noise robustness loss term L_{noise} is defined in [Pir+22] as follows:

$$L_{\text{noise}} = \text{IoU}_{3D}(P, B) \tag{3.1}$$

Where $\text{IoU}_{3D}(P, B)$ represents the 3D IoU loss between the gas exhaust bounding boxes B and the model's predictions P as defined in [Pir+22].

The total training loss L is then formulated by combining the baseline training loss L_{train} with the noise robustness loss, weighted by a parameter β:

$$L = L_{\text{train}} + \beta L_{\text{noise}} \tag{3.2}$$

Where L_{train} is the baseline object detector training loss, and $\beta \in \mathbb{R}$ is a weight parameter that adjusts the influence of the noise robustness loss [Pir+22]. Adjusting the parameter β can control the balance between the baseline training loss and the noise robustness loss.

Experiments and Results The method was tested on two popular object detectors, SECOND [YML18] and PointRCNN [SWL19], using the DENSE's dataset for fog simulation (SeeingThrough Fog dataset [Bij+20]), which contains diverse weather conditions. The results demonstrated that the proposed method improves robustness to gas exhaust and noisy data without negatively impacting the performance in clean conditions. When tested using the Gas exhaust augmented DENSE test set with easy, moderate, and heavy noise augmentations, both SECOND and PointRCNN augmented models show an improvement of performance of 2.5% and 1.14% in 3D AP at 0.7 IoU threshold. These results show that the proposed framework effectively increases the robustness of 3D object detection against vehicle gas exhaust in cold weather.

3.2.4 Densification Methods

[Wan+22] presents a novel framework designed to enhance 3D object detection by learning to densify sparse point clouds in the latent space. The approach involves training two types of 3D detectors: a dense point 3D detector (DDet) with dense point clouds and a sparse point 3D detector (SDet) with regular sparse point clouds. A lightweight S2D module and a point cloud reconstruction (PCR) module are introduced to facilitate the densification process, improving detection performance on sparse data inputs without requiring dense data during inference [Wan+22].

Methodology

Dense Point 3D Detector (DDet) DDet is trained with dense point clouds obtained by fusing multi-frame point clouds. It utilizes a region proposal network (RPN) and multiple heads for object classification and regression [Wan+22].

Sparse Point 3D Detector (SDet) SDet is trained with regular sparse point clouds. It uses DDet to teach SDet to simulate densified 3D features, enabling SDet to generate high-quality 3D features from sparse inputs. The model uses two modules, S2D and PCR, to enhance feature learning further. S2D projects sparse 3D features to BEV space uses 2D convolution operations and employs ConvNeXt blocks for feature aggregation [Wan+22]. Features are then upsampled and concatenated to obtain the final densified feature. PCR reconstructs voxel-level dense object point clouds from the densified features. It predicts a soft voxel-occupancy mask and point offset for each non-empty voxel [Wan+22].

Experiments and Results

The framework is evaluated on the Waymo Open Dataset and Waymo Domain Adaptation Dataset [Sun+20]. Significant improvements in detection performance are observed across all categories and evaluation metrics. The approach consistently outperforms state-of-the-art methods such as baseline SECOND [YML18], PointPillars [Lan+19], and CenterPoint [YZK21] by respectively 3.01%, 4.93%, 3.15% in mAP results.

3.2.5 Data Augmentation Methods

Fog Simulation

[Hah+21] addresses the challenge of LiDAR-based 3D object detection in foggy weather. Collecting and annotating data in such conditions is labor-intensive and costly. To address this, the authors propose a physically accurate fog simulation method that can be applied to any LiDAR dataset, enabling large-scale foggy training data to be created at a minor extra cost. The simulated foggy data improves the robustness of 3D object detection and other perception tasks. The method significantly enhances performance on real foggy data, as demonstrated through experiments with state-of-the-art detection approaches on the Seeing Through Fog dataset [Bij+20]. The authors of [Zha+24a] report using a similar physics-based approach to simulate fog conditions in marine 3D detection scenarios. The results of the experiments in [Zha+24a] show an improvement of $\approx 40\%$ compared to the baseline PointPillars model in the detection of Cargo ships.

Methodology The simulation uses a physically based model to convert clear-weather LiDAR point clouds into foggy ones. The method models the transmission of LiDAR pulses, considering

the system's impulse response in clear weather and fog. By transforming the range and intensity of each original clear-weather point, the new values correspond to measurements in a foggy scenario [Hah+21].

The received signal power P_R in clear weather is given in [Hah+21] by:

$$P_R(R) = C_A \int_0^{2R/c} P_T(t) H\left(R - \frac{ct}{2}\right) dt$$

where R is the emitted signal, C_A is a system constant, $P_T(t)$ is the transmitted signal power, and H is the impulse response of the environment.

In fog, the total one-way transmission loss $T(R)$ is given by:

$$T(R) = \exp(-\alpha R)$$

where α is the attenuation coefficient.

The simulation is applied to the clear-weather training set of the Seeing Through Fog (STF) dataset [Bij+20], which was captured using the Velodyne HDL-64E LiDAR sensor. The fog simulation involves parameters like the attenuation and backscattering coefficients, which are empirically set to match real foggy point clouds.

Experiments The fog simulation method is evaluated using state-of-the-art 3D object detection methods (PV-RCNN [Shi+19], PointRCNN [SWL19], SECOND [YML18], Part-A2 [Shi+21], and PointPillars [Lan+19]) trained on the STF dataset. The results in [Hah+21] show that training with simulated foggy data significantly improves detection performance in real foggy conditions compared to clear weather models and previous fog simulation methods.

Results Quantitative results show that models trained with the proposed fog simulation outperform those trained with clear weather data and previous simulation methods across various evaluation metrics. The results are illustrated in Table 3.3, which shows an improvement of the 3D detection of the car class by at least 1.2% for all models under easy fog conditions and by 0.5% under hard fog conditions.

Table 3.3: Car 3D AP@.5IoU results on all relevant test splits of the Seeing Through Fog dataset [Bij+20] with easy, moderate, and hard fog conditions [Hah+21].

Method	easy	moderate	hard
PV-RCNN (Clear)	64.73	64.41	61.52
PV-RCNN (w/ Fog Simulation)	65.79	65.03	65.03
PointRCNN (Clear)	65.12	64.34	60.95
PointRCNN (w/ Fog Simulation)	66.32	65.14	61.34
SECOND (Clear)	63.85	62.75	60.42
SECOND (w/ Fog Simulation)	64.15	63.20	61.21
Part-A^2 (Clear)	59.76	59.27	56.88
Part-A^2 (w/ Fog Simulation)	62.31	61.46	58.32
PointPillars (Clear)	59.81	59.45	57.12
PointPillars (w/ Fog Simulation)	61.34	60.88	58.55

Rain and Snow simulation

[Kil+21] presents a novel approach called LiDAR Light Scattering Augmentation (LISA) to simulate the impact of adverse weather conditions such as rain and snow on LiDAR data for improving 3D object detection in autonomous navigation systems. By leveraging physical light scattering models, the authors propose LISA to simulate these conditions accurately, providing a diverse dataset for training and testing 3D object detectors.

Methodology LISA uses physical light scattering models to simulate adverse weather effects on LiDAR data. The framework includes models for rain and snow as follows:

Rain Simulation The described approach for rain simulation uses a noise filter of spherical raindrops, with sizes determined by a raindrop size distribution. Beam divergence is obtained by generating multiple rays per point in the point cloud at a specified angle, where points are modified if enough rays intersect with the original point. The point cloud is modified by setting scan points closer to the sensor or deleting them based on intersection ratios [Teu+22; Kil+21]:

$$R_{\text{most}} = \frac{N_{\text{most}}}{N_{\text{intersects}}} \tag{3.3}$$

where N_{most} is the number of rays intersecting the raindrop with the most intersections, and $N_{\text{intersects}}$ is the total number of intersections [Kil+21].

Snow Simulation Based on ray tracing, using the size distribution for snowflakes from the Gunn and Marshall distribution:

$$N(D) = N_0 e^{-\Lambda D} \tag{3.4}$$

where N_0 and Λ are parameters dependent on the precipitation rate R. The intensity of false points due to snowflakes is modeled with a lognormal distribution.

Experiments and Results The models are parameterized using the DENSE [Bij+20] dataset and evaluated on the KITTI [GLU12] and Waymo [Sun+20] Open datasets. To compare the performance of state-of-the-art models like SECOND [YML18], PART-A2 [Shi+21], and PV-RCNN [Shi+19], the authors of [Kil+21] compute the mAP of the class "Vehicle" from the Waymo dataset [Sun+20]. As can be seen in Table 3.4, the use of an augmented dataset using the LISA method resulted in an improvement in the performance of the SECOND [YML18], PART-A2 [Shi+21], and PV-RCNN [Shi+19] models by respectively 5.78%, 2.29%, and 2.77%

3.2.6 Sequence Concatenation in Time

[Kem+23a] introduces another way to enhance the robustness of LiDAR-based object detection under adverse weather conditions by using time-series data.

Methodology [Kem+23a] presents an architecture based on Pillar-based Object Detection, allowing the use of temporal information from sequence data. The authors of [Kem+23a] conduct a comprehensive study of different approaches for using data sequences in the model architecture and provide a quantitative evaluation and comparison of the trained models on three real-world adverse weather datasets. The use of temporal information within a sequence of LiDAR point clouds consists of concatenating n point clouds and using it as input to the detection model [Kem+23a]. The authors refer to the point cloud concatenation approach as IC.

Table 3.4: A comparison of mAP values of networks retrained on simulated rainy LiDAR poitn clouds using LISA and evaluated on real rainy scenes from the Waymo dataset [Sun+20] for the class "Vehicle", based on the results from [Kil+21].

Network	Training Dataset	mAP
SECOND	Clear	39.69
	LISA	**45.47**
		+5.78
Part A^2	Clear	48.29
	LISA	**50.58**
		+2.29
PV-RCNN	Clear	41.98
	LISA	**44.74**
		+2.77

Point Concatenation:

$$P_{\text{IC}} = \{p_1, \ldots, p_m, p_{m+1}, \ldots, p_n\} \tag{3.5}$$

where $p = (x, y, z, \text{intensity})$.

Results and Discussion As shown in Table 3.5, the evaluation results of the PointPillars-based model with and without IC on the NuScenes dataset [Cae+20] show a decrease of performance by 24% in mAP at the IoU threshold of 0.5 with a 10 ms increase in inference time, resulting in a 0.109s/iteration. The study found that while early-stage input concatenation without coordinate transformation between point clouds was less effective, feature concatenation with temporal offset yielded the best results.

Table 3.5: Evaluation results for IC with and without Temporal Offset, according to [Kem+23a]. Inference time in seconds per input sequence. (*) indicates use of temporal offset.

Method	mAP@IoU=0.5	mAP@IoU=0.75	Inference Time (s)	Dataset
PointPillars	0.510	0.246	0.099	NuScenes
PointPillars + IC	0.387	0.144	0.109	NuScenes

3.2.7 Summary

Table 3.6 summarizes the results obtained in all the solutions explained above. It shows the adverse weather's impact on the performance of the state-of-the-art models and the performance improvement using the several techniques used in these solutions. From the results stated above, we can see that the approach in [Kil+21] has a high coverage of adverse weather conditions and the highest number of tested models with reported improvement of results. While the Sparse2Dense model used in [Wan+22] shows promising progress in the AP value of the PointPillars model and the SECOND [YML18] model, its effectiveness has not been proven for adverse weather conditions specifically.

Table 3.6: Summary of methods and key results. mAP Drop: Drop in performance when tested under adverse weather conditions. mCE: mean Corruption Error. AP Improvement: Improved average precision when trained with augmented data compared to the baseline model. Red: Densification. Blue: Augmentation. Green: Noise Robustness Loss.

Model	mAP Drop	mCE	AP Improvement
PointPillars	40% in moderate fog and snow [Kil+21] 42% in snow [Teu+22]	110.7 [Kon+23]	4.93% [Wan+22]
SECOND	40% in moderate fog and snow [Kil+21]	95.9 [Kon+23]	2.50% [Pir+22] 3.01% [Wan+22] 5.78% [Kil+21]
PV-RCNN	40% in moderate fog and snow [Kil+21]	90.0 [Kon+23]	2.76% [Kil+21]
Part-A2	-	82.2 [Kon+23]	2.29% [Kil+21]
PointRCNN	-	91.9 [Kon+23]	1.14% [Pir+22]

Chapter 4

Methodology

This thesis is part of the AUTOtech.agil project. Thus, we re-use the baseline PointPillars model used for 3D detection in [Zim+23a; Ngu23]. Based on the literature review conducted in Chapter 3 and the observed increase of performance for the SECOND [YML18], PV-RCNN [Shi+19], and Part-A2 [Shi+21] models when using data augmentation, we have decided to use a similar approach on the implemented baseline PointPillars model. This technique has been shown by studies like [Kil+21], [Teu+22], and [Hah+21] to enhance the model's ability to generalize across different conditions and improve its robustness in real-world applications. In another experiment, we test the approach of backward concatenation with offset used in [Kem+23a] and report our results on the TUMTraffic Intersection (TUMTraf-I) dataset [Zim+23c]. The experiments and results based on these approaches are further presented in Chapter 5, where we demonstrate the improvements in the model's performance and robustness achieved through these techniques.

4.1 Data Augmentation

The original TUMTraf-I [Zim+23c] dataset used for training initially contains only one type of adverse weather condition, rain, comprising only 25% of the dataset. We use data augmentation techniques to address this limitation and simulate additional adverse weather conditions, including rain, snow, and fog. The approach in [Kil+21] is designed to work for Velodyne HDL-64E LiDAR and to be used on the KITTI dataset [GLU12]. We extend the LISA library in https://github.com/MartinHahner/LISA/ to support the LiDAR sensor used to capture the point clouds in the TUMTraf-I dataset, i.e. the Ouster OS1-64 (generation 2) LiDAR. We then perform augmentation on the training, validation, and test sets and run a series of experiments to track the effect of data augmentation on the performance of the PointPillars model. Figure 4.1 represents the augmentation pipeline and the experiments we run to determine the most optimal augmentation level.

4.1.1 Algorithm Description

Algorithm 2 outlined below describes the steps used for dataset augmentation. It begins by generating rain rates and processing each point cloud in the dataset. For each point cloud, the algorithm checks whether it contains sufficient data points (set to 70000 after running a few trials with several thresholds). The algorithm simulates fog, rain, and snow conditions if these conditions are met. In order to simulate rain and snow similar to realistic conditions, we draw rain rates from the Marshall-Palmer model as described in [Kil+21] and a snow rate of 71 mm/h, representing a realistic snow intensity. We only save the augmented point cloud if a

certain similarity and ratio of the original-to-augmented number of points is reached. We use the ICP similarity method explained in Chapter 2 to calculate the similarity between the original and the augmented point clouds. Based on a trial and error approach, we set the threshold values for similarity and original-to-augmented number of points to (0.75, 0.8), (0.75, 0.6), and (0.7, 0.6), respectively, for fog, rain, and snow. We added the similarity check between the original and the augmented point clouds to avoid injecting random noise into the dataset and to keep the results of our experiments comparable to the baseline results from [Zim+23a].

Weather-Specific Simulation Methods

Rain Rain is simulated by mimicking light scattering of the points in the point clouds when facing rain droplets. We draw three samples of rain rates from an exponential distribution at rain rate $\lambda = 0.05\ mm/h$ (as set by [Kil+21]), and we perform augmentation on each day scene with a total number of points exceeding 70000 using each sampled rain rate. To align the implementation with the configuration of the Ouster LiDAR used in the TUMTraf-I dataset, we initialize the `LISA` class with different minimum and maximum range, beam divergence, and range accuracy as in [Kil+21]. We set the range to [0.5, 200], the beam divergence to 0.002269, and the range accuracy to 0.09. We use the mode `rain` for the rain simulation and keep the same rain droplet diameter of 50μm with `signal` set to `strongest` to return the particles (real point or scattering point) with the strongest power.

Listing 4.1: Rain simulation using LISA library

```
def add_light_scatter(self, pc, Rr):
    self.lisa = LISA(wavelength=850, mode="rain", r_min=0.5, r_max=200,
                     beam_divergence=0.002269, min_diameter=50e-6,
                     range_accuracy=0.09, signal="strongest",
                         show_progressbar=True)
    return self.lisa.augment(pc=pc, Rr=Rr, fixed_seed=True)
```

This choice of sampling is based on the nature of rain rate distributions in real-world scenarios, as described in the LISA library paper [Kil+21]. The exponential distribution effectively models the variability and intensity of rain, ensuring the augmented data reflects realistic weather conditions [Kil+21]. The rain model described by the number of particles and their size distribution as functions of rain rate (Rr) is set following the Marshall Palmer model as explained in [Kil+21], and the refractive index of rain droplets is set to 1.328.

Snow Snow simulation is very similar to the rain simulation as snowflakes can be regarded as rain droplets with much higher density. Therefore, we use the gunn mode from the `LISA` class to simulate ice and snow effects. The gunn mode sets the refractive index of the scattering points to 1.3031 and uses the Marshall-Gunn snow model to determine the number of snowflakes and their size distribution. We use the same configuration for minimum and maximum range values, beam divergence, and range accuracy as for rain simulation.

Listing 4.2: Snow simulation using LISA library

```
def add_snow(self, pc, Rr):
    self.lisa = LISA(wavelength=850, mode="gunn", r_min=0.5, r_max=200,
                     beam_divergence=0.002269, min_diameter=50e-6,
                     range_accuracy=0.09, signal="strongest",
                         show_progressbar=True)
    return self.lisa.augment(pc=pc, Rr=Rr, fixed_seed=True)
```

Algorithm 2 Point Cloud Augmentation

Require: Dataset of point clouds D, maximum point clouds M, processed point clouds count P, adverse weather conditions W
Ensure: Processed point clouds with augmented data

function PROCESS_POINT_CLOUDS(D, M, P, W)
  for each $pc, input\_file\_path\_point\_cloud$ in D **do**
    if $M \neq$ None and $P \geq M$ **then**
      break
    end if
    $labels \leftarrow load\_labels(path\_label)$
    $original\_pc \leftarrow prepare\_original\_pc(pc)$
    if not $check\_night()$ and $len(original\_pc) > 70000$ **then**
      for each $weather$ in W **do**
        if not $is\_already\_processed(weather, file\_name\_point\_cloud)$ **then**
          $augmented\_pcs, augmented\_labels \leftarrow$ process_augmentation($weather, original\_pc, file$
        else
          $P += 1$
        end if
      end for
    end if
  end for
  update_similarity_metrics()
end function
function PROCESS_AUGMENTATION(adverse_weather, original_pc, file_name_point_cloud, path_label)
  if $adverse\_weather ==$ fog **then**
    $augmented\_pc\_array \leftarrow$ add_fog_effect($original\_pc$)
    $similarity \leftarrow calculate\_similarity(augmented\_pc\_array, original\_pc, method = icp)$
    if $len(augmented\_pc\_array) \geq 0.8 * len(original\_pc)$ and $similarity \geq 0.75$ **then**
      write_point_cloud_with_intensities(output_path, augmented_pc_array)
    end if
  else if $adverse\_weather ==$ rain **then**
    for each $rain\_rate$ in $rain\_rates$ **do**
      $augmented\_pc\_array \leftarrow$ add_light_scatter($original\_pc, rain\_rate$)
      $similarity \leftarrow calculate\_similarity(augmented\_pc\_array, original\_pc, method = icp)$
      if $similarity \geq 0.75$ and $len(augmented\_pc\_array)/len(original\_pc) > 0.6$ **then**
        write_point_cloud_with_intensities(output_path, augmented_pc_array)
      end if
    end for
  else if $adverse\_weather ==$ snow **then**
    $augmented\_pc\_array \leftarrow$ add_snow($original\_pc, rate = 71$)
    $similarity \leftarrow calculate\_similarity(augmented\_pc\_array, original\_pc, method = icp)$
    if $similarity \geq 0.7$ and $len(augmented\_pc\_array) \geq 0.6 * len(original\_pc)$ **then**
      write_point_cloud_with_intensities(output_path, augmented_pc_array)
    end if
  end if
  return augmented_pc_arrays, weather_label
end function

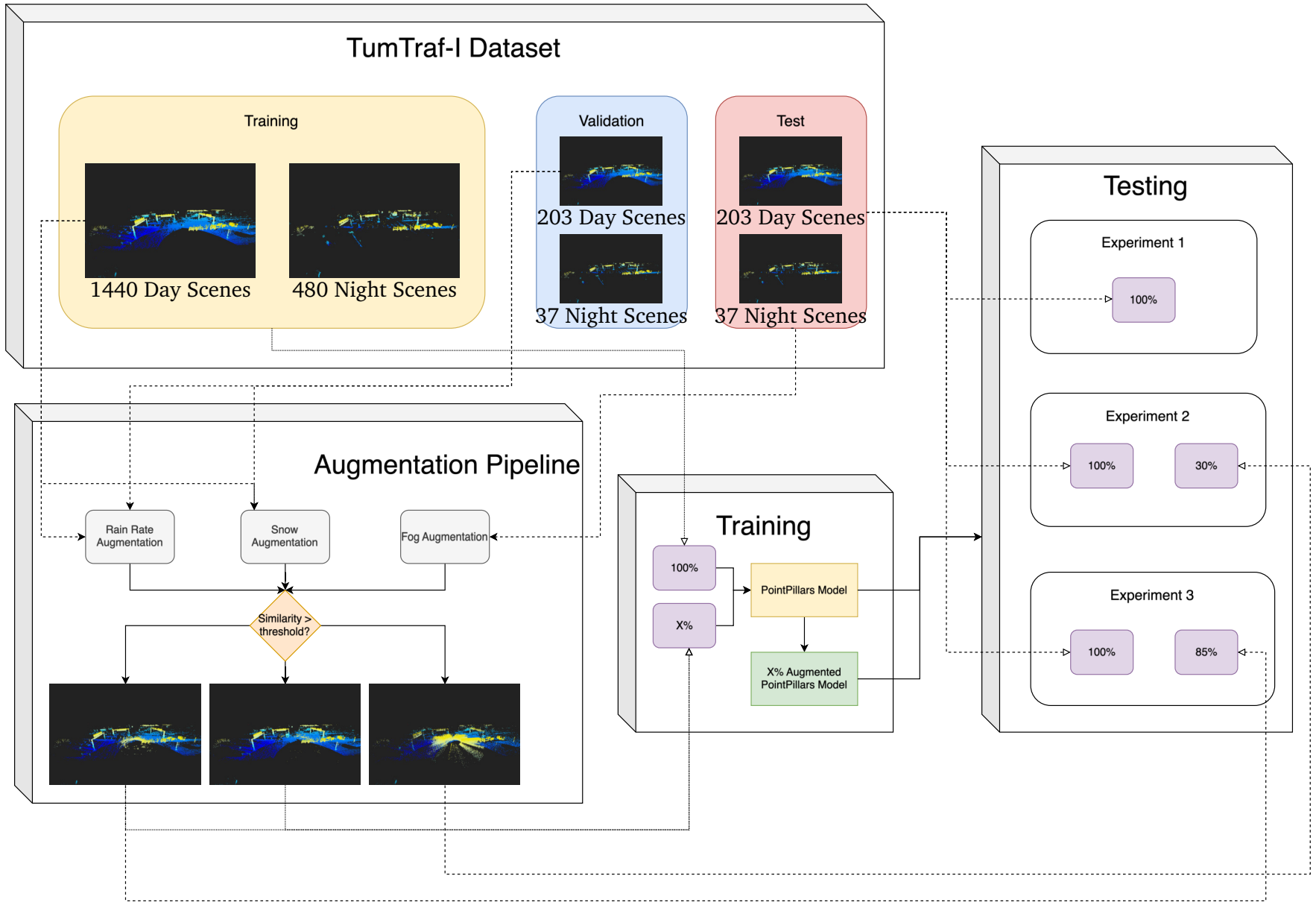


Figure 4.1: Pipeline of the Augmentation Process. Note that X in [20, 40, 60, 80, 100]% and similarity threshold varies from one adverse weather to another.

Fog To simulate the fog effect, we use the fog simulation method used in `LiDAR_fog_sim` library [Hah+21]. The library provides two types of fog simulation: soft and hard.

- The $P\_R\_fog\_hard$ function calculates the Euclidean distance (r_0) of each point from the origin, then modifies the intensity of each point by applying an exponential decay based on the fog attenuation coefficient (alpha) and the distance (r_0). The modified intensities are rounded, and the function returns the updated point cloud. We keep the same fog attenuation coefficient α used in [Hah+21] to 0.06.
- The $P\_R\_fog\_soft$ function used in [Hah+21] simulates a soft fog effect on a point cloud. It calculates fog responses for each point based on distance and original intensity using a backscattering coefficient β, adjusts the point's coordinates and intensity if the fog response exceeds the original intensity, and optionally adds noise using different variants. We set the backscattering effect to ≈ 0.000921 sr^{-1} following the calculation in [Hah+21] and using $\alpha = 0.06$.

  Given $\alpha = 0.06$, we can calculate the values for mor and β.

  The equation for the meteorological optical range (mor) is:

  $$\text{mor} = \frac{\log(20)}{\alpha}$$

  Substituting $\alpha = 0.06$:

  $$\text{mor} = \frac{\log(20)}{0.06} \approx 49.93$$

  The equation for the backscattering coefficient (β) is:

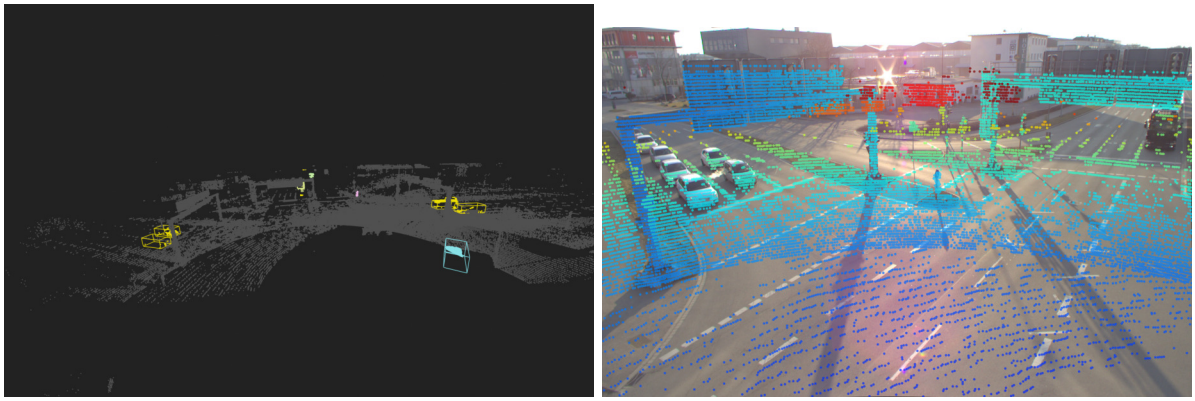
  $$\beta = \frac{0.046\alpha}{\log(20)}$$

  Substituting $\alpha = 0.06$:

  $$\beta = \frac{0.046 \times 0.06}{\log(20)} \approx 0.000921 \text{ sr}^{-1}$$

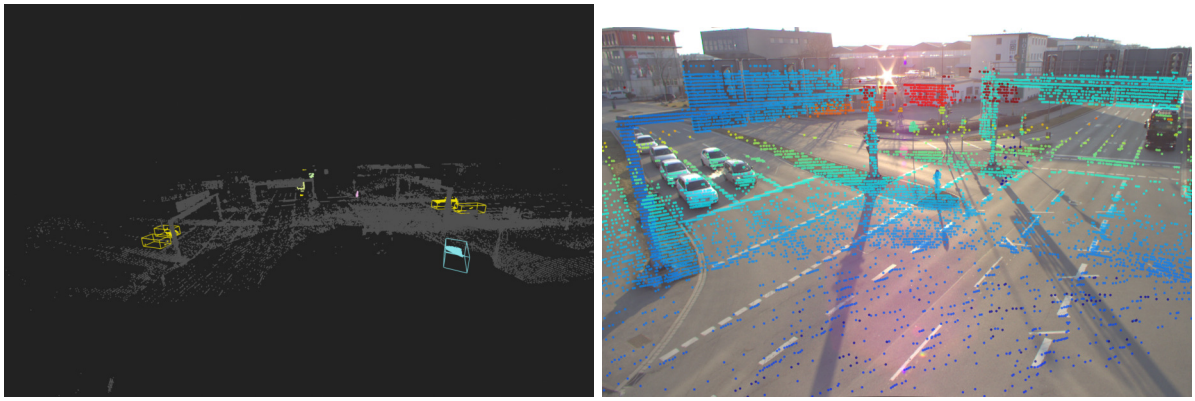
4.1.2 Augmentation Results

The results of the data augmentation process demonstrate the impact of different rain rates on the point cloud data. The following images show the original point cloud and the augmented point clouds for various rain rates, illustrating how the density and visibility of points are affected by simulated rain conditions.



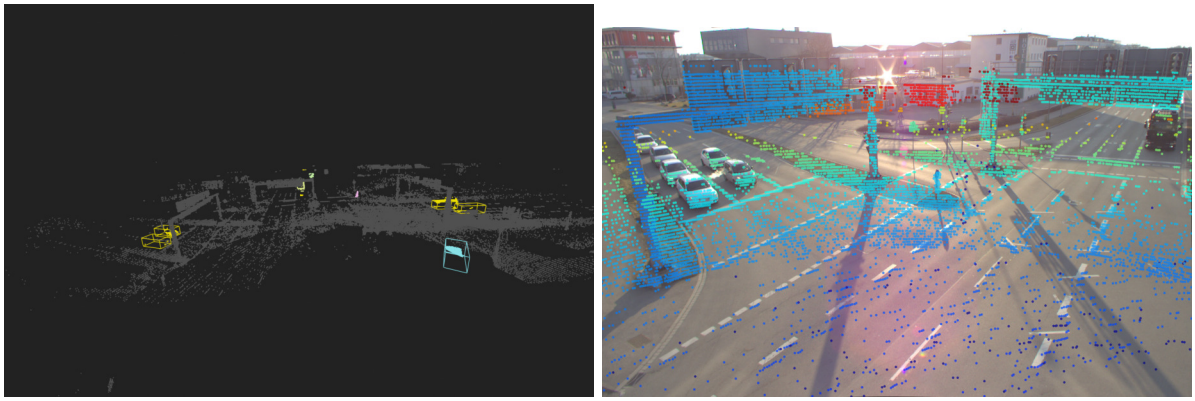
(a) Original Point Cloud **(b)** Original RGB Image

Figure 4.2: Original Point Cloud and RGB Image



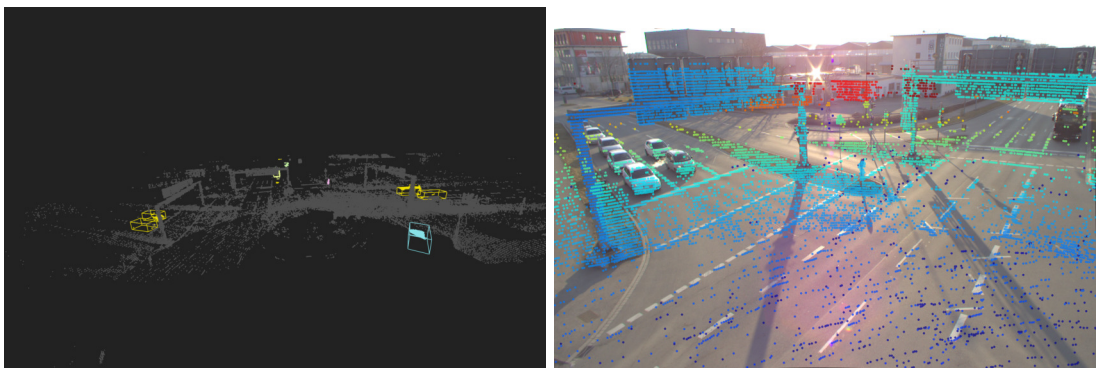
(a) Augmented Point Cloud with Rain Rate of 2.5 mm/hr **(b)** RGB Image with Rain Rate of 2.5 mm/hr

Figure 4.3: Point Cloud and RGB Image with Rain Rate of 2.5 mm/hr



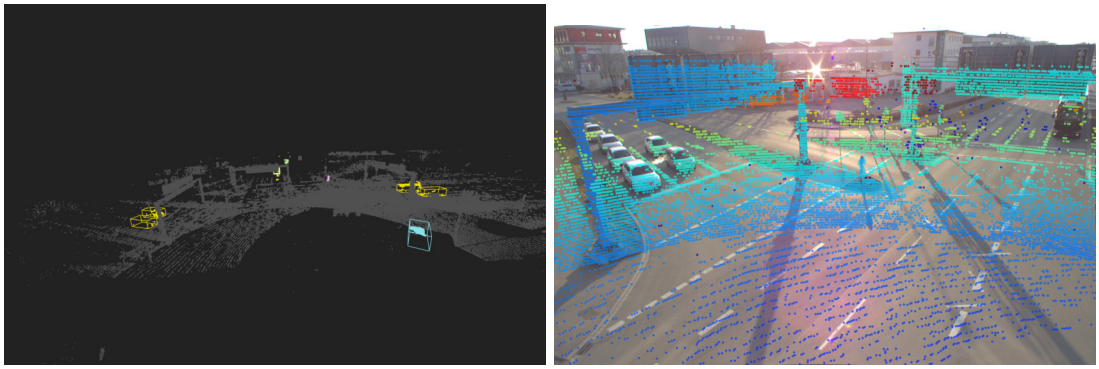
(a) Augmented Point Cloud with Rain Rate of 5.6 mm/hr **(b)** RGB Image with Rain Rate of 5.6 mm/hr

Figure 4.4: Point Cloud and RGB Image with Rain Rate of 5.6 mm/hr



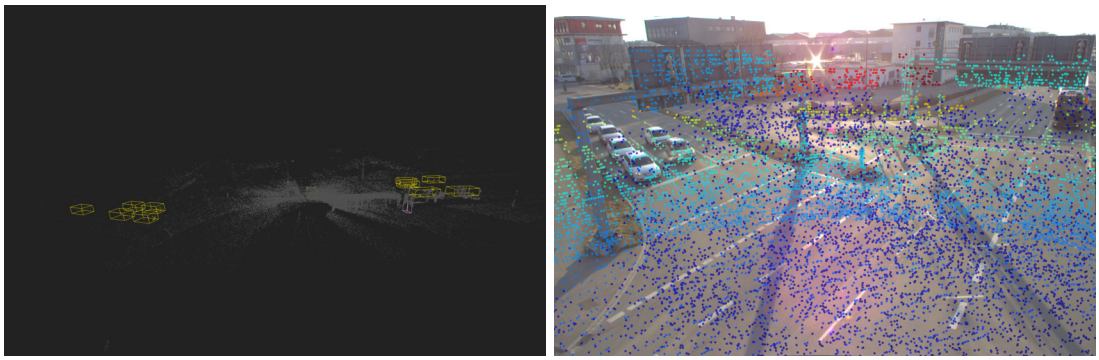
(a) Augmented Point Cloud with Rain Rate of 18.8 mm/hr **(b)** RGB Image with Rain Rate of 18.8 mm/hr

Figure 4.5: Point Cloud and RGB Image with Rain Rate of 18.8 mm/hr



(a) Augmented Point Cloud with Snow **(b)** RGB Image with Snow

Figure 4.6: Point Cloud and RGB Image with Snow



(a) Augmented Point Cloud with Fog **(b)** RGB Image with Fog

Figure 4.7: Point Cloud and RGB Image with Fog

Figure 4.2: Original Point Cloud and RGB Image. This image shows the original point cloud without any augmented adverse weather conditions. The points are of high clarity and dense, providing a clear view of the environment.

Figure 4.3: Point Cloud and RGB Image with Rain Rate of 2.5 mm/hr. This image demonstrates the point cloud augmented with a 2.5 mm/hr light rain rate. The point density starts to reduce slightly, with some scattering visible due to the light rain simulation.

Figure 4.4: Point Cloud and RGB Image with Rain Rate of 5.6 mm/hr. Here, the point cloud is augmented with a moderate rain rate of 5.6 mm/hr. The scattering effect is more pronounced, and the visibility of the points decreases compared to the light rain condition.

Figure 4.5: Point Cloud and RGB Image with Rain Rate of 18.8 mm/hr. This image shows the point cloud augmented with a heavy rain rate of 18.8 mm/hr. The intense rain simulation significantly reduces the point density and affects visibility.

Figure 4.6: Point Cloud and RGB Image with Snow. This image shows the point cloud augmented with simulated snow conditions. As seen on the back of the image, the snow simulation adds some noise resulting from the snowflake points reflected to the LiDAR sensor and some scattered points at the front of the point cloud.

Figure 4.7: Point Cloud and RGB Image with Fog augmentation. This image shows the point cloud augmented with a fog effect. We used both hard and soft fog augmentations, which resulted in the scattering of the points at further range and the formation of heavy noise around the sensor.

4.2 Backwards Point Concatenation

In this section, we present an approach for enhancing the density of point clouds by concatenating points from previous frames; we will refer to it as *Backwards Point Concatenation*. [Kem+23a] have experimented with this approach yet did not report any significant improvement in results. We suspect that the reason behind that is that the concatenation of consecutive point clouds results in the cumulative addition of the same point clouds with a slight position offset. This overlap can lead to redundancy and noise in the data, which diminishes the potential benefits of increased point cloud density.

We introduce a filtering step within the concatenation process to mitigate this issue. This filtering step ensures that unique and meaningful points from previous and subsequent frames are added to the current point cloud. By applying minimum and maximum distance-based thresholds, we can effectively exclude points too close or too far from existing points in the current cloud, thereby maintaining the integrity and quality of the resulting dataset. In Chapter 5, we set the optimal distance threshold between points in a point cloud based on a grid search approach. The whole concatenation pipeline is illustrated in Figure 4.8.

4.2.1 Algorithm Description

Algorithm 3 explains the pipeline we use for the backward concatenation of the point clouds with a given offset. The algorithm iterates on the point clouds in the dataset in a backward direction, concatenating all o (described by `offset`) point clouds preceding the current point cloud. The algorithm accepts the concatenation if the concatenated point cloud is at least X% similar to the current point cloud. Let's consider the current point cloud:

$$p_t = \{x_t, y_t, z_t, i_t\} \tag{4.1}$$

where x_t, y_t, z_t are the coordinates and i_t is the intensity at time t.

A concatenated point cloud with offset $t-i$ can be described as follows:

$$p_{c_{t-i}} = \bigcup_{k=0}^{i} p_{t-k} \tag{4.2}$$

The final concatenated point cloud, given the condition that the original point cloud p_t is at least X% similar to the concatenated point cloud up to $t-i$, is represented with the following equation:

$$p_c = \bigcup_{\substack{i=0 \\ S_{ICP}(p_{c_{t-i}}, p_t) \geq X\%}}^{o} p_{t-i} \tag{4.3}$$

where o is the offset representing the maximum number of point clouds to be concatenated, and $S_{ICP}(p_{c_{t-i}}, p_t)$ denotes the ICP similarity between the current point cloud p_t and the resulting concatenation up to i preceding point clouds. By incorporating the ICP similarity measurement step, we further enhance the robustness of our concatenation method. This step ensures that only point clouds that maintain a structural similarity of over 90% with the original point cloud are included, thereby preserving the overall data integrity and reducing potential noise and ghost objects. The similarity threshold X and the offset o are hyperparameters that we will fine-tune in Chapter 5.

Algorithm 3 Concatenate Point Clouds

Input: Index idx, Distance Threshold Min $distance\_threshold\_min$, Distance Threshold Max $distance\_threshold\_max$, Offset $offset$, Current Point Cloud $current\_pc$
Output: Concatenated Point Cloud $concat\_pc$, Original Point Cloud $current\_pc$
procedure CONCATENATE_POINT_CLOUDS($idx, current\_pc$)
    $concat\_pc \leftarrow current_pc$
    if $offset > 0$ **and** $idx - offset \geq 0$ **then**
        for $i \leftarrow 0$ to $offset - 1$ **do**
            $prev\_pc \leftarrow dataset[idx - offset]$
            $filtered\_points \leftarrow filter\_points(current\_pc, prev\_pc)$
            $concat\_pc \leftarrow concat\_pc + filtered\_points$
            $similarity \leftarrow calculate\_similarity(concat\_pc, current\_pc, method = icp)$
            if $similarity <= 0.9$ **then**
                $concat\_pc \leftarrow save\_current\_pc$
            end if
        end for
    end if
    return $np.unique(current\_pc, return\_index = True, axis = 0)[0], save\_current\_pc$
end procedure

4.2.2 Point Filtering Algorithm

Algorithm 3 shows a further step, `filter_points`, that adds a sanity check on top of the similarity measure. Before concatenating the point cloud at time $t - i$, we filter the points in the point cloud p_{t-i} based on their distances from a set of base points. This filtering is achieved using specified distance thresholds. A point $p_{t-i,j}$ from p_{t-i} is included in the filtered output if the measured distance to its closest base point, denoted by p_{t_c} is within the minimum and maximum thresholds. Suppose we define a point cloud at instance t as p_t, the offset for concatenation as o, and the distance thresholds as D_{min} and D_{max}. In that case, the following rule should apply for a point cloud p_{t-o} to be concatenated to the point cloud p_t: A point $p \in p_{t-o}$ will be concatenated to p_t if $D_{min} < d(p,q) < D_{max}$ for all $q \in p_t$, where $d(p,q)$ is the Euclidean distance between the point p and its closest neighbor in p_t. This guarantees that most ghost objects from previous point clouds that are not part of the point cloud p_t are eliminated. The addition of the filtering step transforms equations 4.2 and 4.3 to the following:

$$p_{c_{t-i}} = \bigcup_{k=0}^{i} \left\{ p_{t-k,j} \mid D_{min} < \min_{p_{t,l} \in p_t} d(p_{t-k,j}, p_{t,l}) < D_{max} \right\} \tag{4.4}$$

where $l \in [0, N_t]$, N_t being the number of points in p_t, and $j \in [O, N_{t-k}]$, N_{t-k} being the number of points in p_{t-k}.

$$p_c = \bigcup_{\substack{i=0 \\ S_{ICP}(p_{c_{t-i}}, p_t) \geq X\%}}^{o} \left\{ p_{t-i,j} \mid D_{min} < \min_{p_{t,k} \in p_t} d(p_{t-i,j}, p_{t,k}) < D_{max} \right\} \tag{4.5}$$

where $k \in [0, N_t]$, N_t being the number of points in p_t, and $j \in [O, N_{t-i}]$, N_{t-k} being the number of points in p_{t-k}.

The filtering process is also described by Algorithm 4. To improve the runtime performance at inference time, we use the KD-tree structure to determine the distances between the points in the original point cloud and those in the precedent point cloud to be calculated. KD-tree efficiently organizes the points to minimize these distance calculations during the query process

[Skr19] and uses the Euclidean distance metric. For large datasets, the KD tree significantly speeds up finding the nearest base point to each new point compared to a brute-force search, which would involve calculating the distance from each new point to every base point.

Algorithm 4 Filter Points

Input: Base Points $base\_points$, New Points new_points, Distance Threshold Min $threshold\_min$, Distance Threshold Max $threshold\_max$
Output: Filtered Points $filtered\_points$
procedure FILTER_POINTS($base\_points, new\_points, threshold\_min, threshold\_max$)
    if len($base\_points$) $= 0$ **then**
        return new_points
    end if
    $tree \leftarrow KDTree(base\_points)$
    $distances, \_ \leftarrow tree.query(new\_points, 1)$
    $mask \leftarrow (distances > threshold\_min)$ **and** $(distances < threshold\_max)$
    $filtered\_points \leftarrow new\_points[mask]$
    return $filtered\_points$
end procedure

In Chapter 5, we describe the experiments we run to define the optimal values for the offset, minimum distance, and maximum distance thresholds, and we evaluate the baseline PointPillars model by applying concatenation on the TUMTraf-I test dataset.

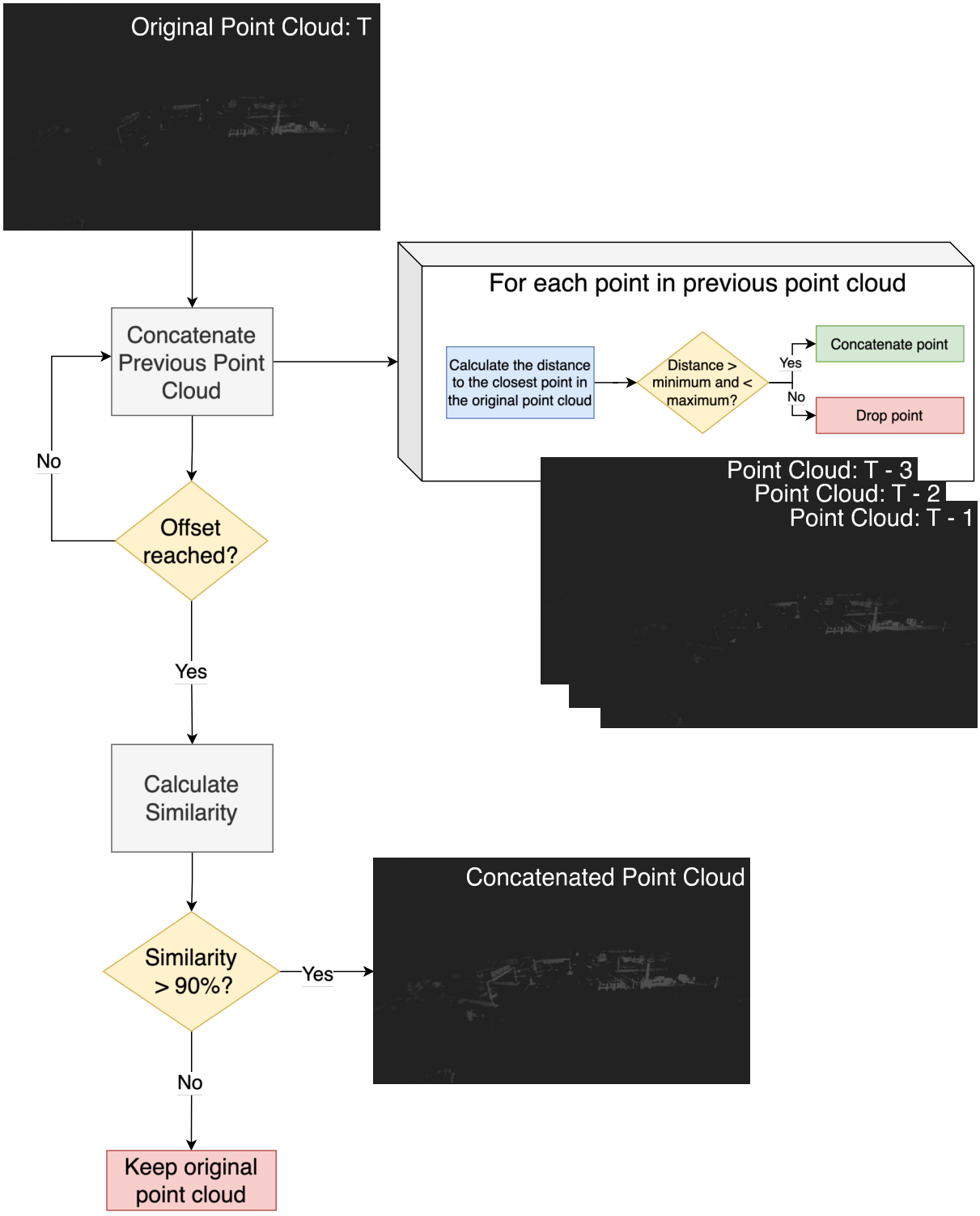


Figure 4.8: Concatenation Pipeline using point filtering based on minimum and maximum distance threshold to original point cloud

Chapter 5

Experiments and Evaluation

In this chapter, we start by describing the experiment setup, then present the experiments we conducted within the scope of the thesis, the results we obtained, and a conclusion on the most optimal approach based on the evaluation results.

5.1 Experimental Setup

5.1.1 Sensors

LiDAR Sensors

The LiDAR sensors used in the experiment are Ouster OS1-64 (generation 2). These sensors have 64 vertical layers and provide a 360-degree field of view (FOV). They are configured in a below-horizon setting, offering a range of up to 120 meters with an accuracy between 1.5 and 10 centimeters. The LiDAR sensors are mounted on a gantry bridge at an intersection, positioned to capture a comprehensive view of the traffic environment from a height of 7 meters. This setup ensures a wide coverage area, allowing detailed 3D point cloud generation essential for robust object detection and tracking in urban scenarios, see Figure 5.1.

Camera Sensors

The camera sensors used in the experiment are Basler ace acA1920-50gc cameras. These cameras have a resolution of 1920x1200 pixels and are equipped with Sony IMX174 global shutter sensors, essential for capturing high-quality images without motion blur. The cameras operate in color mode and are connected via a GigE interface with 8 mm lenses, as described in [Zim+23c]. Like the LiDAR sensors, the cameras are mounted on the gantry bridge, ensuring synchronized and overlapping fields of view with the LiDAR sensors. This configuration allows for the fusion of image and point cloud data, enhancing the overall detection performance by leveraging the strengths of both sensor modalities. The TUMTraf-I dataset used in this series of experiments comes with a set of extrinsic calibration parameters that allow for a precise mapping of the point clouds to the camera-captured pictures (see Figures 5.2 and 5.3).

5.1.2 Dataset Statistics

The TUM Traffic Intersection dataset used in this thesis provides a comprehensive and diverse collection of urban traffic scenarios designed to support the development and validation of autonomous vehicle technologies. This second dataset release contains 2,400 frames, divided

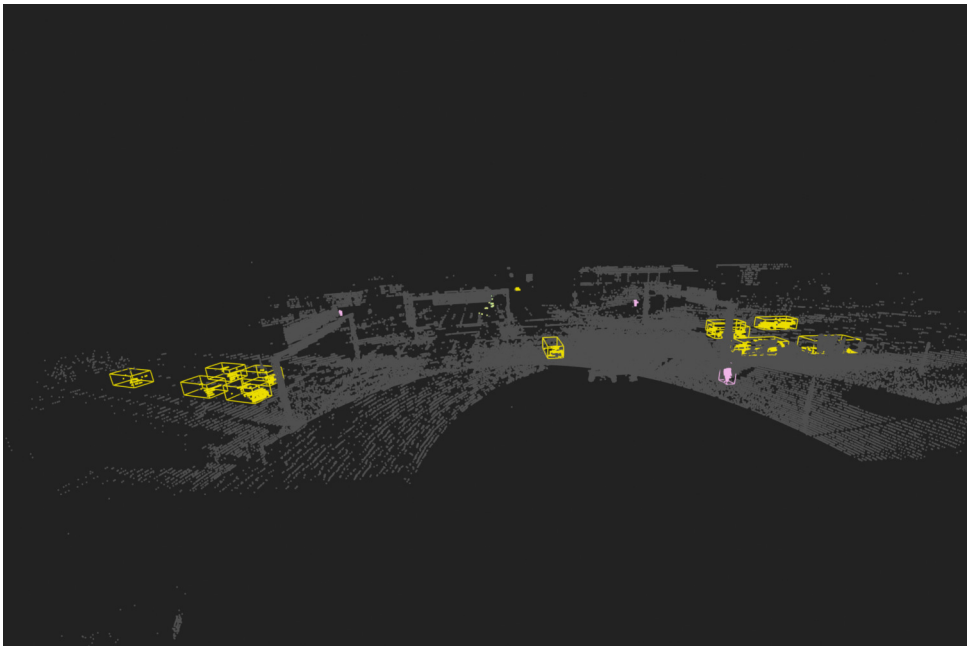
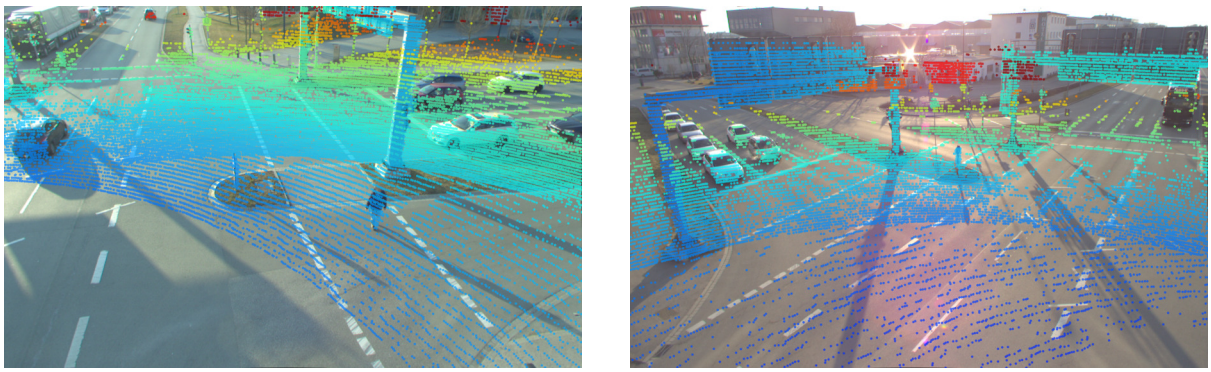


Figure 5.1: 3D point cloud generated by the Ouster OS1-64 LiDAR sensor.

(a) First position

(b) Second position

Figure 5.2: Images captured by the Basler ace camera facing south.



(a) First position with projected point clouds

(b) Second position with projected point clouds

Figure 5.3: Images captured by the Basler ace camera with projected point clouds using calibration data.

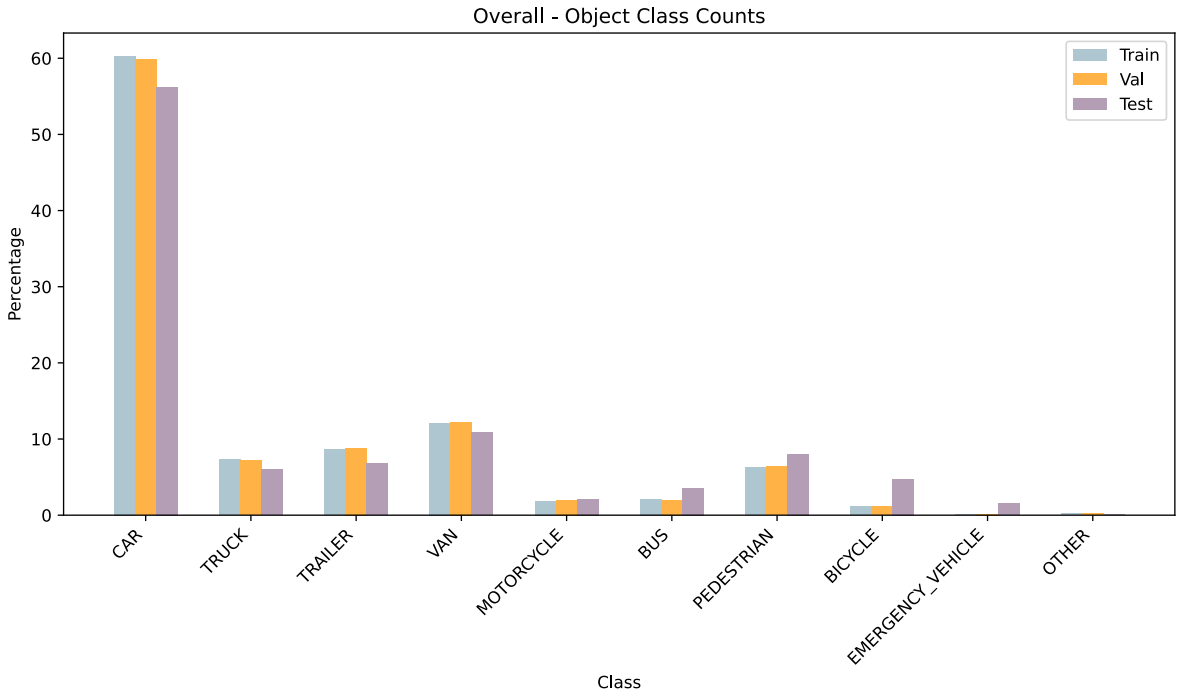


Figure 5.4: Object class count for train, validation, and test datasets.

into 1,920 for training, 240 for validation, and 240 for testing. The dataset includes both point clouds and RGB images, though this thesis focuses specifically on the point cloud data.

The dataset encompasses ten object classes: CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY VEHICLE, and OTHER. These classes reflect a wide range of road users and vehicles, ensuring the robustness and versatility of detection algorithms trained on this data.

Object Class Counts

CARs are the most common class across all datasets for the object class counts, constituting 60.3% of the training set, 59.9% of the validation set, and 56.2% of the test set. VANs, TRAILERs, and TRUCKs follow, with consistent distributions across the datasets, ensuring a balanced evaluation, as seen in Figure 5.4.

Environmental Conditions

As reported in [Zim+23c], the dataset statistics show that environmental variations are captured through rainy scenes that account for 25% of the dataset. The rainy scenes are also scenes that were captured at night time with lower visibility and a lower number of objects in the sensor's surroundings.

Point Cloud Characteristics

Further analysis of the point cloud data reveals essential characteristics such as the number of points, average density, and average intensity. The number of points in each point cloud varies significantly across all datasets, with most clouds containing between 40,000 and 100,000 points. The average density of the point clouds, defined as the number of points per unit volume, falls between 0.02 and 0.16 across all datasets, ensuring a detailed representation of the scanned environment essential for accurate object detection and scene interpretation. The

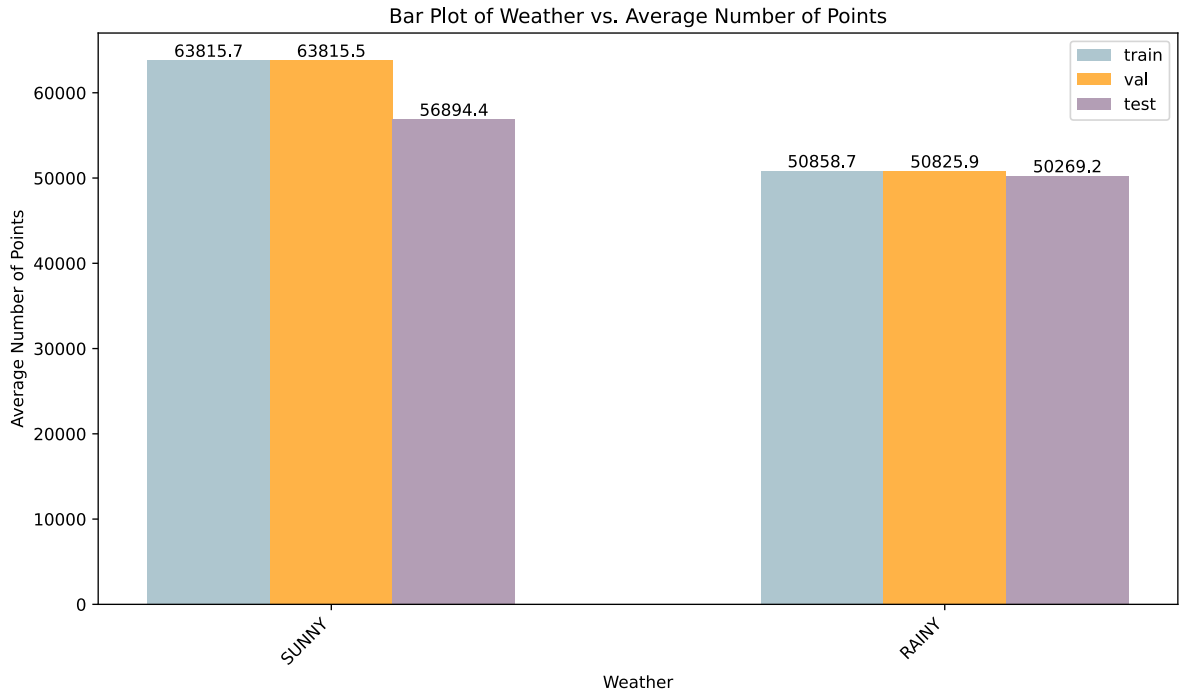


Figure 5.5: Average number of points in the point clouds for sunny and rainy weather conditions in the train, validation, and test datasets.

average intensity values of the points, which indicate the reflectivity of surfaces, generally range from 0.0075 to 0.025 in the training set, 0.008 to 0.022 in the validation set, and 0.008 to 0.020 in the test set, with higher intensity values indicating strong reflections helpful in identifying certain materials and surface types such as wet ground and raindrops. Figure 5.5 illustrates the distribution of points for the two different weather conditions in the dataset, which shows an ≈ 20% drop in the average number of points in rainy weather conditions.

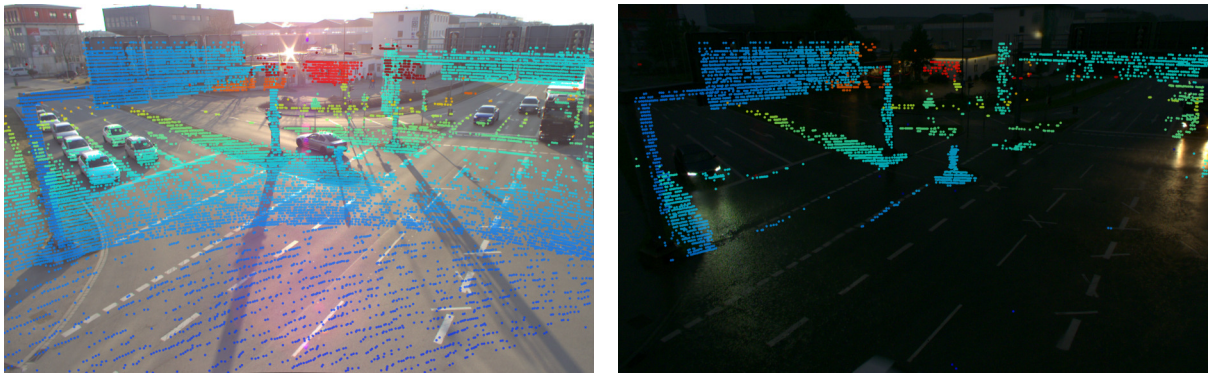
Impact of Environmental Conditions

The average intensity and number of points against weather conditions show that point clouds captured in sunny conditions tend to have slightly higher intensity values and more points than those captured in rainy conditions. These observations show the influence of lighting and weather conditions on the quality and characteristics of point cloud data. Figure 5.6 highlights the impact of different weather conditions on LiDAR point clouds using images from the TUMTraf-I dataset. Subfigure 5.6a shows a point cloud captured during the daytime with clear weather, while subfigure 5.6b shows a point cloud captured at night during rainy conditions. These images highlight how environmental factors such as lighting and precipitation can affect the quality and density of LiDAR data.

5.1.3 Evaluation of the Baseline Model

The baseline model for this thesis, PointPillars, encodes 3D point clouds into a pseudo-image format, facilitating efficient 2D convolutional neural networks. Initially introduced by [Lan+19], PointPillars achieves impressive processing speeds by streamlining the input data structure, thus enabling rapid computation without considerable accuracy loss [Lan+19].

The model is configured to detect ten classes of objects: CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY VEHICLE, and OTHER. Further details



(a) Daytime with Sun

(b) Nighttime with Rain

Figure 5.6: Examples from the TUMTraf dataset showing LiDAR point clouds under different weather conditions.

on the baseline model's configuration are given in Chapter 2. We will adapt the same setup for training in our experiments.

The PointPillars model in [Zim+23a] was evaluated on the TUM Traffic Intersection (TUMTraf-I) dataset and achieved significant performance across various metrics. The authors have measured the model's accuracy, precision, recall, and average precision (AP) across different object classes. They have also evaluated its detection capabilities on an NVIDIA RTX 4090 GPU system and reported the results we summarized in Table 5.1. The PointPillars implementation was adapted from OpenPCDet and trained on the TUMTraf-I dataset with specific configurations: point cloud range limited to -64 to 64 meters in x-y direction and -8 to 0 meters in the z direction, voxel size set to [0.16, 0.16, 8], and trained for 160 epochs using Adam optimizer with a learning rate of 0.003 and weight decay of 0.01 [Zim+23a]. The authors of [CKY24] and [Wan+24] report an improved performance of the 3D detectors when using LiDAR and Radar fusion on frames recorded under adverse weather conditions. Therefore, we will reuse the fusion inference pipeline used for the baseline PointPillars model in [Zim+23a] in our study as well. The series of experiments run in this thesis are done using a system equipped with an NVIDIA GeForce RTX 3090 GPU. Therefore, we have obtained slightly different results for the baseline model from the ones obtained in [Zim+23a]. We reevaluated the baseline model with the same configuration and test data to draw valid conclusions from the results we received in our experiments. The Pointpillars-based model trained with early fusion (EF) and full dataset (Point Cloud S+N) will be the baseline model for our experiments in this thesis. Reevaluating the model using the 240 registered point clouds from the test set from the TUMTraf dataset returned the results highlighted in Tables 5.1 and 5.2.

Table 5.1: Performance Metrics of Baseline PointPillars Model

	RTX 4090			RTX 3090			
Class	**Precision (%)**	**Recall (%)**	**AP (%)**	**Precision (%)**	**Recall (%)**	**AP (%)**	**IoU**
Car	71.75	87.33	71.64	70.46	52.09	69.86	0.72
Truck	91.20	85.03	91.03	64.46	41.47	63.75	0.71
Motorcycle	82.72	70.71	82.37	23.53	4.87	22.00	0.42
Bus	99.93	100.00	99.93	87.27	81.17	87.02	0.58
Pedestrian	31.37	25.49	30.00	25.31	14.56	23.82	0.35
Bicycle	36.02	80.77	35.93	57.54	48.95	56.69	0.33
Trailer	-	-	-	75.75	56.86	75.26	0.70
Van	-	-	-	52.62	27.15	51.67	0.74
Emergency Vehicle	-	-	-	100.00	100.00	100.0	0.39
Other	-	-	-	80.39	64.31	80.00	0.33
Mean	62.85	51.22	62.11	81.5	53.29	63.01	0.53

Table 5.2: Runtime Performance of PointPillars Model with registered North and South LiDAR sensors

GPU	Runtime (ms)	FPS
RTX 4090	26.11	38
RTX 3090	24.21	41

Since we are focusing on improving the model's performance under adverse weather conditions, we also evaluated the baseline model under night rainy scenes using the 37 night frames in the dataset. The results are reported in Table 5.3.

Table 5.3: Performance Metrics and IoU of Baseline PointPillars Model under Night Rainy Scenes

Class	Precision (%)	Recall (%)	AP (%)	IoU
Car	70.59	48.79	70.00	0.82
Truck	100.00	100.00	100.00	0.71
Motorcycle	0.00	0.00	0.00	0.00
Bus	100.00	100.00	100.00	0.84
Pedestrian	0.00	0.00	0.00	0.00
Bicycle	0.00	0.00	0.00	0.00
Trailer	88.24	77.21	88.00	0.65
Van	66.67	44.44	66.00	0.42
Emergency Vehicle	0.00	0.00	0.00	0.00
Other	0.00	0.00	0.00	0.00
Mean	42.50	37.11	42.40	0.34

In addition, we visualized the model's performance in rainy night scenes using the Devkit [Dat24] provided in [Zim+23c]. Figure 5.7 illustrates the false negatives detected in night rainy scenes after running the detection using the baseline model. The qualitative results show that false negatives are registered in close and distant ranges, indicating the model's decreased performance in adverse weather.

5.2 Data Augmentation

We used the augmentation techniques detailed in 4 to enhance the original 1920 training point clouds and the 240 validation points with artificial rain and snow effects. Specifically, rain effects were applied at rates of 2, 5.6, and 18.8 mm/h, corresponding to light and moderate rain conditions, while snow effects were simulated at a rate of 71 mm/h. These point clouds, captured during daylight, comprised over 70,000 points each. Consequently, we generated 1772 augmented training point clouds and 200 validation point clouds. To construct the five augmented models utilized in our experiments, we randomly selected subsets comprising 20%, 40%, 60%, 80%, and 100% of the generated point clouds and integrated them with the original datasets. We retrained the baseline PointPillars model using each of the obtained datasets and obtained five augmented models that we will use in the following series of experiments. To have more insights into the effect on performance resulting from the augmentation approach, we test the models on the subsequent test datasets:

- The original dataset contains 240 frames, with 37 captured at night and in rainy conditions (Experiment 1).
- An augmented version of the test dataset with an unseen adverse weather condition, fog. We added 144 fog scenes to the test dataset to obtain a total of 384 frames (Experiment 2).

(a) False negatives in night rainy scenes (image 1).

(b) False negatives in night rainy scenes (image 2).

Figure 5.7: False negatives detected in night rainy scenes using the baseline model. Blue boxes represent ground truth, and pink boxes represent the model's detections.

Table 5.4: Number of Point Clouds After Augmentation

Augmentation Level	Training Point Clouds	Validation Point Clouds
Baseline	1920	240
+20%	2278	282
+40%	2609	320
+60%	2882	336
+80%	3163	350
+100%	3148	378

- An augmented version of the test dataset with scenes simulating rainy conditions with the same rain rates used for the training process. We added 210 fog scenes to the test dataset to obtain a total of 450 frames (Experiment 3.1).
- An augmented version of the test dataset with scenes simulating rainy conditions with different rain rates from the ones used for the training process (25, 35, and 45 mm/h). We added 210 fog scenes to the test dataset to obtain 450 frames (Experiment 3.2).

Table 5.4 summarizes the datasets used to train the five models we will use in the following series of experiments.

5.2.1 Experiment 1: Original Test Dataset

In this experiment, we evaluated the models using the original test dataset, which contains 240 frames, 37 of which are night scenes in rainy conditions. We used evaluation metrics, including accuracy, precision, Intersection over Union (IoU), and average precision at 0.5 IoU. These metrics were analyzed across different augmentation levels to assess the effectiveness of the data augmentation strategies. The results are summarized in Tables 5.6 and 5.5. We can observe the following trends in the performance results:

Average Precision (AP) Improvements According to the results, data augmentation at different augmentation levels enhanced the Average Precision (AP) metrics across all scenes and object classes. For instance, the CAR class AP increased from 69.86% at baseline to 75.24% at 100% augmentation ($\approx$ 5.38% improvement), while the TRUCK class AP improved from 63.75% at baseline to 71.40% at 60% augmentation ($\approx$ 7.65% improvement).

Intersection over Union (IoU) Metrics IoU values also demonstrate improvement with data augmentation. While the CAR class exhibited the highest IoU at baseline (0.7245) and maintained strong performance across all augmentation levels, the TRUCK class achieved the highest IoU at 60% augmentation (0.7173).

Precision and Recall Augmentation improves both precision and recall metrics. For the CAR class, precision increased from 70.456% at baseline to 75.722% at 100% augmentation, while recall improved from 52.091% to 60.077%. Similar trends were observed for other classes, indicating enhanced detection accuracy and object identification.

Day and Night Scene Performance Data augmentation consistently improved AP and IoU metrics during the day. For example, the CAR class AP increased from 69.70% at baseline to 75.08% at 100% augmentation. For the night scenes, we can observe at least a 0.69% improvement of the mean AP value and an improvement in the detection of the CAR class by

Table 5.5: Summarized Mean IoU, Precision, and Recall metrics for overall, day, and night scenes across various augmentation levels.

Metric	Scene	Baseline	+20%	+40%	+60%	+80%	+100%
IoU	Overall	0.5257	0.5188	**0.5294**	0.5075	0.5134	0.5173
	Day	0.5124	0.5019	0.5179	0.4937	0.4999	0.5026
	Night	0.3445	0.3439	0.3457	**0.3494**	0.3377	0.3310
Precision	Overall	81.500	**83.54**	**83.54**	82.917	82.549	83.370
	Day	79.326	81.476	64.109	80.937	80.549	81.370
	Night	42.500	42.500	42.500	42.500	42.500	42.500
Recall	Overall	53.292	56.718	56.299	**57.720**	54.810	55.661
	Day	50.733	54.617	54.067	55.740	52.767	53.434
	Night	37.107	37.094	**38.057**	37.067	36.409	37.761

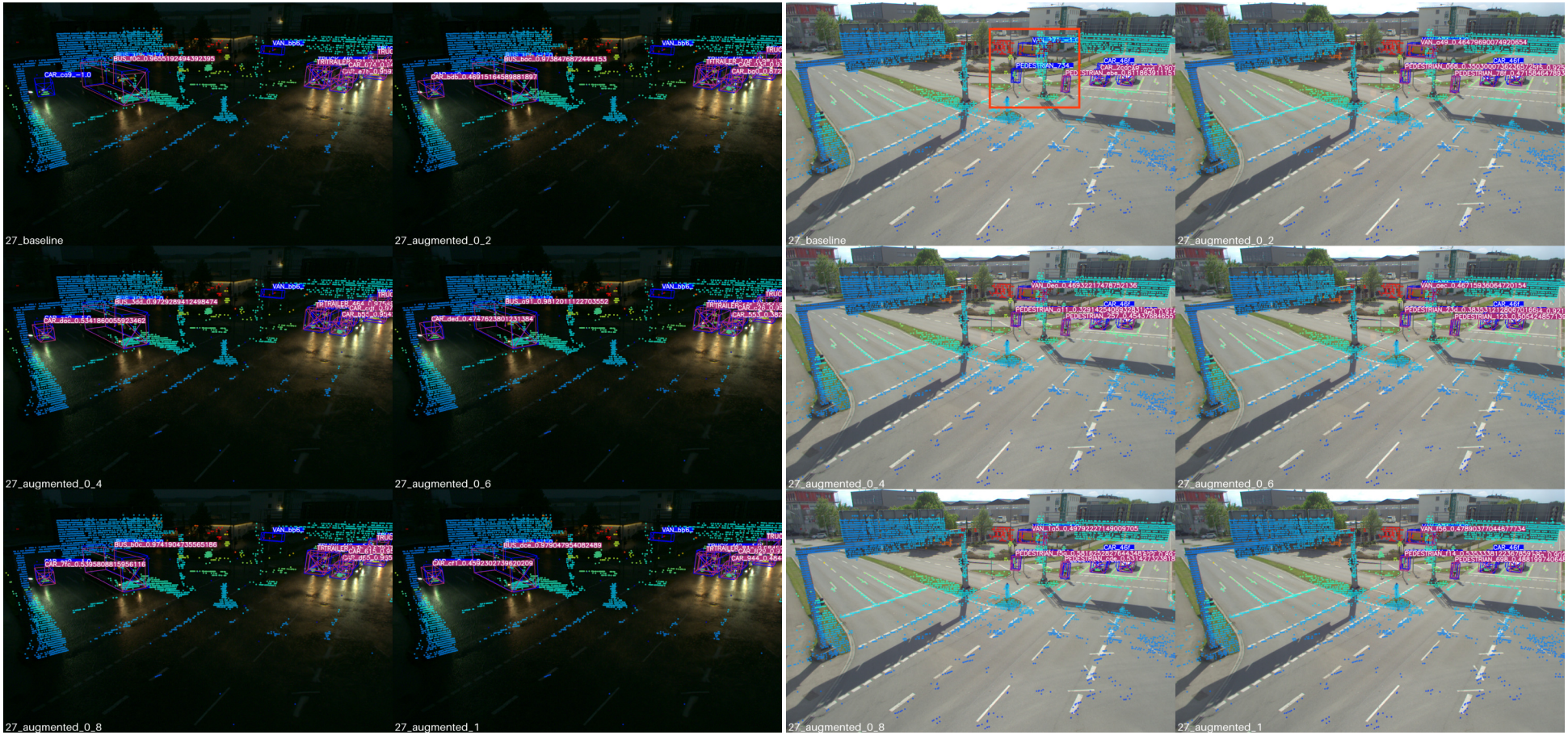
12.84% (at 80% augmentation). Figure 5.8a shows the detection results of the baseline model and all five augmentation levels. While the baseline model missed the car at the back of the visibility range of the LiDAR, all five augmented models could detect the car, with the 60% model having the highest score.

Class-Specific Performance Data augmentation generally enhances model robustness and performance, as seen in the mean IoU for overall scenes, which peak at 40% augmentation (0.5294). However, different classes benefit differently. For example, the CAR class shows the highest AP at 100% augmentation (75.24%). In contrast, the TRUCK class shows substantial improvements, particularly at 60% augmentation (AP: 71.40%, IoU: 0.7173), and classes like VAN and TRAILER peak at lower augmentation levels (e.g., 40%). Despite improvements, sensitive classes like MOTORCYCLE and PEDESTRIAN are absent from the night scenes, suggesting a need for more targeted strategies.

Distance-Based Trends Performance improvements vary by distance, with certain augmentation levels optimizing detection at specific ranges. For instance, the CAR class sees notable improvements in the 40-50 m range with a peak AP at 60% augmentation (79.11%), while the TRUCK class performs best at closer ranges (0-40 m) with a peak AP at 20% augmentation (75.52%). Figure 5.8b shows the augmented models' ability to detect objects farther away from the LiDAR sensor, especially sensitive small object classes like pedestrians.

Optimal Augmentation Level

The 60% augmentation level generally provides the best overall performance (highest mean AP for both day scenes (65.51%) and full dataset (66.49%) as well as for frequent object classes such as TRUCK, TRAILER, and VAN.



(a) Detection results of the baseline model (top left) and the five augmentation results at a night rainy scene.

(b) Detection results of the baseline model (top left) and the five augmentation results at a clear day scene.

Figure 5.8: Detection results of the baseline model (top left) and the five augmentation results at a day vs. night rainy scene. The boxes in blue represent the ground truth, and the ones in pink represent the detections.

Table 5.6: Summarized Average Precision (AP) metrics across all scenes, day scenes, and night scenes for various augmentation levels.

Class	Condition	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	Full	69.86	73.30	74.99	74.98	74.97	**75.24**
	Day	69.70	73.16	74.82	74.83	74.80	**75.08**
	Night	70.00	82.64	82.52	80.90	**82.84**	82.77
TRUCK	Full	63.75	69.50	65.71	**71.40**	67.67	69.79
	Day	59.61	65.36	61.59	**67.21**	63.50	65.69
	Night	**100.00**	**100.00**	**100.00**	98.00	**100.00**	**100.00**
TRAILER	Full	75.26	75.64	**77.21**	73.39	75.21	73.59
	Day	71.34	73.57	**75.03**	71.17	73.03	71.50
	Night	88.00	88.00	88.00	**90.00**	86.00	86.00
VAN	Full	51.67	53.49	53.68	**57.63**	51.79	53.63
	Day	51.64	51.64	53.66	**55.76**	51.79	53.54
	Night	66.00	66.00	66.00	66.00	66.00	66.00
MOTORCYCLE	Full	22.00	30.91	29.35	31.42	**31.52**	26.98
	Day	22.00	30.91	29.35	31.42	**31.52**	26.98
	Night	0.00	0.00	0.00	0.00	0.00	0.00
BUS	Full	87.02	86.57	88.30	85.19	81.38	**89.55**
	Day	86.20	83.95	86.25	83.80	77.67	**87.03**
	Night	**100.00**	**100.00**	**100.00**	96.00	96.00	**100.00**
PEDESTRIAN	Full	23.82	24.45	25.38	28.58	27.86	**30.18**
	Day	23.82	24.45	25.38	28.58	27.86	**30.18**
	Night	0.00	0.00	0.00	0.00	0.00	0.00
BICYCLE	Full	56.69	**62.58**	52.94	62.33	29.54	30.40
	Day	56.69	**62.58**	52.94	62.33	29.54	30.40
	Night	0.00	0.00	0.00	0.00	0.00	0.00
EMERGENCY	Full	100.00	100.00	100.00	100.00	100.00	100.00
	Day	100.00	100.00	100.00	100.00	100.00	100.00
	Night	0.00	0.00	0.00	0.00	0.00	0.00
OTHER	Full	80.00	80.00	80.00	80.00	80.00	80.00
	Day	80.00	80.00	80.00	80.00	80.00	80.00
	Night	0.00	0.00	0.00	0.00	0.00	0.00
Mean AP	Full	63.01	65.64	64.76	**66.49**	61.99	62.94
	Day	62.10	64.56	63.90	**65.51**	60.97	62.04
	Night	42.40	**43.66**	43.65	43.09	43.08	43.48

5.2.2 Experiment 2: Augmented Test Dataset with Unseen Conditions (Fog)

Experiment Setup

In this experiment, we also aim to evaluate the performance of the supervised 3D perception model using an augmented test set. Unlike the first experiment, where the model was tested on the original test set, this experiment introduces an unseen adverse weather condition, specifically fog, forming 30% of the frames in the test dataset. The objective is to assess the model's robustness and generalizability to unseen, challenging conditions. For this experiment, we run the evaluation pipeline for three datasets: only fog simulations, clear scenes, and mixed datasets with clear and foggy scenes. The results for all three sub-experiments are summarized in Tables 5.7 and 5.8.

Table 5.7: Summarized Average Precision (AP) metrics for clear, foggy, and full data (clear + foggy) across various augmentation levels.

Class	Condition	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	Clear	72.06	75.54	75.41	75.55	75.45	**75.72**
	Foggy	41.62	46.42	47.06	48.49	48.25	**49.20**
	Full	63.73	67.05	68.86	68.80	68.71	**69.06**
TRUCK	Clear	65.70	69.31	65.66	69.37	69.68	**69.71**
	Foggy	45.87	52.56	**55.60**	54.92	53.80	49.73
	Full	59.76	65.37	**67.33**	63.61	65.59	65.79
TRAILER	Clear	75.44	**77.51**	77.10	77.37	77.02	75.51
	Foggy	**53.87**	49.81	53.43	49.84	51.80	51.87
	Full	69.31	69.52	67.38	**71.31**	69.23	69.52
VAN	Clear	53.44	53.60	55.66	**57.69**	53.75	55.45
	Foggy	36.26	30.25	30.77	40.04	39.02	**40.84**
	Full	47.43	47.40	**53.49**	47.51	49.21	51.24
MOTORCYCLE	Clear	24.00	33.17	32.94	**33.46**	33.38	31.04
	Foggy	4.00	12.00	12.00	**14.00**	**14.00**	11.27
	Full	18.00	25.13	25.48	23.43	**25.53**	22.83
BUS	Clear	86.91	84.72	87.02	84.25	77.60	**88.24**
	Foggy	**68.03**	58.84	61.83	61.59	45.29	62.09
	Full	84.69	80.78	81.82	**85.75**	73.79	84.65
PEDESTRIAN	Clear	29.11	28.01	30.67	35.43	35.81	**35.81**
	Foggy	3.90	6.24	5.73	**8.62**	7.78	8.49
	Full	18.33	20.52	23.25	21.27	23.43	**25.18**
BICYCLE	Clear	53.17	55.47	46.03	**56.20**	34.90	27.67
	Foggy	0.00	**16.00**	0.67	7.00	6.17	4.00
	Full	45.28	**52.96**	50.66	44.19	23.94	24.36
EMERGENCY	Clear	0.00	0.00	0.00	0.00	0.00	0.00
	Foggy	0.00	0.00	0.00	0.00	0.00	0.00
	Full	100.00	100.00	100.00	100.00	100.00	100.00
OTHER	Clear	74.00	74.00	74.00	74.00	74.00	74.00
	Foggy	50.00	50.00	50.00	50.00	**74.00**	50.00
	Full	70.00	70.00	70.00	70.00	70.00	70.00
Mean AP	Clear	53.38	55.13	54.45	**56.33**	53.16	53.31
	Foggy	30.35	32.21	31.71	**33.45**	33.21	32.75
	Full	57.65	59.87	59.59	**60.83**	56.94	58.26

Average Precision (AP) Improvements The results in Table 5.7 show that the average precision improves using the augmented models for most classes. For instance, the AP score of the CAR class improves by 7.58% compared to the baseline model when using the 100% augmented model and evaluated on the fog-augmented scenes. Besides, while smaller object classes like pedestrians, bicycles, and motorcycles are barely or not at all detectable by the baseline model in foggy scenes (AP is 0% for the bicycle class using the baseline model), the augmented models score 4,7% (PEDESTRIAN with 60% augmentation) to 16% (BICYCLE with 20% augmentation) better. These results are visualized in Figure 5.9.

Precision, Recall, and (IoU) Table 5.8 shows that the IoU values improve with augmentation models. The 80% augmentation model improves the IoU by 3.33% compared to the baseline fog simulation model. The same observation can be made for the precision and recall metrics.

Table 5.8: Summarized Mean IoU, Precision, and Recall metrics for clear, foggy, and overall scenes across various augmentation levels.

Metric	Scene	Baseline	+20%	+40%	+60%	+80%	+100%
IoU	Clear	0.4999	0.4951	**0.5137**	0.4982	0.4948	0.5029
	Foggy	0.4147	0.4389	0.4272	0.4462	**0.4480**	0.4420
	Overall	0.5196	0.5141	0.5036	**0.5278**	0.5101	0.5149
Precision	Clear	76.25	**78.50**	78.33	77.71	77.76	78.17
	Foggy	66.82	**73.42**	68.33	72.50	72.22	72.50
	Overall	81.500	**83.452**	82.917	70.0	82.668	83.370
Recall	Clear	49.11	51.79	51.37	**53.49**	51.09	51.13
	Foggy	26.92	31.28	30.20	**31.65**	31.60	31.53
	Overall	48.543	51.895	51.783	**52.872**	50.540	51.206

The 20% augmentation model scores the highest values for the three test datasets and shows an improvement of 6.6% for foggy scenes and of ≈ 2% overall, which means that the model has the lowest false positive detections. The 60% model, on the other hand, scores best in the recall values, showing an improvement of ≈ 5% in foggy scenes, which means that the model is best at detecting the existing objects.

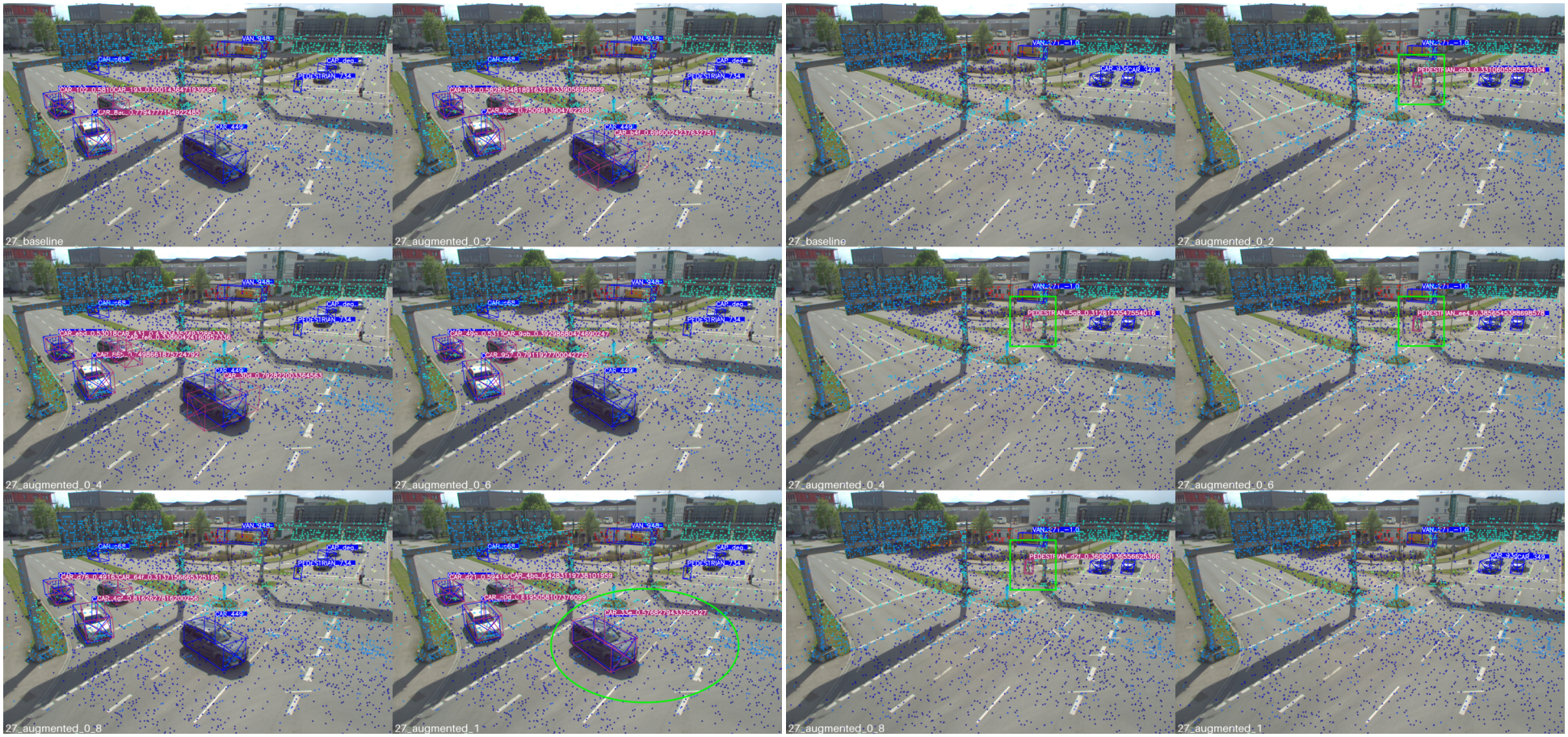
Class-Specific Performance Like in the first experiment, different classes benefit differently from augmentation. For example, the CAR class shows the highest AP at 100% augmentation in clear and foggy conditions. However, the TRUCK class had a higher AP at 40% augmentation, while other object classes, such as VAN and TRAILER, showed varying results at different augmentation levels.

Foggy Scenes Evaluation According to the results in Table 5.7, the model with 60% augmentation shows the most improvement in the mean AP scores as well as in sensitive, yet less frequent object classes (in the TUMTraf dataset), like motorcycles and pedestrians. The results show an improvement of 3.1% in the mean AP value under foggy conditions.

Distance-Based Trends We have studied the effect of fog on objects at different distances from the LiDAR sensor (the detailed results can be seen in the annex in Table A.5). We can observe that the performance improvements vary by distance, with certain augmentation levels optimizing detection at specific ranges. Since fog scatters points mostly in wide ranges, we are interested in the performance results in ranges further away from the sensor (40-50 m and 50-64 m). The results show that the 60% augmentation model performs best at 50-60 m, scoring a 57.48% AP for the VAN class (+26,26% improvement) and 16.17% for the PEDESTRIAN class (the baseline model was not able to detect any pedestrians). The mean AP values for 40-50 m and 50-60 m ranges also show that the 60% model is the most resilient to unseen fog conditions.

Optimal Augmentation Level

According to the results highlighted in Tables 5.7 and 5.8, the 60% augmentation model scores best in the mean AP, mean IoU, and mean recall values as well as at further visibility ranges, which makes it more resilient to unseen adverse weather and more reliable to apply in real life scenarios.



(a) Detection results of the baseline model (top left) and the five augmentation results at foggy scene. The results highlight the 100% augmentation model's performance at detecting cars compared to the other models.

(b) Detection results of the baseline model (top left) and the five augmentation results at a day foggy scene where the baseline model detected none of the objects while the augmentation models could detect the pedestrian in the scene.

Figure 5.9: Detection results of the baseline model and the five augmentation results at a day foggy scene highlighting the improvement of detection results using adverse weather augmentation for training. Blue boxes represent ground truth, and pink boxes represent the model's detections. A video showing the fog-augmented test scenes with detections by the baseline model and the 60% augmented model can be found under the link:https://rb.gy/55ka8j

Table 5.9: Summarized Mean IoU, Precision, and Recall metrics for overall scenes in Experiments 3.1 and 3.2 across various augmentation levels.

Metric	Experiment	Baseline	+20%	+40%	+60%	+80%	+100%
IoU	3.1	**0.5177**	0.5113	0.5176	0.4989	0.5094	0.5141
	3.2	**0.5252**	0.5152	0.5243	0.5012	0.5205	0.5207
Precision	3.1	81.500	83.333	81.5	82.852	82.563	**83.421**
	3.2	81.500	83.542	81.6	82.893	81.765	**84.375**
Recall	3.1	52.713	56.135	54.622	**57.167**	54.232	55.469
	3.2	52.713	55.698	54.447	**56.265**	53.359	54.729

5.2.3 Experiment 3: Augmented Test Dataset with Seen Conditions (Rain)

The following experiment aims to observe the models' performance when evaluated with a seen condition (rain) using rain rates from the training dataset (Experiment 3.1) vs. new unseen rain rates (Experiment 3.2). We summarize the results of Experiments 3.1 and 3.2 in Tables 5.10 and 5.9.

Average Precision (AP) Metrics The evaluation results for the mean average precision show that the 60% augmentation model scores best for both simulated scenarios. The model improves performance by 3.05% for seen rain rates and by 3.39% for unseen rain rates. The qualitative results shown in Figure 5.10 further highlight the positive effect of the augmentation approach on the model's performance.

Mean IoU, Precision, and Recall According to Table 5.9, the mean IoU was highest for both conditions at the baseline model, which means that the augmentation did not improve the spatial accuracy of the model. On the other hand, relevant metrics for the 3D detection task, such as precision and recall, were highest at respectively 100% and 60% augmentation models with improvements of 2.875% and 3.552% for unseen rain rates.

Class-Specific Performance

- **CAR**: The AP improved significantly from the baseline in both experiments. The highest AP was observed in Experiment 3.2 at 80% augmentation (75.10%), suggesting that different rain rates can enhance car detection more effectively.
- **TRUCK and TRAILER**: The 20% augmentation model shows the best performance for larger vehicles like TRUCK and TRAILER while improving the performance of the baseline model for the unseen rain rates simulation by 2.17% for the TRAILER class and 5.88% for the TRUCK class.
- **VAN and BUS**: Both experiments showed the highest AP at 100% augmentation for medium-sized vehicles like buses and vans.
- **PEDESTRIAN**: Both experiments showed steady improvement, with the highest AP at 100% augmentation, Experiment 3.2 (31.71%) outperforming Experiment 3.1 (29.31%), which marks an $\approx$ 10% improvement compared to the baseline model.
- **BICYCLE**: The highest AP was seen at 60% in both experiments, with Experiment 3.2 (59.53%) outperforming Experiment 3.1 (60.18%).

Table 5.10: Summarized Average Precision (AP) metrics for Experiments 3.1 and 3.2.

Class	Experiment	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	3.1	69.78	74.90	74.89	**74.94**	74.90	73.46
	3.2	68.19	73.37	73.38	73.43	**75.10**	73.56
TRUCK	3.1	65.81	**71.61**	65.83	69.68	69.71	69.88
	3.2	63.72	**69.60**	65.76	69.46	67.66	67.89
TRAILER	3.1	75.26	75.49	**77.28**	73.42	75.27	73.35
	3.2	73.33	**75.50**	75.32	75.30	75.19	71.55
VAN	3.1	51.69	53.72	51.80	55.81	53.77	**57.67**
	3.2	53.53	55.64	51.78	55.78	51.86	**57.72**
MOTORCYCLE	3.1	16.00	21.26	22.99	**23.38**	23.34	21.35
	3.2	18.00	25.14	25.25	25.16	**25.31**	23.18
BUS	3.1	86.28	85.31	87.97	84.26	81.83	**88.94**
	3.2	89.40	88.63	90.72	88.69	83.64	**91.68**
PEDESTRIAN	3.1	19.25	21.95	22.84	26.17	25.07	**29.31**
	3.2	22.62	27.59	26.92	29.90	27.84	**31.71**
BICYCLE	3.1	53.08	58.67	48.02	**60.18**	25.65	24.24
	3.2	54.55	59.00	48.18	**59.53**	27.35	33.57
EMERGENCY	3.1	100.00	100.00	100.00	100.00	100.00	100.00
	3.2	100.00	100.00	100.00	100.00	100.00	100.00
OTHER	3.1	86.00	86.00	86.00	86.00	86.00	86.00
	3.2	62.00	62.00	62.00	62.00	62.00	62.00
Mean AP	3.1	62.31	64.89	63.76	**65.38**	61.55	62.42
	3.2	60.53	63.65	61.93	**63.92**	59.59	61.29

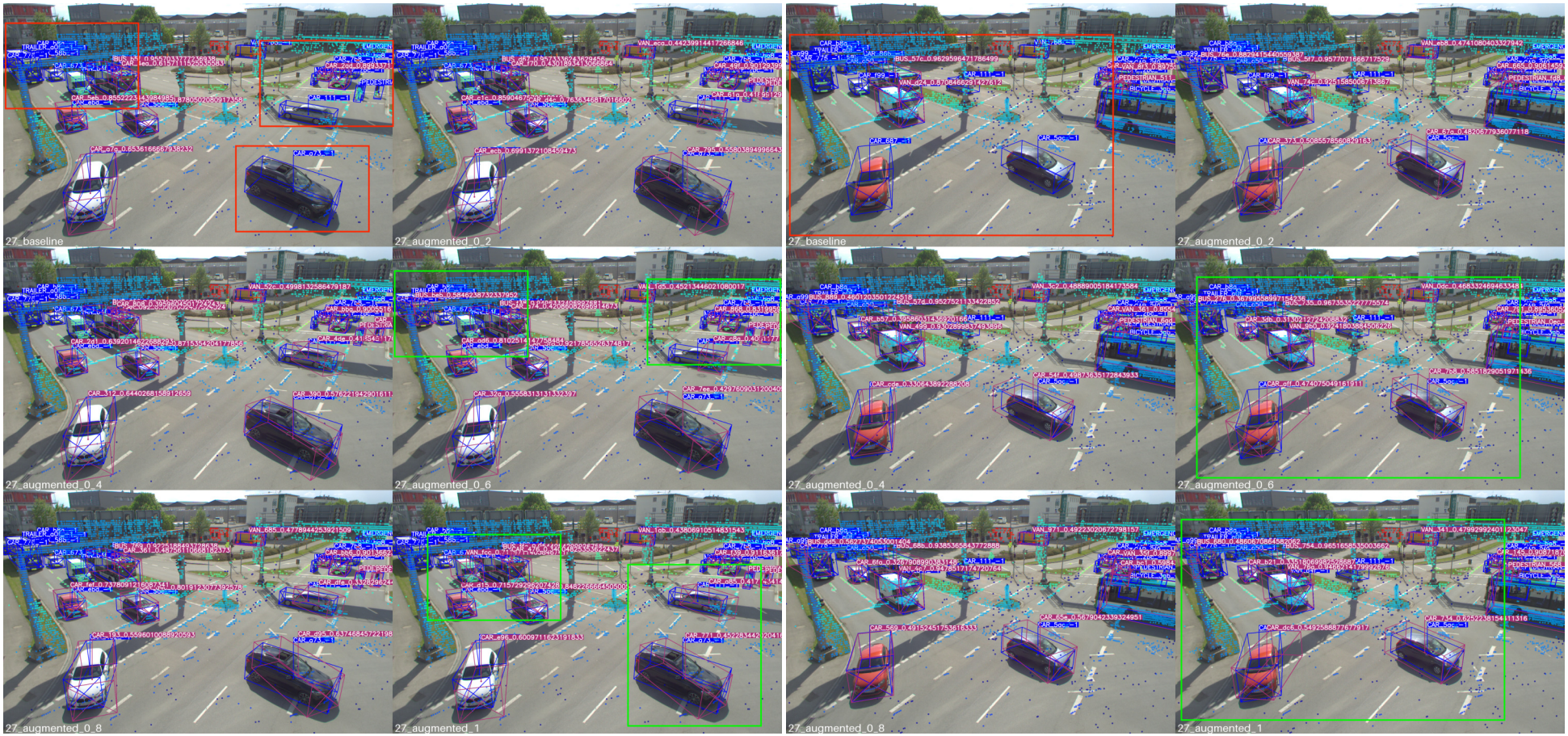
Table 5.11: Runtime performance of the baseline PointPillars model and the five augmented models evaluated on the original test dataset using RTX 3090.

Model	Runtime (ms)	FPS
Baseline	23.57	42.42
+20% augmentation	24.48	40.85
+40% augmentation	24.57	40.70
+60% augmentation	24.39	41.00
+80% augmentation	24.46	40.89
+100% augmentation	24.95	40.08

Optimal Augmentation Model While the rain augmentation did not improve the IoU of the 3D detection model, the mAP results show that the 60% augmentation model performs best on average. Nevertheless, it is worth mentioning that the higher augmentation level (100%) performed consistently best in both experiments for the classes VAN, BUS, and PEDESTRIAN, which makes combining different augmentation models based on the class a promising approach.

5.2.4 Runtime Performance

To evaluate the run time performance of the models, we calculated the Frame per Second (FPS) value for each model when evaluated using the base test set of the TUMTraf-I dataset. The results are noted in Table 5.11 and show that all augmented models have a lower runtime performance than the baseline model. We can observe that the 60% augmentation model reaches 24.39ms/it, which is 0.82ms longer than the baseline model.



(a) Detection results of the baseline model (top left) and the five augmentation results at the rain-simulated scene with rain rates 5.6mm/h (left): same rain rate as in the training dataset. A video showing the rain-augmented (rain rate: 19mm/h) test scenes with detections by the baseline model and the 60% augmented model can be found under the following link: https://drive.google.com/file/d/1L7qKZBjvpmM_myuzJNygVGZuA9K8-38C/view?usp=drive_link

(b) Detection results of the baseline model (top left) and the five augmentation results at the rain-simulated scene with rain rates 25mm/h (left): new unseen rain rate. A video showing the rain-augmented (rain rate: 35mm/h) test scenes with detections by the baseline model and the 60% augmented model can be found under the following link: https://drive.google.com/file/d/1LQJj4e9NNFiauoSwJxQ2jBnUcezQampo/view?usp=drive_link

Figure 5.10: Detection results of the baseline model (top left) and the five augmentation results at the rain-simulated scene with rain rates 5.6mm/h (left) and 25mm/h (right), which is the same rain rate used for training. (The rain effect is only visible on the scattered point cloud). The results highlight the augmentation model's ability to detect objects at further ranges and objects with scattered points that the baseline model could not detect. Blue boxes represent ground truth, and pink boxes represent the model's detections.

5.2.5 mCE and mRR Analysis

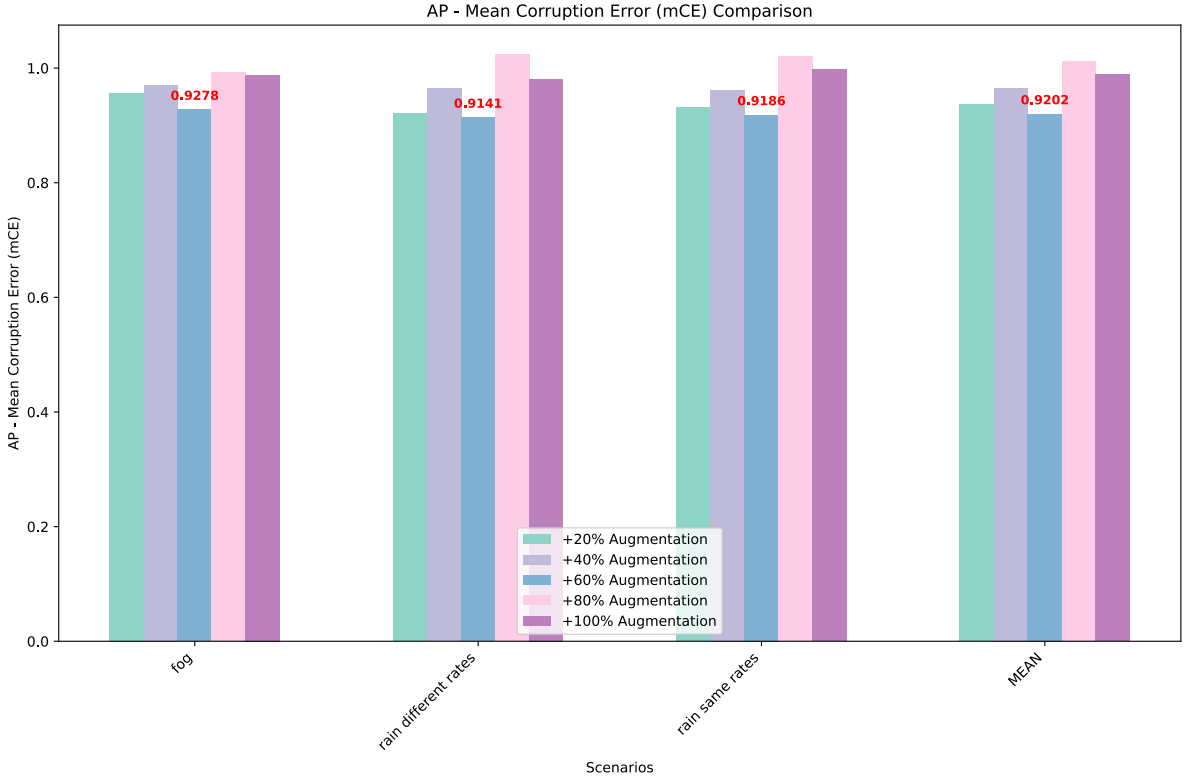
Based on the evaluation of the three experiments, we can conclude that the **60%** augmentation model performs most consistently well when compared to the baseline model and the other augmentation models. However, to have a more solid comparison, we decided to use the mCE and mRR metrics explained in 2 and introduced in [Kon+23] that are designed to calculate the resilience of the models when faced with different corruptions and corruption levels.
To interpret the Mean Corruption Error (mCE) and Mean Resilience Rate (mRR) values, it is important to understand what each metric signifies:

- **Mean Corruption Error (mCE):** This metric quantifies the degradation in model performance under various corruptions. Higher mCE values indicate more significant performance degradation, while lower mCE values indicate better robustness to corruption.
- **Mean Resilience Rate (mRR):** This metric measures the model's resilience or ability to maintain performance across different evaluation scenarios. Higher mRR values indicate better resilience, showing the model performs well under challenging conditions.

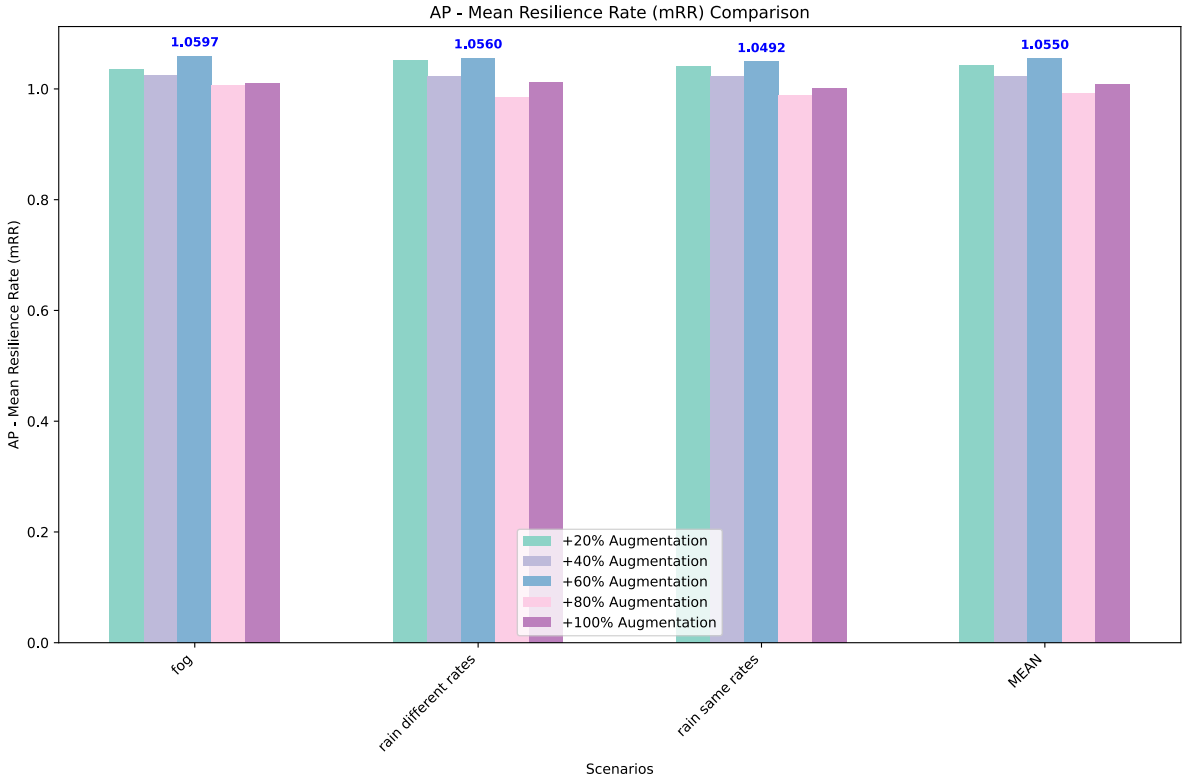
As shown in Figure 5.11, the model with 60% augmentation scores the lowest mCE (0.92) and the highest mRR (1.055), indicating its strong robustness when compared with the other models, which is in total alignment with our conclusion. The model also has consistently low mCE values and high mRR values across all scenarios, highlighting its stability and suitability for deployment in adverse weather conditions in real scenarios.

5.2.6 Conclusions

Based on the results of the 3D detection metrics and the mRR and mCE results shown in Figure 5.11, we can conclude that the 60% augmentation model is the most robust and resilient when evaluated with different adverse weather conditions. However, if we further observe Figure 5.12, we can see that the model performs exceptionally well for the most frequent object class of our dataset (CAR), while its performance for other sensitive classes like BUS, PEDESTRIAN, and BICYCLE leaves room for improvement.

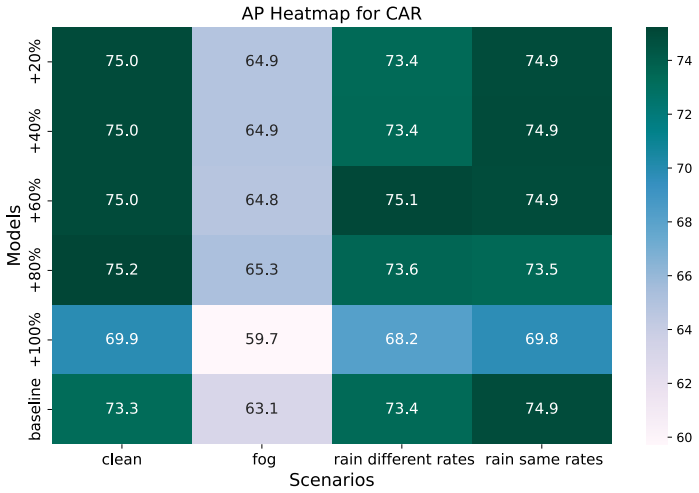


(a) AP - Mean Corruption Error (mCE) Comparison

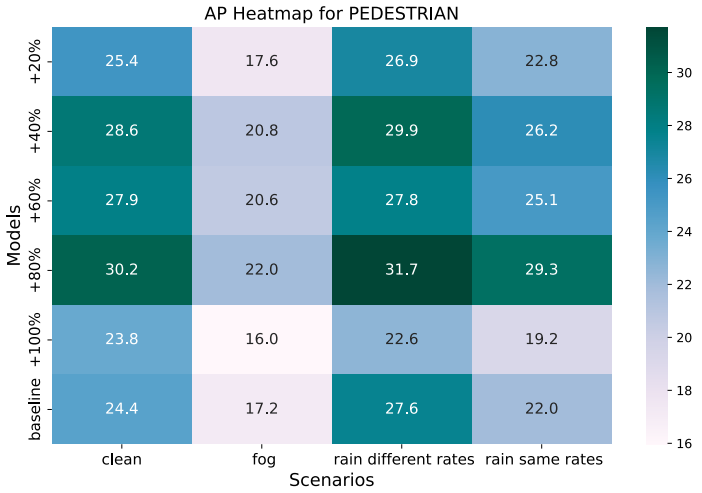


(b) AP - Mean Resilience Rate (mRR) Comparison

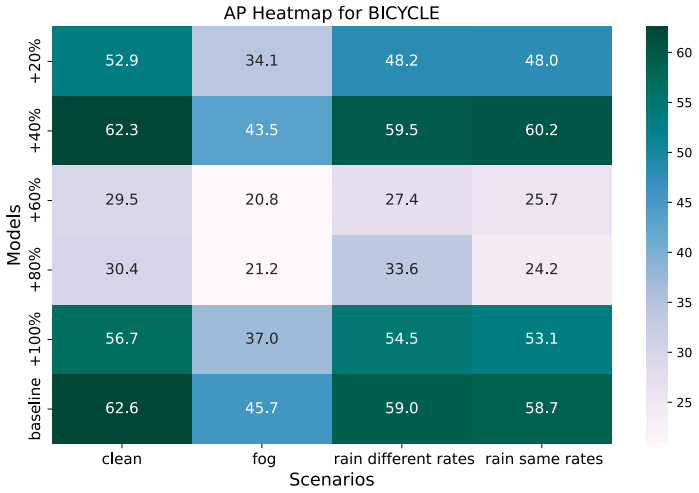
Figure 5.11: Comparison of AP - Mean Corruption Error (mCE) and Mean Resilience Rate (mRR).



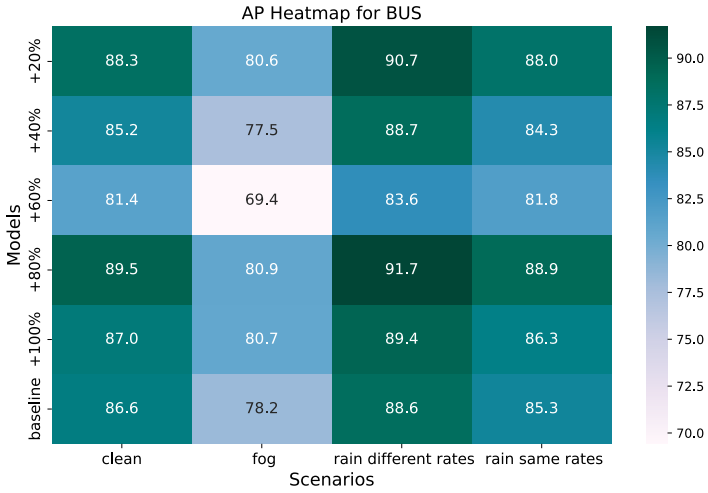
(a) AP Heatmap for CAR



(b) AP Heatmap for PEDESTRIAN



(c) AP Heatmap for BICYCLE



(d) AP Heatmap for BUS

Figure 5.12: Heatmaps representing the average precisions of the different augmentation models and the baseline model for the CAR, PEDESTRIAN, BICYCLE, and BUS classes

5.3 Backwards Point Concatenation

As mentioned in Chapter 4, we follow up on the attempts of [Kem+23a] to improve the 3D detection by using point concatenation, where we test the offset point concatenation approach on our TUMTraf-I dataset. The test dataset we have is composed of frames from four different sequences. Three sequences were captured during the day and dusk, and the rest were captured at night in rainy conditions. Using the test dataset, we will evaluate the concatenation approach with different setups and report the results of both the baseline PointPillars model and the 60% augmented model. We test the approach with and without the filtering method.

In Chapter 4, we have provided the algorithm for this approach that has two hyperparameters as input: the offset, which indicates how many frames will be concatenated to the current frame at time t, and the distance threshold that will define whether a point from a point cloud $X-1$ will be taken into consideration in the concatenation. The algorithm takes two hyperparameters, offset and distance threshold, as input, which we start by finetuning.

5.3.1 Backwards Concatenation without Filtering

The baseline model initially scored an AP value of 42.4% at 0.5 IoU threshold when evaluated on the rainy night scenes from the TUMTraf-I dataset. We will consider this our baseline scenario. We have varied the offset for concatenation from 1 to 10 without using any filtering techniques on the point clouds and re-evaluated the model. The results of the evaluation are shown in Figure 5.13. As seen on the line chart, the performance of the baseline model drops by at least 8.1% in the AP score and 4% in the IoU score when applying concatenation with offset 1. The chart shows that the performance indicators tend to drop more for higher concatenation offset values.

Figure 5.14 shows the concatenation results using offsets 1 and 2 on a point cloud at time t and $t-2$. Both concatenated point clouds show a restored shape of the infrastructure, mainly the bridge and the car underneath the bridge, which is the primary goal of the concatenation approach. However, we can observe the side effects of the concatenation of point clouds without applying any filtering of points on the last image in Figure 5.14a. The point cloud shows an added noise with the shape of a bus that does not appear on the RGB image but appears on the pictures in Figure 5.14b at $t-2$. The last image results from concatenating the point cloud at time t with the two-point clouds at times $t-1$ and $t-2$. This added noise resulting from the raw concatenation of consecutive point clouds explains the significant drop in performance shown in Figure 5.13.

5.3.2 Hyperparameter Tuning

The first iterations of the hyperparameter tuning process we conducted showed that the 3D detection model performs best when setting $offset = 1$. While setting the offset to a value higher than 1 does not result in a drop in the 3D detection model's performance, we did not record any improvement in the evaluation metrics. Therefore, we will set the *offset* parameter to 1 and tune the distance threshold limits. We evaluate the baseline PointPillars model on the 37 rainy night scenes from the TUMTraf-I dataset using distance minimum threshold values in the range of [0.1, 1] and distance maximum threshold values in the range of [0.5, 1]. The results are visualized in Figure 5.16. Figure 5.16 shows an improved performance in the AP value that goes up to 43.36% (+1% compared to the baseline model). The results show that concatenation with *offset* equals 1, minimum distance threshold 0.6, and maximum distance threshold 0.8 scores the best improvement in AP compared to the baseline model.

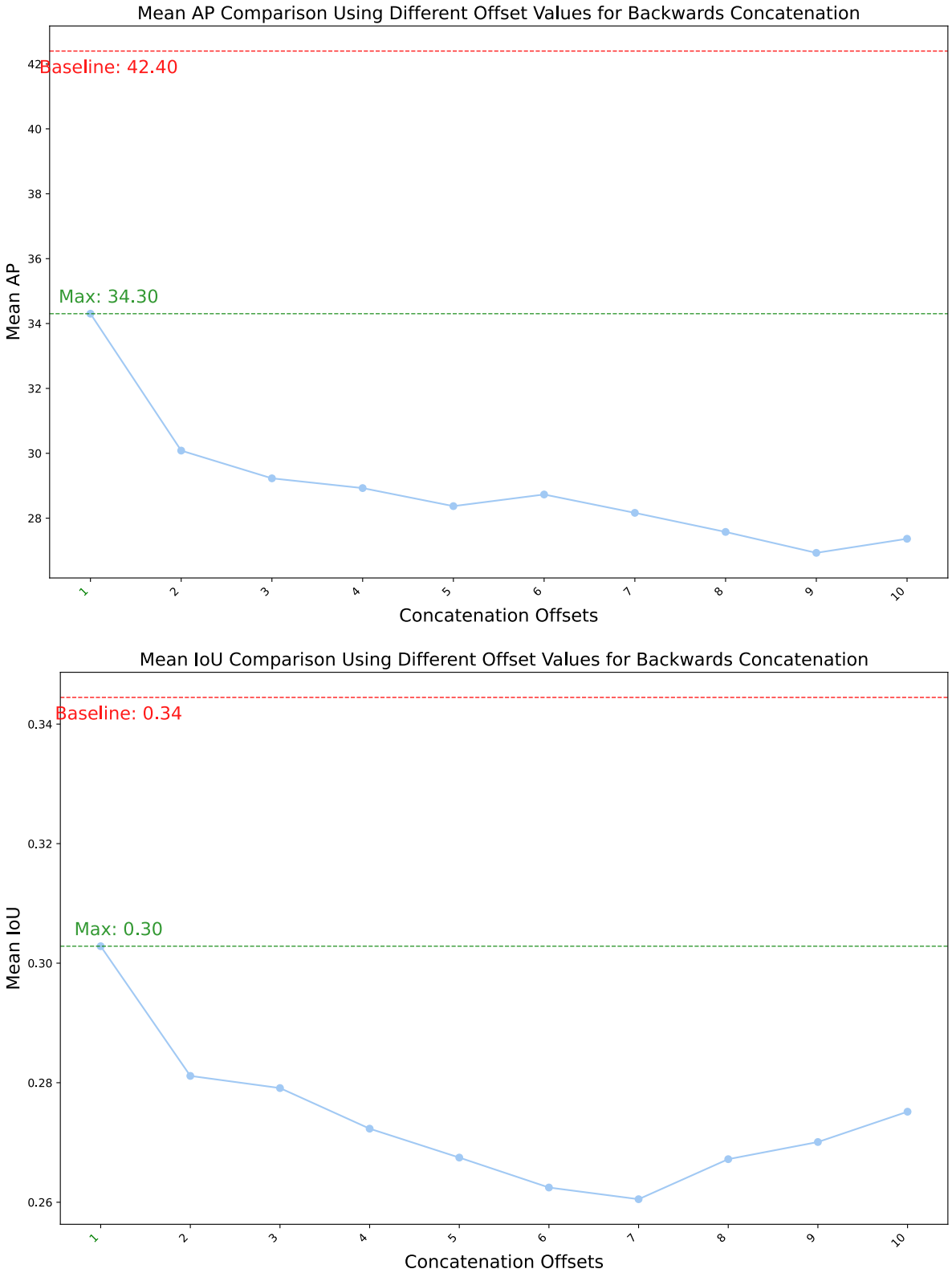
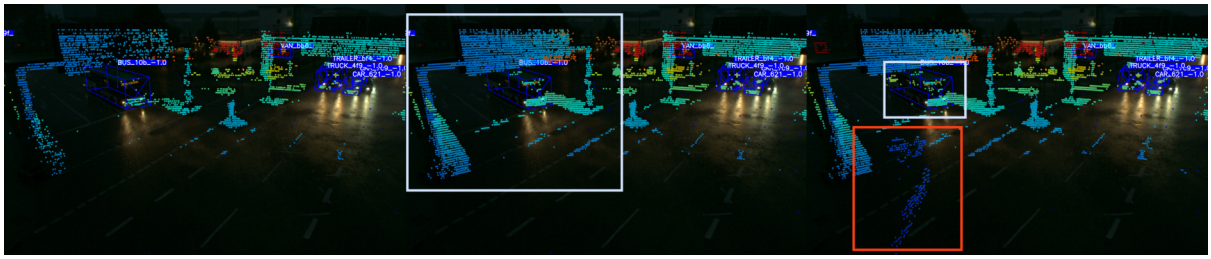
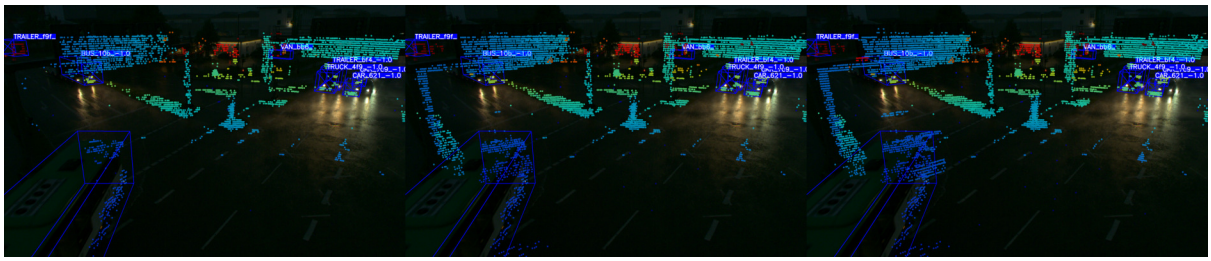


Figure 5.13: Evaluation Results of the baseline PointPillars model using 1 to 10 concatenation offset values. The performance of the baseline model with 0 frame concatenation is represented by the dashed horizontal line.



(a) The observed image at time t.



(b) The observed image at $t-2$

Figure 5.14: Qualitative Results of the baseline PointPillars model using 1 to 10 concatenation offset values. The first image is the original RGB image.

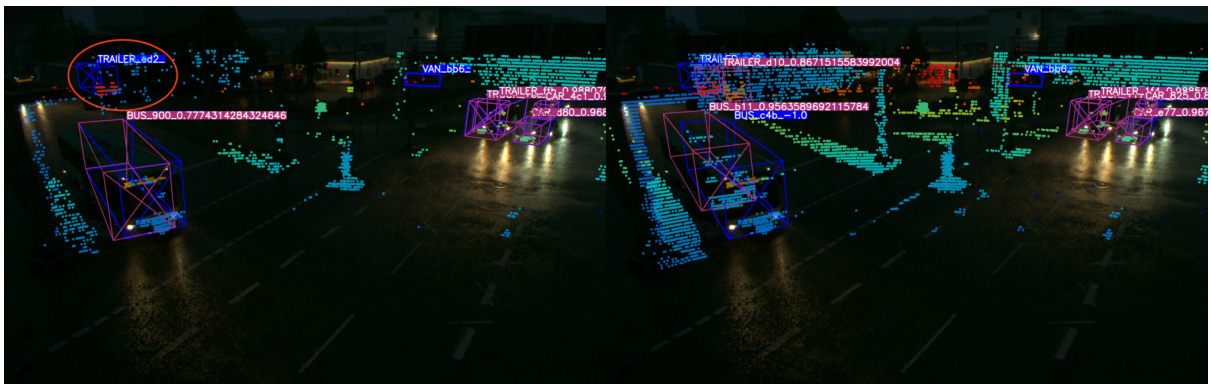


Figure 5.15: The results of the evaluation of the baseline PointPillars once without point cloud concatenation (left) and once with a 1-Frame concatenation (right). Blue boxes represent ground truth, and pink boxes represent the model's detections.

5.3.3 Performance Evaluation

We use the hyperparameter values obtained from the abovementioned finetuning to initialize our algorithm and get the qualitative results visualized in Figure 5.15. The results show the model's ability to recover the points of the trailer positioned far from the LiDAR sensor when using the 1-frame-backward concatenation, and thus its ability to detect the trailer as opposed to the baseline model. Figure 5.15 represents only an example of multiple improvement instances we could track. We further tested the impact of the concatenation on the run-time performance. The inference statistics show that while the baseline model without concatenation reached an FPS of 7.38, the concatenation step reduced the model's runtime performance to 3.96 FPS (almost double). While an increase in the model's FPS is expected, this result encourages further investigation of the approach to find a way to compensate for the latency.

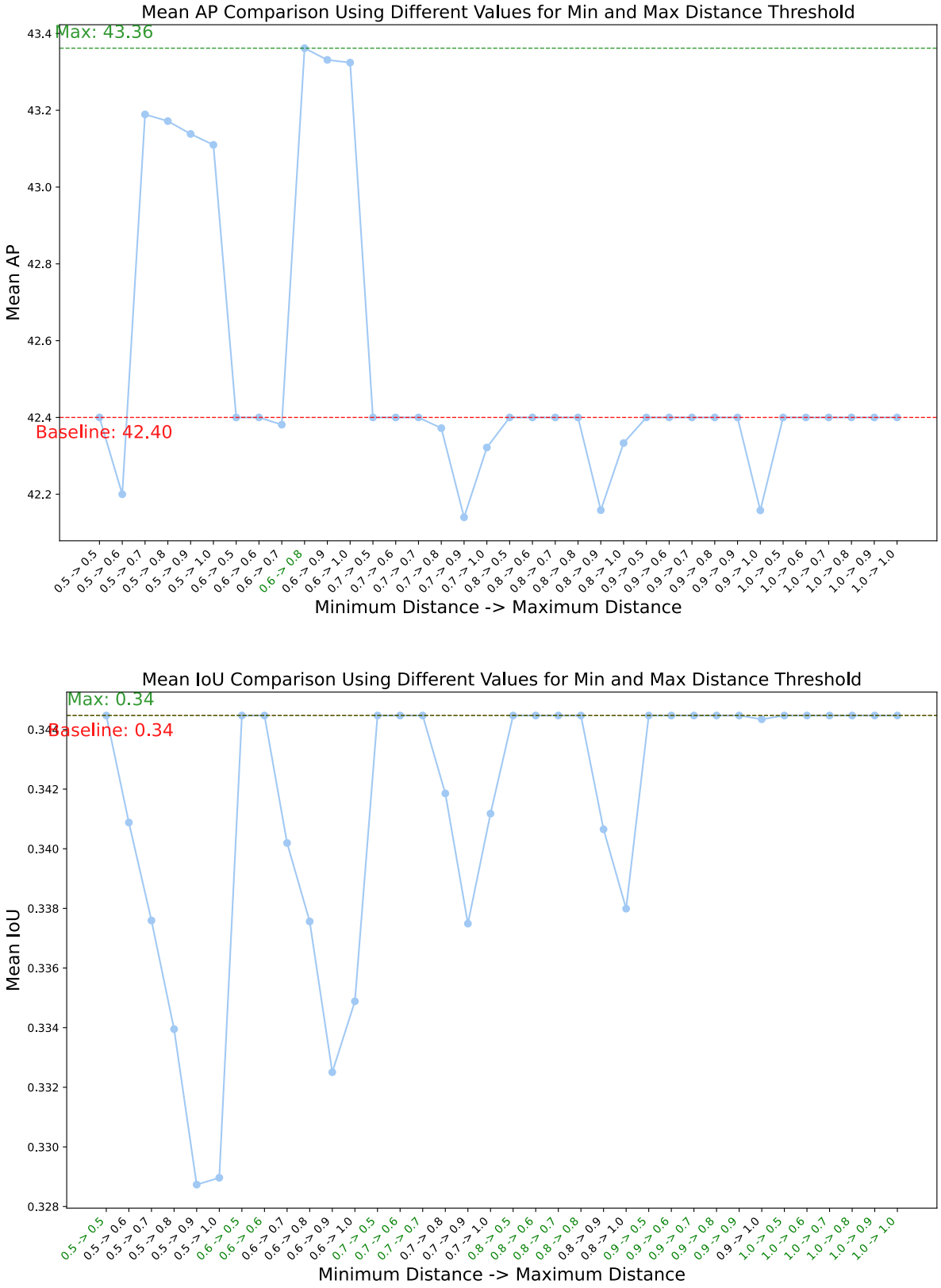


Figure 5.16: Evaluation results of the baseline PointPillars model using minimum and maximum point distance ranging from 0.5 to 1.

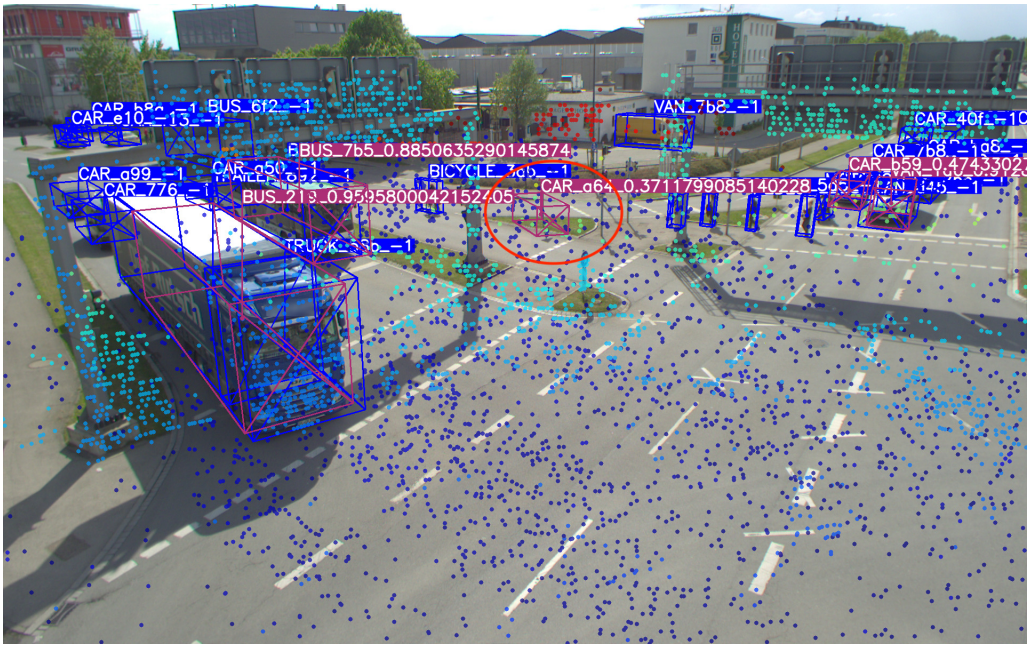
Chapter 6

Conclusion and Future Work

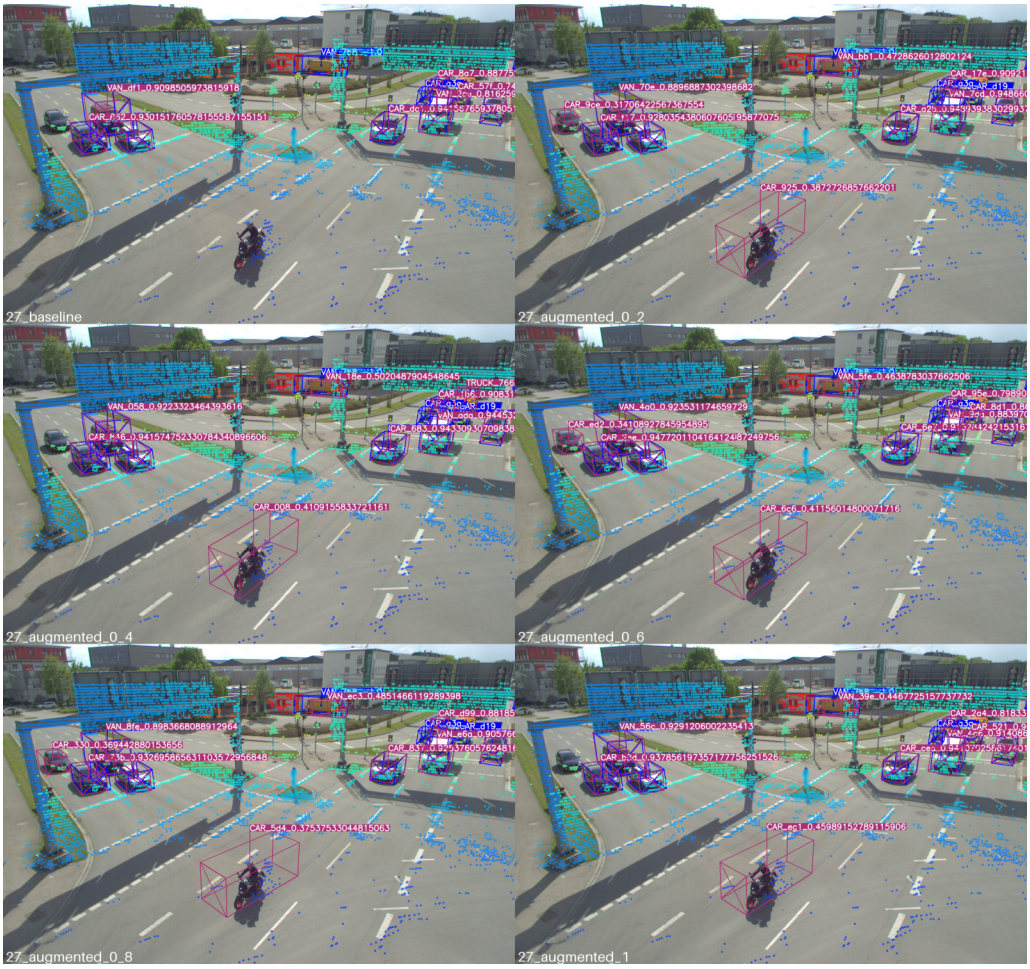
6.1 Conclusion

We conducted a total of 48 experiments using five different augmentation levels of the TUMTraf-I [Zim+23c] dataset where we simulated adverse weather conditions such as fog, moderate rain with rain rates varying from [2,19] mm/h, heavier rain with rain rates varying from [25,45] mm/h, and snow (71 mm/h). The results of the experiments show that augmenting 60% of the original dataset was able to improve the baseline model's performance by $\approx$ 3.5% (mAP) on average when testing on the full original dataset and was able to improve the detection of small and sensitive objects like pedestrians and bicycles by more than 10%, especially under unseen weather conditions like fog. These results validate the augmentation approach's capability to improve the model's resilience to corruption by adverse weather conditions. We validated the approach using more task-related metrics like mCE and mRR, which were designed to compare the model's performance when faced with corruption to its performance on normal weather data. The results showed that the 60% augmentation model scored best on average and consistently through all types of adverse weather conditions we used in our experiments. While we created a more robust and resilient model against adverse weather conditions, the obtained 60% augmented model is still prone to false positives. Both the quantitative (see Table 5.8) and the qualitative results (See Figure 6.1a) show a lower precision for the 60% augmentation level when compared to the other models when evaluated on unseen conditions, like fog. The high rate of false positives results in the prediction of ghost objects. This can cause sudden braking and traffic incidents for autonomous driving.

Since data augmentation is an approach that requires the retraining of the model on a much larger dataset, it is not only time-consuming but also resource-bound. Therefore, we tried another approach: point cloud concatenation at inference time. We added a filtering step and a similarity verification step to the point cloud concatenation approach mentioned in [Kem+23a], and we obtained a slight improvement of $\approx$ 1% compared with the baseline inference results. While this improvement is a testimony of the approach's ability to mitigate detection problems resulting from scattered and sparse point clouds from various corruption types, the approach leaves room for improvement. Due to the considerably small test dataset in the TUMTraf-I dataset, we could only fine-tune the hyper-parameters of the algorithm using 240 instances.



(a) Fog-augmented point cloud showing a false positive detection of traffic shield, classified as a car.



(b) Rain-augmented point cloud showing a misclassification of a motorcycle as a car by all augmented models while remaining undetected by the baseline model.

Figure 6.1: Open issues of augmented models: Misdetections and Misclassifications. Blue boxes represent ground truth, and pink boxes represent the model's detections.

6.2 Future Work

6.2.1 Increasing Dataset Size

In the future, we plan to use a more extensive test dataset with varied conditions. Since our dataset only shows one type of adverse weather conditions, rain, expanding the dataset will allow a more comprehensive evaluation of the model's performance across a broader range of scenarios, enhancing the robustness and generalisability of our results. Besides, the dataset statistics show a significantly lower number of labels in some object classes, such as MOTORCYCLE, EMERGENCY, and BICYCLE. While the 60% augmentation model performed better than the baseline model, the reduced number of detected objects in these classes can lead to faulty classifications despite correct detections. Figure 6.1b shows such a scenario, where a motorcycle was detected by the augmented models, as opposed to the baseline model. However, the models mistakenly classified the motorcycle as a car. We believe such a problem can be mitigated by enlarging the dataset with more instances of the less frequent object classes.

6.2.2 Improving the Spacial Accuracy

While the 60% augmentation model performs best on the 3D prediction task, the lower IoU values recorded across the experiments we conducted show that the model's spatial accuracy still needs improvement. The model has a higher capability of detecting existing objects; however, the qualitative results show that the detected bounding boxes are misaligned with the ground truth bounding boxes. The authors of [ZGK22] propose the DASE-ProPillars Model that introduces a Multi-task Detection Head to the PointPillars model to improve the IoU detection and the classification score of the PointPillars model, which is a promising approach that we would like to try in the future in combination with our augmentation approach.

6.2.3 Improving the Data Augmentation Pipeline

We aim to extend our data augmentation approach to cover more adverse weather conditions. This includes simulating hail and extreme gas exhaust from cold weather, studied in [Has+17] and [Pir+22], and mixed weather scenarios to push the limits of the model's robustness. We plan to experiment with different severity levels for rain, fog, and snow to understand how varying intensities affect model performance and to fine-tune the model accordingly. Another promising direction is to implement a weather classification approach. In this approach, we would first detect the weather conditions in a point cloud and then use the best-performing model for those adverse weather conditions. This method involves training separate models for each weather condition and developing a classifier to identify the current weather accurately from the point cloud data. Based on the detected weather, the dynamic model selection would ensure optimal performance across diverse scenarios. This could help reduce false positives and improve detection reliability in real-world applications. The authors of [Ket+24] propose a method for classifying weather conditions, precisely precipitation rates, that we would like to investigate further in combination with our approaches.

6.2.4 Improving the Point Concatenation Approach

To improve the point cloud concatenation at inference time, we propose investigating the feature concatenation approach, as described by [Kem+23a]. Additionally, [Xie+23] presents a

feature fusion technique using a multi-frame detection head for the PointPillars model, reporting an mAP improvement of 8.5%. Therefore, we would like to explore encoding temporal information from multiple frames as features and test its impact on the model's robustness. A further challenge introduced by the concatenation approach is the potential concatenation of adverse weather points. To avoid the augmentation of noise in the dataset, we would like to enhance the filtering layer of the concatenation approach with a filter of point clouds resulting from adverse weather, such as gas exhaust and snowflakes. Previous work such as [Pir+24] and [Qi+24] propose promising approaches for such a filter based on semantic segmentation, which we would like to test and evaluate with our concatenation approach.

6.2.5 Improving Runtime Performance

Evaluating the inference time of the concatenation-improved inference pipeline and the data augmentation approach shows a decrease in the model's runtime performance. We would like to explore methods to improve efficiency and keep the approach applicable in real-life scenarios. We suggest integrating the CUDA-PointPillars [Blo21] approach introduced by NVIDIA to mitigate the latency introduced by point cloud concatenation. CUDA-PointPillars significantly speeds up point cloud processing using GPU acceleration, helping maintain high detection accuracy without compromising speed. The authors of [Zim+22] report an increase of $\approx 20Hz$ in inference time when using *TesorRT*, the engine used in CUDA-PointPillars [Blo21].

6.2.6 Use of Generative Models

In an attempt to use the high performance of generative AI models in the data augmentation approach, we have tested the results of data augmentation using commonly used generative models. Yet, we did not come to promising results. In contrast, [Zha+24b] reached an increase of 2.46% in 3D average precision by introducing a fusion technique of segmentation maps and raw point cloud data that serves as a preprocessing step to the data fed to the image-to-image translation model CycleGAN [Zhu+17]. We would like to investigate this approach further using the TUMTraf-I point cloud dataset.

Appendix A

Appendix 1

A.1 Experiments and Results

This section presents the extended results from our experiments in Chapter 5. The experiments are as follows:

- **Experiment 1:** We run the evaluation pipeline on the baseline model and the five augmented models using the original test dataset from TUMTraf-I [Zim+23c]. We separated the test dataset into clear weather scenes and rainy night scenes. The results of the experiments are reported in Tables A.1, A.2, A.3, and A.4.

- **Experiment 2:** We run the evaluation pipeline on the baseline model and the five augmented models using the original test dataset from TUMTraf-I [Zim+23c] with additionally 30% of the point clouds augmented with fog effects. We separated the test dataset into clear weather scenes and foggy scenes. The results of the experiments are reported in Table A.5.

- **Experiment 3:** We run the evaluation pipeline on the baseline model and the five augmented models using the original test dataset from TUMTraf-I [Zim+23c] with additionally 80% of the point clouds augmented with rainy effects. We created two test datasets using rain rates from the same rain samples as the training dataset and with different ones and reported the results of the experiments in Table A.7, A.6, A.8, and A.9.

A.1.1 Experiment 1: Data augmentation - Full original Test Dataset

Table A.1: The Average Precision (AP) metrics for various augmentation levels across all scenes. It includes AP for each object class (CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY, OTHER) at different distances (0-40m, 40-50m, 50-64m).

Class	Distance	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	Overall	69.86	73.30	74.99	74.98	74.97	**75.24**
	0-40m	68.97	72.50	74.22	74.22	73.90	**74.28**
	40-50m	77.45	77.03	77.13	**79.11**	77.28	77.42
	50-64m	68.62	**72.18**	69.87	66.30	72.14	68.53
TRUCK	Overall	63.75	69.50	65.71	**71.40**	67.67	69.79
	0-40m	67.81	75.52	69.68	**77.31**	75.63	75.75
	40-50m	68.00	**72.00**	68.00	66.00	68.00	68.00
	50-64m	46.97	49.57	49.85	49.79	47.41	**49.89**
TRAILER	Overall	75.26	75.64	**77.21**	73.39	75.21	73.59
	0-40m	89.40	89.74	**91.74**	87.21	91.27	89.67
	40-50m	**39.51**	31.45	32.85	35.60	29.04	35.37
	50-64m	59.83	**63.92**	63.55	61.87	61.84	59.86
VAN	Overall	51.67	53.49	53.68	**57.63**	51.79	53.63
	0-40m	51.66	51.85	51.92	**53.85**	51.65	53.55
	40-50m	55.88	**57.88**	55.98	55.90	56.00	55.89
	50-64m	27.88	30.11	45.30	**63.34**	30.67	45.57
MOTORCYCLE	Overall	22.00	30.91	29.35	31.42	**31.52**	26.98
	0-40m	74.00	88.00	87.72	**92.00**	88.00	87.49
	40-50m	4.00	8.79	11.83	**12.30**	12.20	7.25
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
BUS	Overall	87.02	86.57	88.30	85.19	81.38	**89.55**
	0-40m	95.97	99.47	**99.88**	93.28	98.47	99.10
	40-50m	39.83	43.66	**47.67**	41.39	38.08	46.94
	50-64m	**91.32**	84.81	87.07	87.88	84.80	89.55
PEDESTRIAN	Overall	23.82	24.45	25.38	28.58	27.86	**30.18**
	0-40m	75.33	75.73	72.76	72.13	**83.86**	79.94
	40-50m	14.81	15.62	14.15	**19.69**	15.85	18.37
	50-64m	0.00	15.50	11.95	**20.00**	13.78	13.78
BICYCLE	Overall	56.69	**62.58**	52.94	62.33	29.54	30.40
	0-40m	60.85	65.07	62.01	**70.31**	29.54	31.90
	40-50m	22.39	**30.39**	18.47	17.33	17.79	18.09
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
EMERGENCY	Overall	100.00	100.00	100.00	100.00	100.00	100.00
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	100.00	100.00	100.00	100.00	100.00	100.00
OTHER	Overall	80.00	80.00	80.00	80.00	80.00	80.00
	0-40m	80.00	80.00	80.00	80.00	80.00	80.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
Mean AP	Overall	63.01	65.64	64.76	**66.49**	61.99	62.94
	0-40m	66.40	**69.79**	68.99	69.62	67.23	67.17
	40-50m	32.19	**33.68**	32.61	32.73	31.42	32.73
	50-64m	39.46	41.61	42.76	**44.92**	41.06	42.72

Table A.2: Performance (AP) metrics for night scenes, detailing performance for each class at different distances under night conditions.

Class	Metric	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	Overall	70.00	82.64	82.52	80.90	**82.84**	82.77
	0-40m	56.00	**89.97**	88.38	87.90	87.85	86.72
	40-50m	**86.00**	80.00	80.00	80.00	80.00	84.00
	50-64m	60.00	62.00	**65.90**	56.00	62.00	62.00
TRUCK	Overall	**100.00**	**100.00**	**100.00**	98.00	**100.00**	**100.00**
	0-40m	100.00	100.00	100.00	100.00	100.00	100.00
	40-50m	**100.00**	**100.00**	**100.00**	96.00	**100.00**	**100.00**
	50-64m	100.00	100.00	100.00	100.00	100.00	100.00
TRAILER	Overall	88.00	88.00	88.00	**90.00**	86.00	86.00
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	**72.00**	26.00	44.00	44.00	18.00	44.00
	50-64m	**90.00**	**90.00**	88.00	**90.00**	**90.00**	**90.00**
VAN	Overall	66.00	66.00	66.00	66.00	66.00	66.00
	0-40m	84.00	84.00	84.00	84.00	84.00	84.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
MOTORCYCLE	Overall	0.00	0.00	0.00	0.00	0.00	0.00
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
BUS	Overall	**100.00**	**100.00**	**100.00**	96.00	96.00	**100.00**
	0-40m	100.00	100.00	100.00	100.00	100.00	100.00
	40-50m	**100.00**	**100.00**	**100.00**	82.00	82.00	**100.00**
	50-64m	100.00	100.00	100.00	100.00	100.00	100.00
PEDESTRIAN	Overall	0.00	0.00	0.00	0.00	0.00	0.00
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
BICYCLE	Overall	0.00	0.00	0.00	0.00	0.00	0.00
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
EMERGENCY	Overall	0.00	0.00	0.00	0.00	0.00	0.00
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
OTHER	Overall	0.00	0.00	0.00	0.00	0.00	0.00
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
Mean AP	Overall	42.40	**43.66**	43.65	43.09	43.08	43.48
	0-40m	34.00	**37.40**	37.24	37.19	37.18	37.07
	40-50m	**35.80**	30.60	32.40	30.20	28.00	32.80
	50-64m	35.00	35.20	**35.39**	34.60	35.20	35.20

Table A.3: IoU, Precision, and Recall metrics for overall scenes, comparing spatial accuracy across augmentation levels for each class at various distances

Class	Metric	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	IoU	**0.7245**	0.7076	0.7084	0.7158	0.7068	0.7115
	Precision	70.456	73.821	75.482	75.472	75.457	**75.722**
	Recall	52.091	58.105	60.152	60.077	**60.329**	60.077
TRUCK	IoU	0.7056	0.7018	**0.7173**	0.7008	0.7023	0.7002
	Precision	64.458	70.103	66.386	**71.958**	68.302	70.385
	Recall	41.474	48.999	43.617	**51.075**	46.725	48.623
TRAILER	IoU	0.6975	0.6939	0.6820	0.6880	0.6884	**0.7044**
	Precision	75.748	76.119	**77.658**	73.915	75.701	74.106
	Recall	56.858	57.106	**59.786**	54.679	57.501	55.207
VAN	IoU	**0.7374**	0.6818	0.6923	0.6999	0.6953	0.6800
	Precision	52.622	54.403	**54.589**	58.457	52.733	54.538
	Recall	27.152	28.701	**28.810**	33.081	27.257	29.245
MOTORCYCLE	IoU	**0.4211**	0.3559	0.3725	0.3657	0.3365	0.3920
	Precision	23.529	**32.266**	30.739	32.760	32.867	28.416
	Recall	4.874	**10.238**	9.412	10.476	10.476	7.983
BUS	IoU	**0.579**	0.5446	0.5513	0.5056	0.5073	0.5042
	Precision	87.272	86.829	**88.530**	85.481	81.746	89.752
	Recall	**81.176**	79.752	80.702	81.176	76.625	83.911
PEDESTRIAN	IoU	0.3491	0.4348	0.4664	0.4632	0.4784	**0.4854**
	Precision	25.314	25.930	26.843	**29.981**	29.274	31.551
	Recall	14.557	**17.762**	17.762	19.393	17.850	17.937
BICYCLE	IoU	0.3307	**0.3787**	0.3425	0.3353	0.3746	0.3321
	Precision	57.536	**63.311**	53.862	63.069	30.919	31.769
	Recall	48.956	**57.970**	52.657	60.721	29.222	32.100
EMERGENCY	IoU	**0.3856**	0.3124	0.3476	0.2445	0.3133	0.2848
	Precision	**100.000**	100.000	100.000	100.000	100.000	100.000
	Recall	**100.000**	100.000	100.000	100.000	100.000	100.000
OTHER	IoU	0.3268	0.3765	**0.4136**	0.3563	0.3312	0.3786
	Precision	**80.392**	80.392	80.392	80.392	80.392	80.392
	Recall	**64.314**	64.314	64.314	64.314	64.314	64.314
Mean IoU	IoU	0.5257	0.5188	**0.5294**	0.5075	0.5134	0.5173
	Precision	81.500	**83.542**	82.917	82.549	83.370	82.550
	Recall	53.292	**56.718**	56.299	57.720	54.810	55.661

Table A.4: IoU, Precision, and Recall metrics for night scenes, comparing spatial accuracy across augmentation levels for each class at various distances

Class	Metric	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	IoU	**0.8194**	0.7387	0.7544	0.7709	0.7362	0.7467
	Precision	70.588	82.976	82.858	81.270	**83.181**	83.111
	Recall	48.789	70.055	70.675	66.609	70.055	**70.675**
TRUCK	IoU	0.7149	0.7748	**0.7756**	0.7695	0.7189	0.6643
	Precision	**100.000**	100.000	100.000	98.039	100.000	100.000
	Recall	**100.000**	100.000	100.000	96.000	100.000	100.000
TRAILER	IoU	0.6501	**0.6779**	0.6600	0.6628	0.6601	0.6602
	Precision	**88.235**	88.235	88.235	90.196	86.275	86.275
	Recall	77.206	77.206	77.206	**80.532**	73.950	73.950
VAN	IoU	0.4167	0.4202	**0.4414**	0.4396	0.4236	0.4153
	Precision	**66.667**	66.667	66.667	66.667	66.667	66.667
	Recall	**44.444**	44.444	44.444	44.444	44.444	44.444
MOTORCYCLE	IoU	0.00	0.00	0.00	0.00	0.00	0.00
	Precision	0.00	0.00	0.00	0.00	0.00	0.00
	Recall	0.00	0.00	0.00	0.00	0.00	0.00
BUS	IoU	0.8436	0.8270	0.8252	**0.8507**	0.8383	0.8230
	Precision	**100.000**	100.000	100.000	96.078	96.078	100.000
	Recall	**100.000**	100.000	100.000	92.383	92.383	100.000
PEDESTRIAN	IoU	0.00	0.00	0.00	0.00	0.00	0.00
	Precision	0.00	0.00	0.00	0.00	0.00	0.00
	Recall	0.00	0.00	0.00	0.00	0.00	0.00
BICYCLE	IoU	0.00	0.00	0.00	0.00	0.00	0.00
	Precision	0.00	0.00	0.00	0.00	0.00	0.00
	Recall	0.00	0.00	0.00	0.00	0.00	0.00
EMERGENCY	IoU	0.00	0.00	0.00	0.00	0.00	0.00
	Precision	0.00	0.00	0.00	0.00	0.00	0.00
	Recall	0.00	0.00	0.00	0.00	0.00	0.00
OTHER	IoU	0.00	0.00	0.00	0.00	0.00	0.00
	Precision	0.00	0.00	0.00	0.00	0.00	0.00
	Recall	0.00	0.00	0.00	0.00	0.00	0.00
Mean IoU	IoU	0.3445	0.3439	0.3457	**0.3494**	0.3377	0.3310
	Precision	**42.500**	42.500	42.500	42.500	42.500	42.500
	Recall	**37.107**	37.094	38.057	37.067	36.409	37.761

A.1.2 Experiment 2: Data augmentation - Full original Test Dataset with Fog augmentation

Table A.5: Performance (AP) metrics for fog-augmented scenes, detailing performance for each class at different distances under foggy conditions.

Class	Metric	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	Overall	63.73	67.05	68.80	68.86	68.71	**69.06**
	0-40m	63.12	68.20	**70.09**	70.01	69.59	70.07
	40-50m	71.25	70.90	70.98	**72.98**	71.14	71.28
	50-64m	54.52	57.80	55.80	54.10	**57.94**	56.33
TRUCK	Overall	59.76	65.37	63.61	**67.33**	65.59	65.79
	0-40m	63.81	69.39	67.55	**73.20**	71.63	71.78
	40-50m	58.00	**65.59**	63.95	59.94	62.00	62.00
	50-64m	40.98	47.32	47.48	47.32	**48.72**	47.61
TRAILER	Overall	69.31	69.52	**71.31**	67.38	69.23	69.52
	0-40m	87.03	83.61	**89.72**	81.26	87.08	85.56
	40-50m	**33.55**	25.58	27.10	29.23	25.30	29.60
	50-64m	51.75	**55.93**	55.45	53.78	53.72	53.75
VAN	Overall	47.43	47.40	47.51	**53.49**	49.21	51.24
	0-40m	49.29	49.46	47.69	**51.46**	47.40	49.41
	40-50m	49.63	47.80	47.80	**53.77**	51.70	51.77
	50-64m	31.22	33.33	44.19	**57.48**	34.21	46.02
MOTORCYCLE	Overall	18.00	25.13	23.43	25.48	**25.53**	22.83
	0-40m	64.00	76.00	75.68	75.88	76.00	**77.17**
	40-50m	4.00	7.13	9.83	**10.42**	10.15	5.50
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
BUS	Overall	84.69	80.78	**85.75**	81.82	73.79	84.65
	0-40m	90.57	93.84	**98.75**	88.11	93.67	94.48
	40-50m	39.00	40.33	**48.21**	43.58	37.81	45.86
	50-64m	**88.67**	82.45	84.05	85.18	76.87	84.43
PEDESTRIAN	Overall	18.33	20.52	21.27	23.25	**23.43**	**25.18**
	0-40m	59.90	68.40	63.88	63.56	**74.77**	69.10
	40-50m	11.94	12.52	11.40	**16.17**	13.14	14.94
	50-64m	0.00	12.83	10.33	**16.67**	11.56	11.69
BICYCLE	Overall	45.28	**52.96**	44.19	50.66	23.94	24.36
	0-40m	48.61	54.73	49.22	**55.67**	25.20	24.41
	40-50m	19.66	**26.86**	16.19	15.33	15.78	16.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
EMERGENCY	Overall	100.00	100.00	100.00	100.00	100.00	100.00
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	100.00	100.00	100.00	100.00	100.00	100.00
OTHER	Overall	70.00	70.00	70.00	70.00	70.00	70.00
	0-40m	70.00	70.00	70.00	70.00	70.00	70.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
Mean AP	Overall	57.65	59.87	59.59	**60.83**	56.94	58.26
	0-40m	59.63	**63.36**	63.26	62.91	61.53	61.20
	40-50m	28.70	29.67	29.55	**30.14**	28.70	29.70
	50-64m	36.71	38.97	39.73	**41.45**	38.30	39.98

A.1.3 Experiment 3: Data augmentation - Full original Test Dataset with Seen Conditions (Rain)

Experiment 3.1: Augmented Test Dataset with Seen Conditions (Rain)

Table A.6: IoU, Precision, and Recall metrics for Experiment 3.1, comparing spatial accuracy across augmentation levels for each class at various distances.

Class	Metric	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	IoU	**0.718**	0.703	0.709	0.712	0.710	0.708
	Precision	70.371	**75.397**	75.382	**75.430**	75.393	73.978
	Recall	51.706	59.843	60.039	**60.335**	60.291	58.116
TRUCK	IoU	0.699	0.692	**0.707**	0.698	0.698	0.705
	Precision	66.483	**72.168**	66.501	70.272	70.308	70.476
	Recall	42.797	**51.545**	43.543	49.333	48.276	48.375
TRAILER	IoU	0.695	0.675	0.677	0.683	0.688	**0.696**
	Precision	75.748	75.973	**77.729**	73.936	75.755	73.876
	Recall	57.092	56.679	**60.152**	55.125	57.198	54.207
VAN	IoU	**0.723**	0.672	0.677	0.690	0.677	0.674
	Precision	52.643	54.632	52.748	56.679	54.680	**58.502**
	Recall	27.255	28.937	27.309	31.306	28.601	**33.107**
MOTORCYCLE	IoU	**0.422**	0.379	0.367	0.362	0.342	0.391
	Precision	17.647	22.808	24.505	**24.883**	24.848	22.893
	Recall	2.783	4.977	5.686	**6.078**	6.078	4.977
BUS	IoU	**0.550**	0.509	0.549	0.491	0.476	0.468
	Precision	86.549	85.596	88.206	84.565	82.182	**89.166**
	Recall	80.201	79.714	80.201	80.689	76.312	**83.439**
PEDESTRIAN	IoU	0.356	0.432	0.462	0.466	0.479	**0.484**
	Precision	20.833	23.483	24.358	27.622	26.544	**30.704**
	Recall	11.827	14.555	14.680	**17.474**	15.807	17.566
BICYCLE	IoU	0.308	**0.372**	0.314	0.300	0.331	0.297
	Precision	53.998	**59.486**	49.036	60.964	27.117	25.730
	Recall	44.258	**54.474**	43.697	57.192	25.276	25.704
EMERGENCY	IoU	**0.386**	0.312	0.332	0.238	0.339	0.380
	Precision	**100.000**	**100.000**	**100.000**	**100.000**	**100.000**	**100.000**
	Recall	**100.000**	**100.000**	**100.000**	**100.000**	**100.000**	**100.000**
OTHER	IoU	0.319	0.368	**0.383**	0.350	0.355	0.356
	Precision	86.275	86.275	86.275	86.275	86.275	86.275
	Recall	75.490	75.490	75.490	75.490	75.490	75.490
Mean IoU	IoU	**0.518**	0.511	**0.518**	0.499	0.509	0.514
	Precision	81.500	83.333	0.000	82.852	82.562	**83.421**
	Recall	52.713	56.135	54.622	**57.167**	54.232	55.469

Table A.7: the Average Precision (AP) metrics for various augmentation levels across all scenes in Experiment 3.1. It includes AP for each object class (CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY, OTHER) at different distances (0-40 m, 40-50 m, 50-64 m).

Class	Distance	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	Overall	69.78	**74.90**	74.89	**74.94**	74.90	73.46
	0-40m	67.05	72.37	74.04	**74.14**	73.83	72.52
	40-50m	75.32	76.88	76.89	**77.15**	76.97	78.93
	50-64m	70.67	72.21	71.99	68.68	**74.09**	72.27
TRUCK	Overall	65.81	**71.61**	65.83	69.68	69.71	69.88
	0-40m	69.87	77.62	69.81	**79.43**	75.68	75.87
	40-50m	68.00	**72.00**	68.00	64.00	68.00	68.00
	50-64m	45.01	47.46	47.90	47.74	45.46	**47.94**
TRAILER	Overall	75.26	75.49	**77.28**	73.41	75.27	73.35
	0-40m	89.24	87.65	**91.79**	87.23	91.38	85.52
	40-50m	37.76	35.47	35.27	37.56	29.61	**39.48**
	50-64m	59.80	59.95	**63.56**	59.73	61.80	59.63
VAN	Overall	51.69	53.72	51.80	55.81	53.77	**57.67**
	0-40m	51.69	53.89	51.93	53.90	53.80	**57.82**
	40-50m	**57.76**	55.89	55.99	55.97	55.99	55.91
	50-64m	23.90	33.30	34.97	**56.77**	30.72	51.77
MOTORCYCLE	Overall	16.00	21.26	22.99	**23.38**	23.34	21.35
	0-40m	68.00	76.00	83.02	87.72	**88.00**	85.58
	40-50m	2.00	4.98	**5.02**	6.12	5.20	3.50
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
BUS	Overall	86.27	85.31	87.97	84.26	81.82	**88.94**
	0-40m	92.99	98.65	**99.82**	92.25	97.80	98.09
	40-50m	31.16	39.67	36.29	37.73	36.74	**42.15**
	50-64m	**87.31**	83.42	86.50	85.67	82.42	87.23
PEDESTRIAN	Overall	19.25	21.95	22.85	26.17	25.08	**29.31**
	0-40m	69.85	72.86	69.72	74.75	79.10	**83.99**
	40-50m	12.51	14.06	12.91	**18.11**	14.71	18.10
	50-64m	0.00	7.29	5.26	**9.43**	6.56	6.48
BICYCLE	Overall	53.08	58.68	48.02	**60.18**	25.66	24.25
	0-40m	56.04	60.82	57.05	**68.35**	27.11	26.69
	40-50m	26.35	**37.16**	14.59	18.41	19.51	19.15
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
EMERGENCY	Overall	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
OTHER	Overall	86.00	86.00	86.00	86.00	86.00	86.00
	0-40m	86.00	86.00	86.00	86.00	86.00	86.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
Mean AP	Overall	62.32	64.89	63.76	**65.38**	61.56	62.42
	0-40m	65.07	68.59	68.32	**70.38**	67.27	67.21
	40-50m	31.09	**33.61**	30.50	31.51	30.68	32.52
	50-64m	38.67	40.37	41.02	**42.80**	40.11	42.53

Experiment 3.2: Test

Table A.8: Average Precision (AP) metrics for various augmentation levels across all scenes in Experiment 3.2. It includes AP for each object class (CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY, OTHER) at different distances (0-40 m, 40-50 m, 50-64 m).

Class	Distance	Baseline	+20%	+40%	+60%	+80%	+100%
CAR	Overall	68.19	73.37	73.38	73.43	**75.10**	73.56
	0-40m	67.50	72.52	72.67	**74.41**	74.13	72.75
	40-50m	77.46	77.29	77.30	**81.16**	79.25	79.23
	50-64m	64.87	68.47	68.07	64.39	**70.25**	66.40
TRUCK	Overall	63.72	**69.60**	65.76	69.46	67.66	67.89
	0-40m	67.78	73.54	69.78	**77.27**	73.59	73.87
	40-50m	**70.00**	74.00	**70.00**	64.00	68.00	68.00
	50-64m	42.63	47.46	45.73	43.55	43.31	**45.94**
TRAILER	Overall	73.33	**75.50**	75.32	75.30	75.19	71.55
	0-40m	89.08	87.46	89.39	87.15	**91.09**	83.39
	40-50m	29.73	33.49	29.56	**37.59**	25.37	37.54
	50-64m	59.87	63.91	**67.66**	63.84	61.81	59.90
VAN	Overall	53.53	55.64	51.78	55.78	51.86	**57.72**
	0-40m	51.45	53.80	49.90	53.83	49.81	**57.78**
	40-50m	57.79	57.89	55.92	57.86	**58.00**	55.91
	50-64m	35.94	48.80	41.14	**65.05**	36.79	56.20
MOTORCYCLE	Overall	18.00	25.14	25.25	25.16	**25.31**	23.18
	0-40m	66.00	78.00	83.44	83.53	**84.00**	83.42
	40-50m	2.00	6.64	**7.13**	7.82	6.99	5.33
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
BUS	Overall	89.40	88.63	90.72	88.69	83.64	**91.68**
	0-40m	95.94	99.20	**99.82**	95.12	98.54	99.17
	40-50m	46.85	46.10	46.53	45.98	48.35	**56.47**
	50-64m	91.60	88.69	91.29	90.83	86.99	**91.62**
PEDESTRIAN	Overall	22.62	27.59	26.92	29.90	27.84	**31.71**
	0-40m	75.84	77.69	74.46	75.19	81.46	**82.39**
	40-50m	13.73	17.04	15.58	**20.32**	15.24	19.68
	50-64m	0.00	11.74	12.00	**20.70**	10.33	7.67
BICYCLE	Overall	54.55	59.00	48.18	**59.53**	27.35	33.57
	0-40m	61.30	63.90	59.38	**69.07**	29.28	38.18
	40-50m	13.81	**21.86**	12.53	10.15	13.16	15.87
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
EMERGENCY	Overall	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	0-40m	0.00	0.00	0.00	0.00	0.00	0.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
OTHER	Overall	62.00	62.00	62.00	62.00	62.00	62.00
	0-40m	62.00	62.00	62.00	62.00	62.00	62.00
	40-50m	0.00	0.00	0.00	0.00	0.00	0.00
	50-64m	0.00	0.00	0.00	0.00	0.00	0.00
Mean AP	Overall	60.53	63.65	61.93	**63.92**	59.59	61.29
	0-40m	63.69	66.81	66.08	**67.76**	64.39	65.30
	40-50m	31.14	33.43	31.45	32.49	31.44	**33.80**
	50-64m	39.49	42.91	42.59	**44.83**	40.95	42.77

Table A.9: Precision, Recall, and Intersection over Union (IoU) metrics for various augmentation levels across all scenes in Experiment 3.2. It includes precision, recall, and IoU for each object class (CAR, TRUCK, TRAILER, VAN, MOTORCYCLE, BUS, PEDESTRIAN, BICYCLE, EMERGENCY, OTHER) for the overall scene.

Class	Metric	Baseline	20%	40%	60%	80%	100%
CAR	Precision	68.81	73.89	73.90	73.95	**75.59**	74.08
	Recall	49.35	57.81	57.79	58.06	**59.98**	57.75
	IoU	**0.72**	0.70	0.70	0.71	0.70	0.71
TRUCK	Precision	64.43	**70.19**	66.43	70.06	68.29	68.52
	Recall	40.95	**48.73**	43.01	48.29	45.97	46.41
	IoU	0.70	0.70	**0.71**	0.70	0.70	0.70
TRAILER	Precision	73.86	**75.98**	75.80	75.78	75.68	72.11
	Recall	54.47	57.17	**58.01**	57.17	57.17	51.63
	IoU	0.71	0.70	0.70	0.70	**0.71**	0.71
VAN	Precision	54.44	56.51	52.72	56.65	52.80	**58.55**
	Recall	28.64	30.92	27.50	31.76	27.45	**33.41**
	IoU	**0.73**	0.68	0.70	0.70	0.70	0.67
MOTORCYCLE	Precision	19.61	26.60	**26.71**	26.62	26.78	24.68
	Recall	3.32	6.65	**6.97**	7.18	7.18	5.88
	IoU	**0.42**	0.36	0.37	0.35	0.33	0.38
BUS	Precision	89.61	88.85	90.90	88.91	83.96	**91.84**
	Recall	83.44	83.69	84.44	84.44	80.45	**87.50**
	IoU	**0.57**	0.52	0.54	0.50	0.53	0.50
PEDESTRIAN	Precision	24.14	29.01	28.36	31.28	29.25	**33.05**
	Recall	13.16	**18.98**	17.83	19.12	17.32	19.07
	IoU	0.37	0.42	0.48	0.46	**0.49**	0.49
BICYCLE	Precision	55.44	**59.80**	49.19	60.32	28.78	34.87
	Recall	46.17	**54.87**	46.79	57.01	29.45	34.22
	IoU	**0.33**	0.39	0.34	0.31	0.35	0.31
EMERGENCY	Precision	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	Recall	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	IoU	**0.39**	0.31	0.33	0.24	0.34	0.38
OTHER	Precision	62.75	62.75	62.75	62.75	62.75	62.75
	Recall	39.93	39.93	39.93	39.93	39.93	39.93
	IoU	0.31	0.37	**0.38**	0.35	0.35	0.36
Mean	Precision	81.50	83.54	80.89	82.89	81.76	**84.38**
	Recall	51.48	**55.70**	54.45	**56.27**	53.36	54.73
	IoU	**0.53**	0.52	0.52	0.50	0.52	0.52

Bibliography

[AB22] Alaba, S. Y. and Ball, J. E. “A Survey on Deep-Learning-Based LiDAR 3D Object Detection for Autonomous Driving”. In: *Sensors* 22.24 (2022). ISSN: 1424-8220. DOI: 10.3390/s22249577. URL: https://www.mdpi.com/1424-8220/22/24/9577.

[Bai22] Bai, H. “ICP Algorithm: Theory, Practice And Its SLAM-oriented Taxonomy”. In: June 2022. DOI: 10.48550/arXiv.2206.06435.

[Beh+19] Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., and Gall, J. “SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences”. In: *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*. 2019.

[Bij+20] Bijelic, M., Gruber, T., Mannan, F., Kraus, F., Ritter, W., Dietmayer, K., and Heide, F. “Seeing Through Fog Without Seeing Fog: Deep Multimodal Sensor Fusion in Unseen Adverse Weather”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11679–11689. DOI: 10.1109/CVPR42600.2020.01170.

[Bli+22] Blieninger, B., Creß, C., Gräfe, R., Honer, J., Jesorsky, O., Kaulbersch, H., Lakshminarasimhan, V., Lenz, C., Strand, L., and Zimmer, W. *Providentia++-Basis der digitalisierten Autobahn der Zukunft: Highway Real-Time Digital Twin: Schlussbericht des Verbundes: Laufzeit des Vorhabens: von: 01.01. 2020 bis: 30.06. 2022*. Technische Universität München, 2022.

[Blo21] Blog, N. D. *Detecting Objects in Point Clouds with CUDA PointPillars*. 2021. URL: https://developer.nvidia.com/blog/detecting-objects-in-point-clouds-with-cuda-pointpillars/.

[Cae+20] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. “nuScenes: A multimodal dataset for autonomous driving”. In: *CVPR*. 2020.

[CKY24] Chae, Y., Kim, H., and Yoon, K.-J. “Towards Robust 3D Object Detection with LiDAR and 4D Radar Fusion in Various Weather Conditions”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 15162–15172.

[CBK23] Creß, C., Bing, Z., and Knoll, A. C. “Intelligent Transportation Systems Using Roadside Infrastructure: A Literature Survey”. In: *IEEE Transactions on Intelligent Transportation Systems* (2023). Conference Name: IEEE Transactions on Intelligent Transportation Systems, pp. 1–. ISSN: 1558-0016. DOI: 10.1109/TITS.2023.3343434. URL: https://ieeexplore.ieee.org/document/10375912 (visited on 05/21/2024).

[Dat24] Dataset, T. T. *tum-traffic-dataset-dev-kit*. Version 1.0. 2024. URL: https://github.com/tum-traffic-dataset/tum-traffic-dataset-dev-kit.

[GLU12] Geiger, A., Lenz, P., and Urtasun, R. “Are we ready for autonomous driving? The KITTI vision benchmark suite”. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 3354–3361. DOI: 10.1109/CVPR.2012.6248074.

[Hah+22] Hahner, M., Sakaridis, C., Bijelic, M., Heide, F., Yu, F., Dai, D., and Gool, L. V. “LiDAR Snowfall Simulation for Robust 3D Object Detection”. In: *FSR 2019 paper 47* (2022). Available at ETH Zurich Repository.

[Hah+21] Hahner, M., Sakaridis, C., Dai, D., and Van Gool, L. “Fog Simulation on Real LiDAR Point Clouds for 3D Object Detection in Adverse Weather”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 15263–15272. DOI: 10.1109/ICCV48922.2021.01500.

[Has+17] Hasirlioglu, S., Riener, A., Huber, W., and Wintersberger, P. “Effects of exhaust gases on laser scanner data quality at low ambient temperatures”. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. 2017, pp. 1708–1713. DOI: 10.1109/IVS.2017.7995954.

[Hei+19] Heinzler, R., Schindler, P., Seekircher, J., Ritter, W., and Stork, W. “Weather Influence and Classification with Automotive LiDAR Sensors”. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2019, pp. 1527–1534.

[Hei+20] Heinzler, R., Piewak, F., Schindler, P., and Stork, W. “CNN-Based Lidar Point Cloud De-Noising in Adverse Weather”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 2514–2521. DOI: 10.1109/LRA.2020.2972865.

[JKP19] Jokela, M., Kutila, M., and Pyykönen, P. “Testing and Validation of Automotive Point-Cloud Sensors in Adverse Weather Conditions”. In: *Applied Sciences* 9.11 (2019). ISSN: 2076-3417. DOI: 10.3390/app9112341. URL: https://www.mdpi.com/2076-3417/9/11/2341.

[Kem+23a] Kempen, R. van, Rehbronn, T., Jose, A., Stegmaier, J., Lampe, B., Woopen, T., and Eckstein, L. “Enhancing Lidar-Based Object Detection in Adverse Weather Using Offset Sequences in Time”. In: Nov. 2023, pp. 1–6. DOI: 10.1109/ICECET58911.2023.10389489.

[Kem+23b] Kempen, R. van et al. *AUTOtech.agil : Architecture and Technologies for Orchestrating Automotive Agility*. RWTH-2023-09783. Conference Name: 32. Aachen Colloquium Sustainable Mobility. Aachener Kolloquium Fahrzeug- und Motorentechnik GbR, 2023. URL: https://publications.rwth-aachen.de/record/971700 (visited on 07/14/2024).

[Ket+24] Kettelgerdes, M., Sarmiento, N., Erdogan, H., Wunderle, B., and Elger, G. “Precise Adverse Weather Characterization by Deep-Learning-Based Noise Processing in Automotive LiDAR Sensors”. In: *Remote Sensing* 16.13 (2024), p. 2407.

[Kil+21] Kilic, V., Hegde, D., Sindagi, V., Cooper, A. B., Foster, M. A., and Patel, V. M. *Lidar Light Scattering Augmentation (LISA): Physics-based Simulation of Adverse Weather Conditions for 3D Object Detection*. 2021. arXiv: 2107.07004 [cs.CV]. URL: https://arxiv.org/abs/2107.07004.

[Kon+23] Kong, L., Liu, Y., Li, X., Chen, R., Zhang, W., Ren, J., Pan, L., Chen, K., and Liu, Z. “Robo3D: Towards Robust and Reliable 3D Perception against Corruptions”. In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2023, pp. 19937–19949. DOI: 10.1109/ICCV51070.2023.01830. URL: https://doi.ieeecomputersociety.org/10.1109/ICCV51070.2023.01830.

[Kra22] Kraftfahrzeuge (ika) RWTH Aachen University, I. für. *AUTOTECH AGIL Project*. https://www.ika.rwth-aachen.de/de/kompetenzen/projekte/automatisiertes-fahren/autotech-agil.html. 2022.

[Kut+20] Kutila, M., Pyykönen, P., Holzhüter, H., Colomb, M., and Duthon, P. "Automotive LIDAR Performance Verification in Fog and Rain". In: *IEEE Intelligent Vehicles Symposium (IV)*. 2020, pp. 1337–1342. DOI: 10.1109/IV47402.2020.9304721.

[Lan+19] Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., and Beijbom, O. "PointPillars: Fast Encoders for Object Detection from Point Clouds". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12697–12705.

[Li+22] Li, N., Ho, C. P., Xue, J., Lim, L. W., Chen, G., Fu, Y. H., and Lee, L. Y. T. "A Progress Review on Solid-State LiDAR and Nanophotonics-Based LiDAR Sensors". In: *Laser & Photonics Reviews* 16.11 (2022), p. 2100511. DOI: https://doi.org/10.1002/lpor.202100511. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/lpor.202100511. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/lpor.202100511.

[Lin+22] Lin, J., Yin, H., Yan, J., Ge, W., Zhang, H., and Rigoll, G. "Improved 3D Object Detector Under Snowfall Weather Condition Based on LiDAR Point Cloud". In: *IEEE Sensors Journal* 22.16 (2022), pp. 16276–16292. DOI: 10.1109/JSEN.2022.3188985.

[Lin+04] Linnhoff, C., Elster, L., Rosenberger, P., and Winner, H. *Road Spray in Lidar and Radar Data for Individual Moving Objects*. 2022-04. DOI: 10.48328/tudatalib-930. URL: https://tudatalib.ulb.tu-darmstadt.de/handle/tudatalib/3537.

[Liu+19] Liu, K. et al. "Survey on LiDAR Perception in Adverse Weather Conditions". In: *arXiv preprint arXiv:2304.06312* (2019). URL: https://arxiv.org/abs/2304.06312.

[Liu+20] Liu, W., Wang, X., Owens, J., and Li, Y. "Energy-based Out-of-distribution Detection". In: *Advances in Neural Information Processing Systems* (2020).

[MS15] Maturana, D. and Scherer, S. "VoxNet: A 3D Convolutional Neural Network for real-time object recognition". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 922–928. DOI: 10.1109/IROS.2015.7353481.

[Mäy+17] Mäyrä, A., Hietala, E., Kutila, M., and Pyykönen, P. "Spectral attenuation in low visibility artificial fog: Experimental study and comparison to literature models". In: *2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE. 2017, pp. 303–308.

[Mon+21] Montalban, K., Reymann, C., Atchuthan, D., Dupouy, P.-E., Riviere, N., and Lacroix, S. "A Quantitative Analysis of Point Clouds from Automotive Lidars Exposed to Artificial Rain and Fog". In: *Atmosphere* 12.6 (2021). ISSN: 2073-4433. DOI: 10.3390/atmos12060738. URL: https://www.mdpi.com/2073-4433/12/6/738.

[Ngu23] Nguyen, H. T. "LiDAR 3D Object Detection for Roadside Infrastructure Sensors Using Transformer". In: (2023). URL: https://www.ce.cit.tum.de/fileadmin/w00cgn/air/_my_direct_uploads/tung_nguyen_IDP_final.pdf.

[Pir+23a] Piroli, A., Dallabetta, V., Kopp, J., Walessa, M., Meissner, D., and Dietmayer, K. "Energy-Based Detection of Adverse Weather Effects in LiDAR Data". In: *IEEE Robotics and Automation Letters* 8.7 (2023), pp. 4322–4329. DOI: 10.1109/LRA.2023.3282382.

[Pir+23b] Piroli, A., Dallabetta, V., Kopp, J., Walessa, M., Meissner, D., and Dietmayer, K. *SemanticSpray Dataset*. https://semantic-spray-dataset.github.io/. Accessed: 2024-07-15. 2023.

[Pir+24] Piroli, A., Dallabetta, V., Kopp, J., Walessa, M., Meissner, D., and Dietmayer, K. "Label-Efficient Semantic Segmentation of LiDAR Point Clouds in Adverse Weather Conditions". In: *IEEE Robotics and Automation Letters* (2024).

[Pir+22] Piroli, A., Dallabetta, V., Walessa, M., Meissner, D. A., Kopp, J., and Dietmayer, K. C. J. "Robust 3D Object Detection in Cold Weather Conditions". In: *2022 IEEE Intelligent Vehicles Symposium (IV)* (2022), pp. 287–294.

[Pit+20] Pitropov, M., Garcia, D., Rebello, J., Smart, M., Wang, C., Czarnecki, K., and Waslander, S. "Canadian Adverse Driving Conditions dataset". In: *The International Journal of Robotics Research* 40 (Dec. 2020), p. 027836492097936. DOI: 10.1177/0278364920979368.

[Qi+17] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[Qi+24] Qi, Y., Liu, C., Scaioni, M., Li, Y., Qiao, Y., Ma, X., Wu, H., Zhang, K., and Wang, D. "Geometric information constraint 3D object detection from LiDAR point cloud for autonomous vehicles under adverse weather". In: *Transportation research part C: emerging technologies* 161 (2024), p. 104555.

[SOT22] Seppänen, A., Ojala, R., and Tammi, K. "4DenoiseNet: Adverse Weather Denoising From Adjacent Point Clouds". In: *IEEE Robotics and Automation Letters* 8 (2022), pp. 456–463. URL: https://api.semanticscholar.org/CorpusID:252280602.

[Shi+21] Shi, S., Wang, Z., Shi, J., Wang, X., and Li, H. "From Points to Parts: 3D Object Detection From Point Cloud With Part-Aware and Part-Aggregation Network". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.08 (Aug. 2021), pp. 2647–2664. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2020.2977026.

[Shi+19] Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., and Li, H. *PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection*. Dec. 2019.

[SWL19] Shi, S., Wang, X., and Li, H. "PointRCNN: 3D object proposal generation and detection from point cloud". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 770–779.

[Skr19] Skrodzki, M. "The k-d tree data structure and a proof for neighborhood computation in expected logarithmic time". In: (Mar. 2019).

[Sta+21] Stanislas, L., Nubert, J., Dugas, D., Nitsch, J., Sünderhauf, N., Siegwart, R., Cadena, C., and Peynot, T. "Airborne Particle Classification in LiDAR Point Clouds Using Deep Learning". In: *Field and Service Robotics (FSR)*. Springer. 2021, pp. 1–14.

[Sun+20] Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., Zhang, Y., Shlens, J., Chen, Z., and Anguelov, D. "Scalability in Perception for Autonomous Driving: Waymo Open Dataset". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 2443–2451. DOI: 10.1109/CVPR42600.2020.00252.

[Teu+22] Teufel, S., Volk, G., Von Bernuth, A., and Bringmann, O. “Simulating Realistic Rain, Snow, and Fog Variations For Comprehensive Performance Characterization of LiDAR Perception”. In: *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*. 2022, pp. 1–7. DOI: 10.1109/VTC2022-Spring54318.2022.9860868.

[Wan+24] Wang, J., Wu, Z., Liang, Y., Tang, J., and Chen, H. “Perception Methods for Adverse Weather Based on Vehicle Infrastructure Cooperation System: A Review”. In: *Sensors* 24.2 (2024), p. 374.

[Wan+22] Wang, T., Hu, X., Liu, Z., and Fu, C.-W. “Sparse2Dense: Learning to Densify 3D Features to Boost 3D Object Detection”. In: *Advances in Neural Information Processing Systems*. Ed. by Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. 2022. URL: https://openreview.net/forum?id=P6uZ7agiyCT.

[WL24] Wu, J. and Li, W. “P-19.8: 3D Object Detection Data Improvement Based on LiDAR for Autonomous Driving in Adverse Weather Conditions”. In: *SID Symposium Digest of Technical Papers*. Vol. 55. Wiley Online Library. 2024, pp. 1584–1587.

[Xie+23] Xie, G., Li, Y., Wang, Y., Li, Z., and Qu, H. “3D Point Cloud Object Detection Algorithm Based on Temporal Information Fusion and Uncertainty Estimation”. In: *Remote Sensing* 15.12 (2023). ISSN: 2072-4292. DOI: 10.3390/rs15122986. URL: https://www.mdpi.com/2072-4292/15/12/2986.

[YML18] Yan, Y., Mao, Y., and Li, B. “SECOND: Sparsely Embedded Convolutional Detection”. In: *Sensors* 18.10 (2018). ISSN: 1424-8220. DOI: 10.3390/s18103337. URL: https://www.mdpi.com/1424-8220/18/10/3337.

[YZK21] Yin, T., Zhou, X., and Krahenbuhl, P. “Center-Based 3D Object Detection and Tracking”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 11784–11793.

[Zha+21] Zhang, C., Huang, Z., Jr, M. H. A., and Rus, D. “LiDAR Degradation Quantification for Autonomous Driving in Rain”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 3458–3464.

[Zha+24a] Zhang, Q., Wang, L., Meng, H., Zhang, Z., and Yang, C. “Ship Detection in Maritime Scenes under Adverse Weather Conditions”. In: *Remote Sensing* 16.9 (2024), p. 1567.

[Zha+23] Zhang, Y., Carballo, A., Yang, H., and Takeda, K. “Perception and sensing for autonomous vehicles under adverse weather conditions: A survey”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 196 (2023), pp. 146–177. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2022.12.021. URL: https://www.sciencedirect.com/science/article/pii/S0924271622003367.

[Zha+24b] Zhang, Y., Ding, M., Yang, H., Niu, Y., Ge, M., Ohtani, K., Zhang, C., and Takeda, K. “LiDAR Point Cloud Augmentation for Adverse Conditions Using Conditional Generative Model”. In: *Remote Sensing* 16.12 (2024), p. 2247.

[ZT17] Zhou, Y. and Tuzel, O. “VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2017), pp. 4490–4499. URL: https://api.semanticscholar.org/CorpusID:42427078.

[Zhu+17] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. 2017.

[Zim+23a] Zimmer, W., Birkner, J., Brucker, M., Nguyen, H. T., Petrovski, S., Wang, B., and Knoll, A. C. "InfraDet3D: Multi-Modal 3D Object Detection based on Roadside Infrastructure Camera and LiDAR Sensors". In: *2023 IEEE Intelligent Vehicles Symposium (IV)*. 2023, pp. 1–8.

[Zim+23b] Zimmer, W., Creß, C., Nguyen, H. T., and Knoll, A. C. *A9 Intersection Dataset: All You Need for Urban 3D Camera-LiDAR Roadside Perception*. 2023. arXiv: 2306.09266 [cs.CV]. URL: https://arxiv.org/abs/2306.09266.

[Zim+23c] Zimmer, W., Creß, C., Nguyen, H. T., and Knoll, A. C. "TUMTraf Intersection Dataset: All You Need for Urban 3D Camera-LiDAR Roadside Perception". In: *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*. 2023, pp. 1030–1037. DOI: 10.1109/ITSC57777.2023.10422289. URL: https://ieeexplore.ieee.org/document/10422289.

[Zim+22] Zimmer, W., Ercelik, E., Zhou, X., Ortiz, X. J. D., and Knoll, A. "A Survey of Robust 3D Object Detection Methods in Point Clouds". In: *arXiv preprint arXiv:2204.00106* (2022). DOI: https://doi.org/10.48550/arXiv.2204.00106. URL: https://arxiv.org/abs/2204.00106.

[ZGK22] Zimmer, W., Grabler, M., and Knoll, A. "Real-time and robust 3D object detection within road-side LiDARs using domain adaptation". In: (2022). Submitted to BMVC British Machine Vision Conference 2022. DOI: https://doi.org/10.48550/arXiv.2204.00132. URL: https://arxiv.org/abs/2204.00132.

[ZRT19] Zimmer, W., Rangesh, A., and Trivedi, M. "3D BAT: A Semi-Automatic, Web-based 3D Annotation Toolbox for Full-Surround, Multi-Modal Data Streams". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. ISSN: 2642-7214. June 2019, pp. 1816–1821. DOI: 10.1109/IVS.2019.8814071. URL: https://ieeexplore.ieee.org/document/8814071 (visited on 05/12/2024).