GaussianFusion++: Adaptive Radar–Camera Fusion for Maritime 3D Detection

Lars Bosma
4967321

Thesis
RO57035

TU Delft

GaussianFusion++: Adaptive Radar–Camera Fusion for Maritime 3D Detection

Thesis

written by

Lars Bosma

Student Name	Student Number
Lars Bosma	4967321

Supervisor:	Dr. Holger Caesar
External supervisor:	Ehab El Amam - RH Marine
External supervisor II:	Tianlei Miao - RH Marine
Faculty:	Faculty of Mechanical Engineering
Course Code:	RO57035

The image on the cover was created by OpenAI's image generator.

TU Delft RH MARINE

Acknowledgement

I would like to take this opportunity to express my sincere gratitude to everyone who supported me throughout the course of this thesis.

First and foremost, I wish to thank Ehab El Amam and Tianlei Miao from RH Marine, my external supervisors, for their invaluable practical insights and constant encouragement. Their extensive knowledge of the maritime sector and radar systems helped me gain a clear understanding of the unique challenges in this field. Early discussions with them were instrumental in defining the research problem and shaping the direction of this work. Their feedback and suggestions throughout the project were vital in ensuring that the research remained relevant to real-world applications.

I am deeply indebted to Dr. Holger Caesar from TU Delft, my academic supervisor, for his expert guidance and constructive advice. His deep expertise in machine perception and multi-modal learning greatly enriched this research. His critical insights, timely feedback, and encouragement have helped me maintain a high scientific standard and have been key in turning ideas into concrete contributions.

I would also like to thank Yunjia Wang and Zhongbi Luo, PhD candidates at KU Leuven, for taking the time to attend an intermediate presentation of this work and providing thoughtful feedback. Their comments offered a fresh perspective and inspired improvements in both the experimental design and the presentation of the results.

Furthermore, Sabri Demir, a fellow thesis intern at RH Marine, deserves appreciation. I would like to sincerely thank him for generously sharing the 2D object detection dataset that was used to pretrain the CenterNet backbone. His assistance saved significant time and contributed to building a solid foundation for the 3D detection models.

I would like to extend my appreciation to Sol Han, PhD candidate at KAIST, and Yeongha Shin, MSc student at KAIST, for their help in understanding the Pohang Canal Dataset in depth. Their explanations and clarifications enabled me to construct the multi-modal 3D dataset that underpins this thesis.

Additionally, I wish to thank Gerard Kruisheer, CTO of Captain AI, for sharing his practical experience in constructing custom maritime datasets. His advice on data collection and annotation strategies was invaluable in ensuring the quality and completeness of the dataset used in this work.

Finally, I am very grateful to my family and housemates for their continuous support and patience. Casual conversations with them about this research often brought clarity and helped me maintain perspective and composure during challenging periods.

I am truly grateful to all these individuals for their generous support, insightful discussions, and encouragement, without which this thesis would not have been possible.

Abstract

Autonomous Surface Vehicles (ASVs) must operate safely in dynamic and often cluttered maritime environments such as ports and inland waterways. Achieving reliable situational awareness in these settings remains challenging due to the limited availability of annotated datasets, the complexity of multi-sensor alignment, and the degradation of monocular depth estimation at long range. This thesis addresses these challenges by developing a multi-modal 3D object detection framework that combines radar and camera data to improve perception robustness and depth accuracy.

To support supervised learning and evaluation, a custom dataset was constructed using real-world maritime sensor data, including radar, camera, and navigation information. Three-dimensional object annotations were manually created to capture static and moving targets across a wide range of distances.

Other core contributions include a probabilistic radar association strategy that models uncertainty in sensor measurements, a learnable Radar Gaussian Parameter Network (RGPNet) for dynamic estimation of radar association parameters, and a depth-aware mechanism for dynamically adjusting the region used for radar–camera fusion. These methods are designed to improve object localisation performance, particularly in conditions where sensor data is sparse, noisy, or spatially misaligned.

The proposed models were evaluated through a series of controlled experiments examining detection performance across different depth intervals and clutter conditions. The results show that probabilistic radar fusion improves robustness to noise and depth estimation errors, while depth-aware association strategies enhance localisation at long range without sacrificing near-field precision. Incorporating adaptive mechanisms into the fusion process was shown to be particularly effective in scenarios involving large depth variation or limited visual information.

Overall, the findings demonstrate that uncertainty-aware, adaptive fusion methods can significantly improve 3D object detection performance for ASVs. The approaches developed in this work offer a robust foundation for future research on scalable, data-efficient maritime perception systems for autonomous navigation in complex environments.

List of Acronyms

AAE Average Attribute Error

AHRS Attitude and Heading Reference System

AI Artificial Intelligence

AIS Automatic Identification System

AOE Average Orientation Error

AP Average Precision

ASE Average Scale Error

ASV Autonomous Surface Vehicle

ATE Avergae Translation Error

Attentive FPN Attentive Feature Pyramid Network

AVE Average Velocity Error

BEV bird's-eye view

CAMF Cross-attention Multi-layer Fusion

CAS Collision Avoidance System

COCO Common Objects in Context

EMSA European Maritime Safety Agency

FN false negative

FP false positive

FPS Farthest Point Sampling

GELU Gaussian Error Linear Unit

GPU Graphics Processing Unit

IoU Intersection over Union

KITTI Karlsruhe Institute of Technology and Toyota Institute

LFANet Learnable Frustum Association Network

mAP mean Average Precision

MDCA Multi-modal Deformable Cross Attention

MLP multilayer perceptron

NDS nuScenes Detection Score

NLL Negative Log-Likelihood

OS own ship

PCA Principal Component Analysis

PDF probability density function

RCS Radar Cross-Section

ReLU Rectified Linear Unit

RGPNet Radar Gaussian Parameter Network

SAF Spatial Attention Fusion

scSE Concurrent Spatial and Channel Squeeze & Excitation

SMCA Spatially Modulated Cross Attention

TP true positive

Contents

Chapter 1
Introduction

In the last decennia, developments in Artificial Intelligence (AI) [1], combined with financial and societal demands [2–4], have driven significant progress toward automation across the aviation, automotive, and maritime industries [5]. As a result, the transportation sector has been experiencing a rapid technological evolution. While the automotive industry has historically had a head start, due to earlier support from national governments as well as its massive industry size [6], the maritime industry's Autonomous Surface Vehicle (ASV) sector is growing at an impressive rate, as its market value is projected to grow to $3.29 billion by 2032 [7].

A key driver stimulating the evolution of ASVs is the demand for safety. According to an annual report of maritime incidents from the European Maritime Safety Agency (EMSA), human error is the primary cause for most accidents [8]. Notably, more than half of all accidents occur in internal waters, such as port areas and nearby channels. In these areas, traffic density and navigational complexity are highest. Consequently, this research's scope is bounded to inner and outer port environments.

A fundamental prerequisite for safe autonomous navigation is an effective vision system. A vessel cannot avoid obstacles or plan safe trajectories without accurately perceiving its surroundings. Therefore, robust, reliable, and context-aware 3D object detection systems are critical components in enabling autonomous vessels to operate safely and efficiently, particularly in the challenging and dynamic environments of ports.

This thesis project is a collaborative effort between RH Marine Netherlands B.V. and the Delft University of Technology, aiming to advance the field of multi-modal 3D object detection in maritime settings to enhance the situational awareness of ASVs. The goal is to design a novel multi-modal 3D object detection model, using radar and camera data, that has the potential to operate next to RH Marine's Collision Avoidance System (CAS) [9, 10].

1.1 Challenges in maritime 3D object detection

Designing an effective vision system capable of multi-modal 3D object detection in maritime environments introduces six unique challenges that differentiate this domain from those encountered in automotive or aerial applications. The first challenge is the scarcity of high-quality, annotated datasets for maritime 3D object detection [11]. Unlike the automotive domain, where large-scale datasets such as Karlsruhe Institute of Technology and Toyota Institute (KITTI) [12], Waymo [13] or nuScenes [14] are readily available, maritime datasets are limited in volume, diversity, and public availability. Moreover, the datasets that do exist are often unlabelled or lack sufficient metadata, making supervised training of detection models difficult. This scarcity necessitates either the development of custom datasets or the use of weak supervision, synthetic data, or transfer learning approaches to enable initial model training despite limited annotations.

In addition to limited data, the spatial distribution of objects in maritime scenes is particularly broad and variable. Target vessels or obstacles may appear just a few meters away or well over a kilometer in the distance. This wide dynamic range in object proximity demands highly flexible perception models capable of resolving both near-field detail and far-field structure. Standard detection architectures, often tuned for more constrained environments like roads, struggle to maintain accuracy across such varying scales.

Thirdly, more complexity arises from the nature of maritime radar systems. These radars typically perform a full 360-degree scan, often starting from a reference angle aligned with the bow of the vessel [15, 16]. When fusing radar data with front-view camera images, temporal misalignment has the potential to degrade the effectiveness of the fusion process. A single radar scan takes time to complete, meaning that the portion of the scan corresponding to the front view may not be temporally synchronised with the camera frame. As a result, objects in the radar frame may appear slightly earlier or later compared to their positions in the camera image, which can reduce the spatial accuracy of detection and introduce inconsistencies in object localisation.

Furthermore, the training of 3D object detection models depends on the quality of the initial 2D detections [17]. Since most 3D detection frameworks build upon 2D bounding boxes, inadequate 2D detection performance can prevent the model from consistently identifying relevant objects, thereby limiting its ability to learn the features necessary for accurately predicting 3D attributes such as depth, orientation, and spatial extent. When training from scratch, this dependency necessitates either transfer learning [18] using a robust 2D detection backbone with well-optimised regression heads, and a carefully designed training strategy, such as curriculum learning [19], staged

training [20], or multi-task co-adaptation [21], to compensate for early-stage weaknesses. Without such a foundation, models often struggle to converge or fail to generalise under diverse environmental conditions.

Sensor alignment presents the fifth challenge in dynamic maritime conditions. Maritime radar systems generally provide range and bearing, but no elevation information, making them insensitive to changes in pitch or roll of the own ship (OS). In contrast, cameras are directly affected by such oscillations, which are common in maritime environments due to waves and engine-induced vibrations. Accurate synchronisation and correction of sensor data, therefore, require high-frequency and precise measurements from an Attitude and Heading Reference System (AHRS) to maintain alignment between modalities.

Lastly, maritime environments are inherently noisy and unpredictable. External factors such as sunlight glare, sea spray, fog, and reflections from the water surface introduce significant visual and radar noise. These environmental effects degrade the quality of sensor data and introduce false positives and negatives in both detection and classification. Developing models that are robust to such conditions remains an open and critical challenge for safe autonomous navigation in ports and coastal waters.

1.2 Related work

This section provides an overview of research and concepts that underpin the core focus of this thesis: the development of a novel multi-modal 3D object detection model using radar and camera data for maritime environments. First, multi-modal sensor fusion methods are examined to illustrate how combining complementary sensor modalities can enhance perception robustness and accuracy. Next, two existing multi-modal 3D object detection frameworks that utilise frustum association are discussed. This is followed by an exploration of a publicly available maritime dataset used to construct a 3D detection benchmark. Finally, several deep learning training strategies are reviewed in the context of building robust multi-task detection models.

1.2.1 Multi-modal sensor fusion methods

In the context of autonomous navigation systems, the ability to perceive and understand the surrounding environment accurately and robustly is paramount. Relying on a single sensor modality is often insufficient due to the inherent limitations and vulnerability of each sensing technology. For example, vision-based systems such as RGB cameras offer rich semantic detail but suffer in poor lighting or adverse weather conditions. In contrast, radar systems are highly robust under challenging environmental conditions (e.g., fog, rain, or direct sunlight) but provide lower resolution and less semantic information. By integrating information from multiple sensor modalities, it is possible to harness the strengths of each sensor while mitigating their individual weaknesses [22].

Multi-modal sensor fusion has been extensively studied in robotics, autonomous driving, and increasingly in the maritime domain, where sensor reliability and environmental variability are critical. Sensor fusion enables improved detection accuracy, redundancy for fault tolerance, and more comprehensive scene understanding [23]. The key challenge lies in how to effectively integrate the different data streams in a manner that is both computationally efficient and semantically meaningful. Sensor fusion methods are generally categorised into three different levels: data-level fusion (early fusion), feature-level fusion (intermediate fusion), and decision-level fusion (late fusion) [24, 25].

Data-level fusion involves merging raw data from different sensor modalities before further processing [24]. This method benefits from being able to take all of the available information into account, potentially leading to a more comprehensive view of the environment. Feature-level fusion combines feature representations of multiple sensor modalities after individual preprocessing [26]. This allows each modality to be processed by its own feature extractor, allowing deep learning models to capture both modality-specific and shared representations. Decision-level fusion combines the outputs of individual models before making final decisions [25]. This fusion method is the most interpretable, but is not capable of joint feature learning.

In practice, some models employ a combination of these fusion strategies across different stages of the processing pipeline, an approach known as hybrid fusion [27]. For instance, raw radar point clouds may be fused with camera images at the data level, while later-stage feature maps are combined at the feature level before making final decisions. Hybrid fusion allows designers to exploit the advantages of each fusion level, often leading to improved robustness and flexibility in complex environments.

Within data- and feature-level fusion, there are different techniques that can be used to effectively combine the extracted features from each modality. Concatenation and element-wise aggregation are the most straightforward approaches [28]. Concatenation stacks feature maps or raw data from each modality along the channel dimension. This technique preserves modality-specific information, but burdens subsequent layers with the task of learning

meaningful joint representations. Element-wise aggregation can be achieved through either addition or multiplication. This reduces the number of feature maps that need to be processed compared to concatenation, but it does assume a certain degree of alignment between modalities. Element-wise aggregation allows deep learning models to emphasise specific modalities, and down-weight less informative modalities, on a per-sample basis. This can improve robustness in multi-modal settings [29].

Attention-based fusion introduces learnable mechanisms to dynamically weigh features across or within modalities [28]. This allows the model to focus on the most informative signals, improving robustness in conditions where one modality is degraded. Attention can be implemented in various forms, most notably as self-attention (within a single modality) or cross-attention (between modalities). These mechanisms are typically built on top of basic fusion methods like concatenation or element-wise aggregation, refining the merged features rather than replacing the core fusion operation.

Several attention modules have been proposed for camera-radar or camera-LiDAR fusion. For instance, Spatial Attention Fusion (SAF) computes a spatial attention map from radar features and uses it to re-weight vision features, focusing the model on image regions where radar data suggests the presence of objects [30]. The Attentive Feature Pyramid Network (Attentive FPN) uses Concurrent Spatial and Channel Squeeze & Excitation (scSE) blocks to highlight relevant radar-informed regions while suppressing irrelevant ones [31]. More advanced techniques such as Multi-modal Deformable Cross Attention (MDCA) allow flexible spatial alignment between modalities by computing attention scores based on input features, thus adapting to varying scene configurations [32].

Other recent innovations include Cross-attention Multi-layer Fusion (CAMF) [33], which uses deformable attention to align bird's-eye view (BEV) feature maps before fusing them via channel and spatial modules, and LearnableAlign [34], which dynamically aligns features from LiDAR and camera streams using cross-attention where LiDAR features act as queries. Similarly, Spatially Modulated Cross Attention (SMCA) constrains the attention region around object proposals using 2D Gaussian masks, ensuring attention is concentrated on semantically relevant areas [35]. These examples demonstrate how attention-based fusion can significantly enhance the precision and adaptability of sensor fusion in complex, real-world environments.

Modality interaction is a newer fusion technique that preserves the individuality of sensor modalities while enabling deep, structured interaction between them [36]. Instead of merging features into a single unified representation, this approach maintains parallel feature streams that exchange information selectively through cross-modal attention mechanisms. By allowing features to interact without being collapsed into a single vector space, modality interaction retains modality-specific characteristics while enriching them with complementary cues. Although this strategy is more computationally intensive than simpler fusion methods, it has been shown to improve performance in tasks that require high-fidelity reasoning across diverse sensor inputs.

1.2.2 Multi-modal 3D object detection methods

This thesis builds upon and further develops the CenterFusion [37] and CenterFusion++ [38] frameworks. These models extend the monocular detection capabilities of CenterNet [39] by incorporating radar signals to improve 3D localisation, depth estimation, and object motion prediction in complex driving environments. Both methods are built around the idea of frustum association, but differ in how and when radar information is fused into the detection pipeline.

The foundation of both models is CenterNet, a keypoint-based object detector that represents objects by the center of their 2D bounding boxes [39]. CenterNet predicts a class-specific heatmap, where the peak indicates the likely location of the object's center in the image. At each predicted center location, additional regression heads are used to estimate object properties such as depth, orientation, size, and motion. The network architecture typically uses a DLA-34 backbone, which aggregates semantic and spatial features across different layers to enable precise localisation and regression.

CenterFusion extends this approach by introducing a feature-level fusion mechanism using radar data [37]. After the monocular 2D and 3D detections are generated by CenterNet, the model constructs a 3D frustum for each detected object. This frustum originates from the camera and extends into 3D space, covering the region where the object is likely to be located. To perform radar association, CenterFusion applies a technique called pillar expansion, where each radar detection is extruded into a vertical 3D column. This compensates for the lack of height information in radar and improves spatial coverage within the frustum.

Radar pillars that intersect with the frustum are treated as candidate associations. Among them, the pillar closest to the camera is selected. Radar attributes are projected into a radar feature map. The radar feature map includes both depth and radial velocity channels, which providing complementary spatial and motion cues. This map is then fused with the image feature map, typically through concatenation at the object's center location.

The fused representation is passed to secondary regression heads, which refine 3D attributes such as depth, size, orientation, and motion. CenterFusion's architecture is illustrated in Figure 1.1.

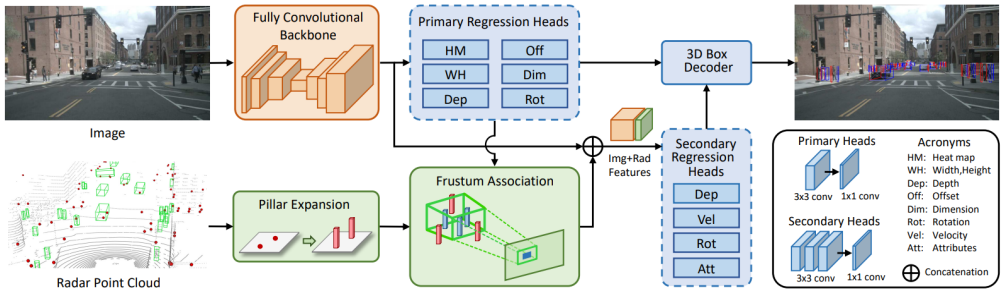


Figure 1.1: Overview of the CenterFusion architecture [37].

This form of frustum-based radar association is a key innovation in CenterFusion, enabling sparse radar returns to be selectively incorporated into object-level reasoning without explicit point cloud segmentation or object-level tracking. However, since the association depends on selecting the closest radar point within the frustum, it remains sensitive to the quality of the initial monocular 3D predictions. An important parameter in this process is δ, which controls the depth-wise expansion of the frustum. By enlarging the frustum volume, the likelihood increases that radar returns associated with an object will fall inside the candidate region. This makes the model somewhat more tolerant to inaccuracies in the initial 3D object location predicted from the image, thus improving robustness in challenging or uncertain conditions. This effect is illustrated in Figure 1.2.

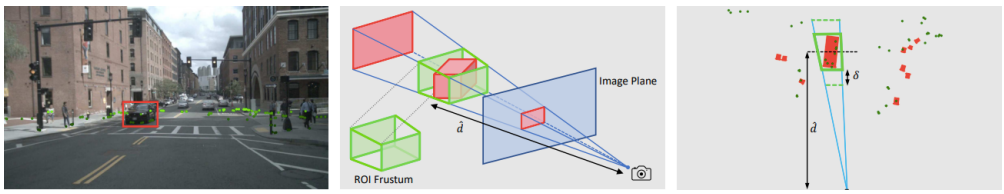


Figure 1.2: Visualisation of how the δ parameters operates [37].

To address the limitations of associating radar data using only the closest point within a frustum, CenterFusion++ introduces two key improvements: early fusion and a Learnable Frustum Association Network (LFANet) [38]. Its architecture is illustrated in Figure 1.3. Early fusion is applied by augmenting the input to the backbone with radar-derived channels. The resulting image tensor includes standard RGB channels along with projected radar features: depth, radial velocities (v_x, v_z), and Radar Cross-Section (RCS). The inclusion of RCS as an additional input channel allows the network to leverage material and shape-related cues, as RCS can reflect object surface characteristics. Incorporating RCS into the early fusion input was found to improve detection robustness, especially in challenging conditions. This provides the backbone with fused information from the earliest stages of processing, enabling better feature learning. To prevent over-reliance on the camera, the authors introduce a training-time regularisation technique called BlackIn, which randomly masks out the RGB channels to ensure the network learns to utilise radar features effectively.

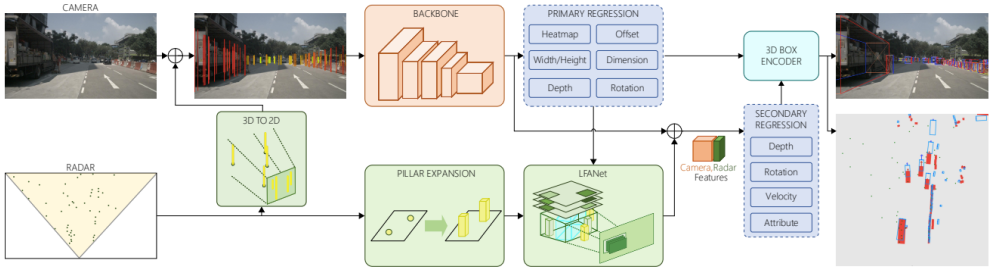


Figure 1.3: Overview of the CenterFusion++ architecture [38]. Radar information is fused both early in the backbone and again using LFANet.

The second major component, LFANet, replaces the single radar point selection method with a fully learnable mechanism. LFANet operates on a radar-generated snapshot of the frustum in the form of a BEV tensor, which encodes radar-derived features within the region of interest. This tensor is processed by a convolutional network that predicts an artificial radar point $\mathbf{r}^*$, which represents the most likely 3D location of the object's center. Similar to CenterFusion, this point is then inserted into a radar feature map and fused at the object's image center. The use of LFANet enables the network to suppress irrelevant or noisy radar points and adaptively associate radar signals based on scene context.

The variant used in CenterFusion++, LFANet-IMG, is specifically designed to extract semantically rich features from the radar snapshot. It takes as input a tensor $X \in \mathbb{R}^{\kappa \times \kappa \times n_{\text{feat,LFA}}}$, where κ is the spatial resolution of the snapshot, and $n_{\text{feat,LFA}}$ is the number of radar feature channels. The network architecture consists of a series of convolutional blocks, each comprising a 3×3 convolutional layer with ReLU activation followed by a max-pooling operation. These blocks progressively reduce the spatial dimensions of the input until it reaches 1×1, effectively summarising the entire snapshot into a feature vector of size n_f. This vector is then passed through two fully connected layers: the first with n_f neurons and Rectified Linear Unit (ReLU) activation, and the second with three output neurons that regress the artificial radar point $\mathbf{r}^* = [d^*, v_x^*, v_z^*]$, representing estimated depth and velocity components. This output is then inserted into the radar feature map and used for fusion at the object's center in the image. The depth-oriented, semantically expressive architecture of LFANet-IMG yielded better performance than alternative variants and was therefore adopted in the final model. Its architecture is illustrated in Figure 1.4.

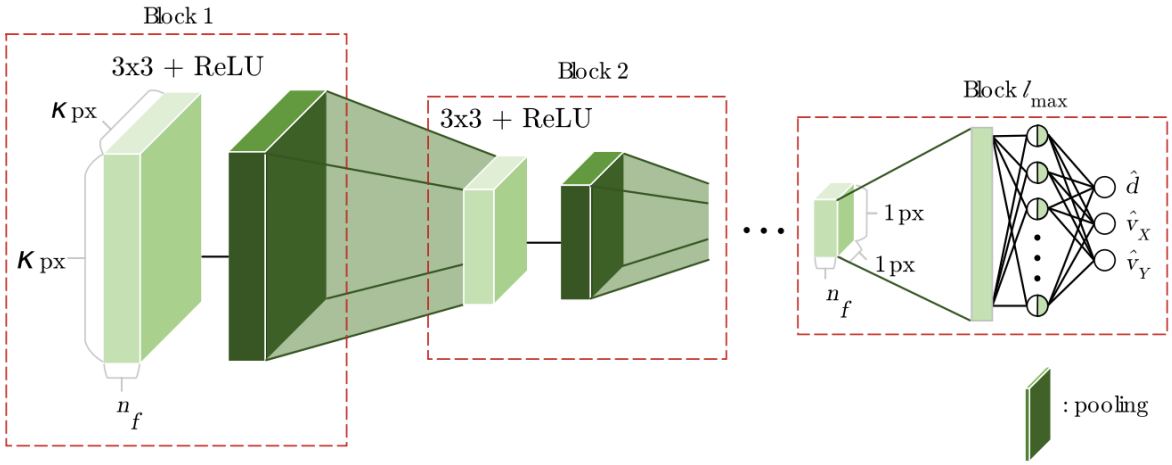


Figure 1.4: Overview of the LFANet architecture [38].

Together, these modifications allow CenterFusion++ to leverage radar data more effectively and robustly than its predecessor. By performing early fusion and learning to associate radar points dynamically, the model improves object detection and tracking in sparse or visually degraded conditions, making it particularly suited for use in maritime environments.

1.2.3 Deep learning model training strategies

Designing and training multi-modal 3D object detection models from scratch presents several challenges, especially when annotated data is limited or the model architecture incorporates multiple interdependent tasks. To address these challenges, various training strategies have been developed to improve convergence, generalisation, and modular learning efficiency. This section discusses three such strategies relevant to this thesis: transfer learning, staged training, and multi-task co-adaptation.

Transfer learning refers to the practice of pretraining a neural network on one dataset or task, and then fine-tuning it on a different but related task [18]. This approach is particularly useful when the target dataset is small or lacks diversity, as the pretrained model already encodes useful low-level and mid-level features that can accelerate learning. In the context of 3D object detection, transfer learning is often used by initialising a backbone network with weights pretrained on large-scale 2D object detection datasets such as Common Objects in Context (COCO) [40] or KITTI [12]. This not only reduces training time but also improves stability and accuracy, especially in the early stages of training.

Curriculum learning is another training strategy designed to improve model convergence by controlling the order in which training samples or tasks are introduced [19]. Inspired by the way humans learn, starting with simpler concepts before tackling more complex ones, this approach allows the model to gradually adapt to increasing difficulty. In the context of 3D object detection, curriculum learning can involve prioritising well-illuminated or close-range objects early in training, before exposing the model to more ambiguous, distant, or occluded targets. This structured learning path can stabilise gradients, reduce overfitting in early stages, and improve generalisation in scenarios with high data variability, such as maritime environments.

Staged training involves partitioning the model's learning process into distinct phases, during which different components of the architecture are trained with varying emphasis [20]. This is particularly effective for models with separate sub-tasks, such as 2D detection and 3D regression. In early stages, the training may focus primarily on optimising the 2D detection heads to ensure reliable object localisation in the image plane. Once the model demonstrates sufficient 2D detection performance, subsequent stages can focus on training the 3D regression heads, allowing them to learn more effectively from stable feature maps. This phased approach helps avoid unstable gradients and conflicting objectives between components during joint optimisation. Similar staged training strategies have been employed in other domains, such as screen content image quality assessment, where one network first classifies distortion types before a second predicts quality [41]. In multi-modal learning, staged strategies have also been used to selectively emphasise clean modalities before introducing noisier ones [29].

Multi-task co-adaptation is a training strategy in which all sub-tasks of a model—such as 2D detection and 3D regression—are trained jointly from the start, but with shared representations and loss terms that adaptively balance the contribution of each task [21]. Instead of training components in isolation or sequentially, this approach encourages the model to learn features that are simultaneously useful across multiple outputs. In the context of 3D object detection, the backbone and feature extractors are jointly optimised based on feedback from both 2D and 3D prediction heads. This promotes cross-task generalisation and can lead to more compact and efficient representations. However, it requires careful design of loss weighting and gradient flow to avoid one task dominating the learning process.

1.3 Contributions

Reliable multi-modal 3D object detection is essential for enabling ASVs to safely operate in complex and dynamic maritime environments, such as ports and coastal waters. This thesis contributes to advancing this capability through both dataset creation and novel model development. The main contributions are:

1. **A maritime 3D object detection benchmark dataset:** A custom dataset was created to support the training and evaluation of multi-modal 3D object detection models in maritime environments. This dataset includes synchronised data acquired from front-facing camera images, marine radar data (both as 2D radar sweeps and pseudo point clouds), and LiDAR point clouds, along with AHRS and GPS measurements. It covers a range of port environments and is fully annotated with 3D object labels, enabling benchmarking and supervised training in a domain where such resources are scarce.

2. **A novel frustum association method using a bivariate Gaussian weighting mechanism (GaussianFusion):** This model introduces a novel radar association mechanism based on a bivariate Gaussian distribution. Unlike CenterFusion [37], which selects only the nearest radar point within a frustum, GaussianFusion considers all radar points within a frustum, weighting them by their spatial proximity to the frustum center. A weighted average is then used to form an artificial radar point, which is fused with image features for 3D regression. This improves robustness to noise, sparsity, and misalignment.

3. **A novel learnable frustum association mechanism using neural Gaussian estimation (GaussianFusion++):** Building on the previous method, this variant introduces a lightweight neural network, called Radar Gaussian Parameter Network (RGPNet) that learns to estimate the bivariate Gaussian parameters from the radar data within a frustum. The network takes all radar points within a frustum and predicts a learnable Gaussian distribution to create an artificial radar point. This approach dynamically adapts to scene complexity and object characteristics, further enhancing the precision of camera-radar fusion under diverse environmental conditions.

4. **Depth-adaptive frustum construction:** A novel frustum expansion technique is proposed to address the difficulty of radar association at longer ranges. The method increases the size of the 3D frustum in proportion to the predicted object depth, improving radar point inclusion and association quality in distant scenes. While this technique is still under investigation, it holds promise for improving detection recall at long ranges.

1.4 Outline

This rest of this thesis report is structured into four chapters. Chapter 2 details the construction of the maritime 3D benchmark dataset. Chapter 3 analyses the limitations of baseline detection frameworks and introduces two novel camera–radar fusion models, GaussianFusion and GaussianFusion++. It also discusses the dyanmic frustum enlargement module. Chapter 4 describes the training strategies and the evaluation metrics employed. Chapter 5 outlines the experiments, including implementation details and training configurations. It also presents the results of these experiments, evaluating model performance across multiple metrics to assess robustness, accuracy, and task-specific effectiveness. Chapter 6 interprets the experimental outcomes and explores the implications of key findings. It critically analyses how model modifications influence performance and discusses trade-offs between complexity and robustness in camera–radar fusion. Chapter 7 summarises the main contributions and findings, and reflects on the overall success and limitations of the proposed approach. Additionally, it discusses potential directions for future work, including improvements to frustum construction, fusion strategies, dataset annotation, and robustness in real-world deployment.

Chapter 2
Dataset

This chapter provides a comprehensive overview of the dataset used for training and evaluating the 3D object detection models developed in this thesis. It begins with a description of the Pohang Canal Dataset [42], a publicly available collection of maritime sensor data that served as the foundation for this research. The remainder of the chapter outlines the structured pipeline used to transform the raw data into a usable benchmark, including frame selection, annotation formatting, radar-to-point cloud conversion, and depth-stratified statistics.

The construction of this fully annotated benchmark dataset forms one of the core contributions of this thesis. It was developed through the manual curation and annotation of multimodal sensor recordings, enabling supervised learning and performance evaluation in long-range maritime 3D perception tasks.

2.1 The Pohang Canal Dataset

The dataset used in this thesis originates from the Pohang Canal Dataset [42], a publicly available, multi-modal maritime dataset designed to support research on autonomous navigation in complex marine environments. Collected over a 7.5 km route in Pohang, South Korea, it spans diverse settings such as a narrow canal, port zones, and near-coastal waters. These environments introduce varying levels of complexity in traffic, visibility, and sensor challenges, making the dataset well-suited for benchmarking perception algorithms.

2.1.1 Sensor setup

The data acquisition platform was a 7.9 m-long cruise vessel equipped with a broad suite of perception and navigation sensors. The sensor suite includes: a stereo RGB camera, an infrared camera, an omnidirectional camera, three LiDARs (front, port, and starboard) with a range of 120 m, a marine radar with a range of 1654.8 m, a GNSS-based GPS system with RTK capability, and an AHRS. The stereo and infrared cameras were front-mounted with hardware-synchronised triggering, while the radar and omnidirectional camera were placed on a vertical slide at the stern. The sensor configuration is illustrated in Figure 2.1.

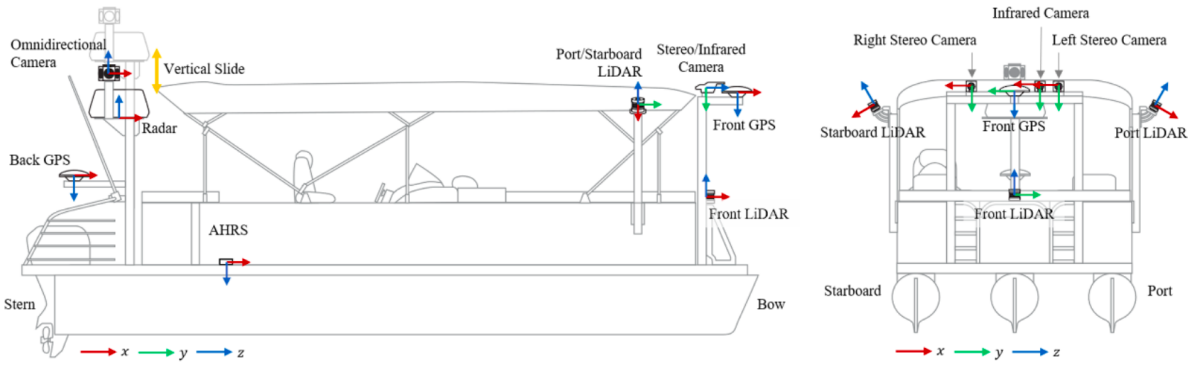


Figure 2.1: Sensor configuration of the Pohang Canal Dataset collection platform [16].

2.1.2 Calibration and synchronisation

Each sensor is provided with intrinsic calibration parameters (e.g., focal length, distortion coefficients) and extrinsic transformations relative to the AHRS [42]. These include rotation quaternions and translation vectors. The system architecture consisted of three computers for data logging, with NTP-synchronisation [43] achieved through the Chrony time-sync library [44]. All data is timestamped in UNIX format [45], allowing for precise temporal alignment during post-processing.

2.1.3 Sampling rates

Sensors operated at varying frequencies depending on their function:

- **Stereo and infrared cameras:** 10 Hz
- **Omnidirectional camera:** 10 Hz (daytime only)
- **LiDARs:** 10 Hz per sensor (asynchronous)
- **Marine radar:** 1.0–1.3 rotations per second (approx. 1 Hz)
- **GPS:** 5 Hz
- **AHRS:** 100 Hz

2.1.4 Data formats

Each sensor's output is organised in a clear directory structure. RGB and thermal images are provided as PNG or JPG files, with stereo images at 2048×1080 resolution and thermal images as 16-bit PNGs (640×512). Radar scans are stored as 2D PNG images per sweep at 2048×2048 resolution. Omnidirectional camera data includes six synchronised JPG images (640×512) at per timestamp (five horizontal, one top). LiDAR data is provided in binary format, storing XYZ coordinates, intensity, reflectivity, ambient values, and internal sensor timestamps for each point. The GPS data is stored in a .txt file and contains information on the latitude, longitude, heading, precision and quality indicators, and geoid height. The AHRS data is also stored in a .txt file, containing information on the orientation, angular velocity, and acceleration of the OS [16]. All data is accompanied by timestamp logs.

2.1.5 Motion compensation and trajectory

Accurate motion compensation is critical in maritime environments due to vessel roll, pitch, and yaw dynamics. The dataset includes a high-fidelity baseline trajectory, estimated using a tightly coupled SLAM framework that fuses front LiDAR data with AHRS and RTK-GPS measurements via iSAM-based graph optimisation [46]. This ensures accurate ground truth for both sensor fusion and 3D object detection, even in areas with GPS degradation such as under bridges.

2.1.6 Use in benchmarking

The dataset's rich sensory coverage, precise calibration, and dynamic maritime scenarios make it uniquely suited for training and evaluating multi-modal 3D object detection frameworks. It supports tasks such as camera–radar and camera–LiDAR fusion, and long-range maritime perception. However, the dataset does not provide labelled object annotations. Therefore, all 3D object labels used in this thesis were manually created as part of this research. This effort not only enabled effective model training and evaluation but also contributed toward developing annotated benchmarks for maritime computer vision research.

2.2 Dataset construction

To transform the raw multi-modal recordings from the Pohang Canal Dataset [42] into a usable benchmark for 3D object detection, a structured pipeline was developed for filtering, annotating, and data formatting. This section outlines the steps involved in refining the dataset, including frame selection, COCO-style annotation design, radar-to-point cloud conversion, and the creation of a train/validation split suitable for model training and evaluation.

2.2.1 Frame selection and filtering

Initially, raw sensor data contained many frames with little to no relevant information or redundant views. These were manually filtered out to improve dataset quality and reduce unnecessary duplication. Specifically, camera images lacking meaningful obstacles, or appearing overly similar to adjacent frames, were excluded. Once the remaining camera images were selected, corresponding LiDAR and radar data were retrieved using their respective timestamps to ensure temporal consistency.

2.2.2 Annotation format

With the filtered dataset established, annotations were created in the COCO format [40] to facilitate compatibility with standard object detection and tracking pipelines. The final annotation structure includes the following main keys:

- `images`: a list of dictionaries containing metadata about each image
- `annotations`: a list of dictionaries with object annotations for each image
- `categories`: a single category entry with {`"name": "obstacle", "id": 1`}
- `attributes`: a single attribute entry with {`"": 0`}
- `pointclouds`: an empty list (as this is stored in `images`)

Each entry under `images` includes the following information:

- `id`: a unique integer identifier for an image
- `file_name`: file path to the image
- `calib`: a 3×4 camera calibration matrix
- `frame_id`: indicating frame number (same as `id`)
- `sensor_id`: sensor identifier (which is always 1)
- `trans_matrix`: 4×4 transformation matrix for pose from the camera coordinate frame to the global coordinate frame
- `velocity_trans_matrix`: 4×4 transformation matrix for rotating velocity from the camera coordinate frame to the global coordinate frame
- `width`, `height`: image width and height
- `pose_record_trans`, `pose_record_rot`: AHRS-to-global translation vector and quaternion
- `cs_record_trans`, `cs_record_rot`: camera-to-AHRS translation vector and quaternion
- `radar_pc`: radar point cloud projected into the camera coordinate frame
- `camera_intrinsic`: 3×3 matrix capturing camera focal lengths and optical center

This information was derived using available AHRS, GPS, calibration data, and radar projections.

Each entry under `annotations` includes:

- `id`: unique identifier for the annotation
- `image_id`: unique identifier for the corresponding image
- `category_id`: unique identifier for the category label (always 1)
- `dim`: object dimensions [height, width, length],
- `location`, `depth`: 3D position and depth in the camera coordinate frame
- `occluded`, `truncated`: binary indicators for occlusion and truncation
- `rotation_y`: yaw angle in the x-z plane
- `amodel_center`: [x, y] of the projected center of the obstacle in the image
- `alpha`: the observation angle of an object relative to the camera's viewing direction on the image plane

- `iscrowd`: crowd flag (always 0)
- `track_id`: object instance ID
- `attributes`: attribute ID (always 0)
- `velocity`, `velocity_cam`: object velocity in both global and camera coordinate frames
- `bbox`, `area`: 2D bounding box and its area

2.2.3 Object annotation

To achieve this, each image was first manually annotated by drawing 2D bounding boxes around visible obstacles. For each bounding box, the following metadata was recorded: a unique annotation ID, track ID (shared across frames for the same object), visibility of width/length, occlusion and truncation status, movement status, and an estimate of yaw angle in the camera frame.

To obtain 3D localisation and orientation, LiDAR point clouds were transformed into the camera coordinate frame. Points falling within each 2D bounding box were extracted by projection, followed by clustering and manual filtering to remove outliers. The resulting clusters were used to compute:

- Object location: [x, z] centroid and y at object bottom
- Yaw angle: computed using Principal Component Analysis (PCA) [47]
- Dimensions: length estimated as the extent along the yaw angle, width estimated as the extent along the axis perpendicular to the yaw angle in the ground plane, and height estimated as the vertical extent
- Velocities: derived only for moving objects by comparing object displacements across frames using temporally adjacent LiDAR scans and transformation matrices from the camera to the world coordinate frame
- Camera-frame velocities: obtained by applying the `velocity_trans_matrix` to the global velocity vector

For moving objects, the yaw angle was additionally computed from the camera-frame velocity vector using the angle between its x and z components. All other metadata were calculated accordingly. The `iscrowd` field was always set to 0, reflecting separate annotations for each object.

2.2.4 Radar-based pseudo point cloud generation

LiDAR coverage was limited by range, especially for distant objects. To supplement this, radar data was used. However, radar was only available in the form of 2048×2048 PNG images. An example of such an image is shown in Figure 2.2.

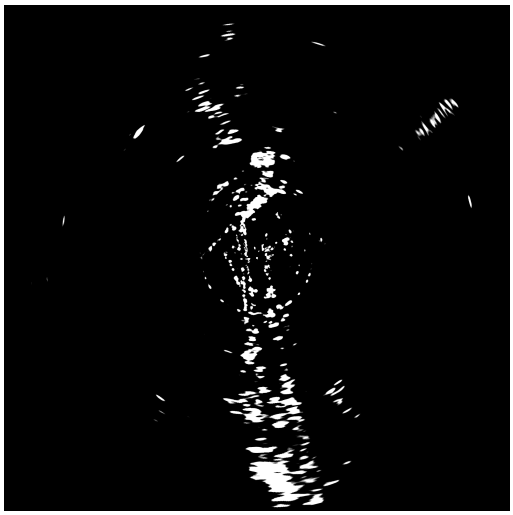


Figure 2.2: Example radar image [42]. White pixels indicate returns.

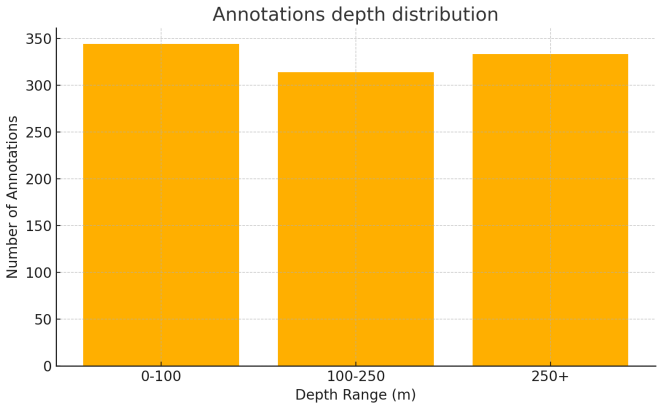
Pseudo point clouds were generated by randomly converting white pixels in the radar images into 3D points, scaled based on the radar's maximum range and an image's resolution. The RCS for each point was estimated heuristically, based on the size of the radar blob and the relative position of the pixel within that blob. This method of RCS estimation does not account for material properties, which also influence true reflectivity. Consequently, the RCS values are not physically meaningful and should not be interpreted as accurate indicators of radar reflectivity. They were retained solely to maintain compatibility with the required point cloud format and for possible future use, should more reliable estimation methods or material data become available. As the radar data originated from 2D images, no velocity or temporal metadata was available; only spatial coordinates and estimated RCS values could be recovered.

To ensure compatibility with the CenterFusion pipeline, originally designed for use with the nuScenes dataset [14], the pseudo point clouds were saved in a `.pcd` format with 18 fields. Among these, only the `X`, `Y`, and `Z` fields were assigned meaningful values. The remaining fields were populated with default values (either `0` or `NaN`) to maintain the expected data structure. These point clouds were then processed analogously to LiDAR data for the purpose of generating 3D object annotations. The radar-to-point cloud conversion procedure is summarised in Algorithm 1 in Appendix A.1.

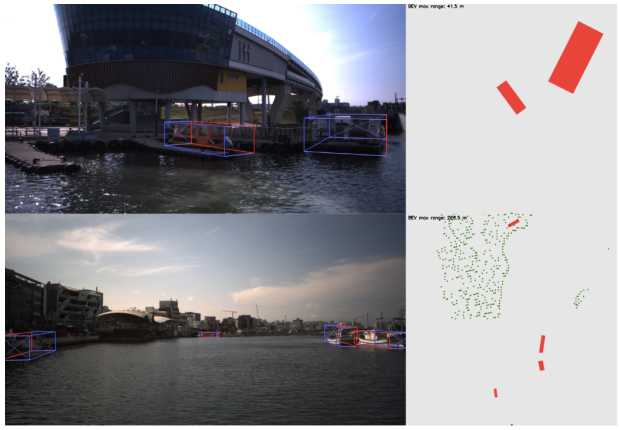
2.2.5 Dataset statistics and train/val split

To characterise the distribution of annotated objects, the dataset contains a total of 991 obstacle annotations, of which 160 are labelled as moving and 831 as stationary. This reflects a class imbalance that mirrors real-world scenarios in maritime environments, where stationary objects such as docked vessels, bridge pillars and buoys are more common than actively moving vessels. In terms of depth distribution, 344 objects are located within 0–100 m, 314 objects fall in the range of 100–250 m, and 333 objects are beyond 250 m. This relatively even spread across depth ranges ensures that the dataset supports training and evaluation across short, mid, and long-range perception tasks. Figure 2.3a visualises this depth distribution in a histogram.

After generating the complete set of labels, 3D bounding boxes were visualised on the camera images for inspection with their corresponding BEV representations. Errors or inconsistencies were corrected manually. Two representative examples are illustrated in Figure 2.3b. The complete dataset consists of 258 camera-radar image combinations with 991 annotations. The dataset was finally split into a training and a validation set. The training set contained 206 image-radar combinations with 769 annotations. The validation set contained 52 camera-radar combinations with 222 annotations. The annotated objects include ships, buoys, and bridge pillars.



(a) Depth distribution of objects in the dataset. The depth distribution includes 344 objects at 0–100 m, 314 at 100–250 m, and 333 beyond 250 m.



(b) Example of annotated 3D bounding boxes visualised on a camera image with their corresponding BEV representations.

Figure 2.3: Overview of annotated object distribution and examples.

Chapter 3
Methodology

This chapter presents the remaining core methodological contributions of this thesis, which together aim to improve radar–camera fusion for monocular 3D object detection in maritime environments. Two novel architectures are proposed: GaussianFusion, which introduces a probabilistic radar association strategy based on Gaussian-weighted aggregation, and GaussianFusion++, which extends this idea by incorporating the RGPNet, a learnable module that predicts radar association parameters from scene context. In addition, a dynamic frustum enlargement module is developed to adapt the radar–camera fusion region based on object distance, addressing the increasing uncertainty in monocular depth estimation at long range. These components form a robust and adaptive perception pipeline tailored to the challenges of sparse, cluttered, and visually ambiguous maritime environments.

3.1 GaussianFusion

The GaussianFusion architecture was developed to address two critical limitations in CenterFusion [37]. The first limitation is that it heavily relies on the accuracy of the initial monocular 3D object detection to estimate depth and project a frustum with which radar points are associated. This dependence can introduce significant error, particularly in challenging visual conditions. The second limitation is that the radar point closest to the camera is associated with the frustum, which is inherently uncertain. This selection strategy effectively assumes that the closest point corresponds to the correct object, which may not hold true, especially in crowded or noisy scenes, if the accuracy of the initial monocular 3D object detection is inadequate.

GaussianFusion improves this process by replacing the single-point association mechanism with a novel probabilistic weighting scheme over all radar points inside the frustum. Specifically, it uses a bivariate Gaussian distribution [48] to assign weights to candidate radar points based on their spatial proximity to the predicted center of the frustum in both horizontal and depth dimensions. Prior research has shown that Gaussian weighting improves robustness in noisy, sparse, or ambiguous environments [49,50]. Figure 3.1 illustrates this novel frustum association mechanism.

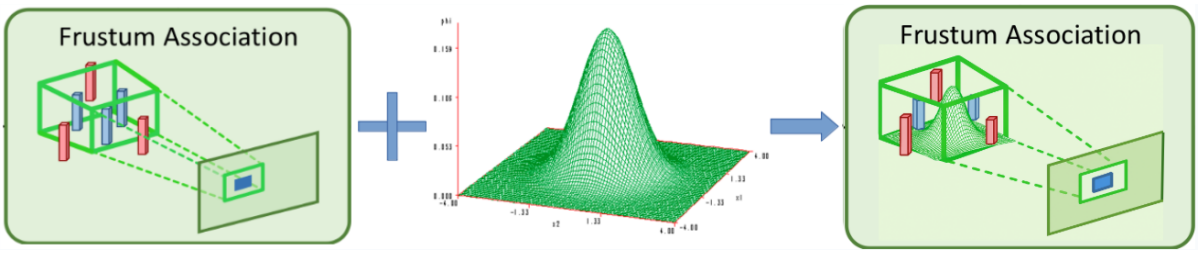


Figure 3.1: Visualisation of the bivariate Gaussian frustum association mechanism [37,48]. Radar points are assigned weights based on their spatial proximity to the predicted object's center.

The remainder of the pipeline is identical to CenterFusion. Image features are extracted using a DLA-34 convolutional backbone, and object centers are predicted via heatmaps. At each detected center, the model regresses the corresponding 2D bounding box and initial monocular depth, size, and orientation estimates. From each of these estimates, a 3D frustum is constructed, and all radar points that fall within a frustum are collected. Instead of selecting the single nearest radar point per frustum, GaussianFusion computes a weighted combination of all radar points using a bivariate Gaussian distribution, centered at the bounding box midpoint and monocular depth estimate. This produces an artificial radar point that is more robust to noise, sparsity, and frustum misalignment. From this artificial radar point, features, such as depth and velocity, are extracted and written into a radar feature heatmap at its corresponding location. This is done for every prediction. The final radar feature heatmap is concatenated to the image features for further processing by secondary regression heads. Figure 3.2 shows an illustration of the GaussianFusion architecture.

The standard probability density function (PDF) for bivariate Gaussian distribution is given by Equation 3.1 [48]:

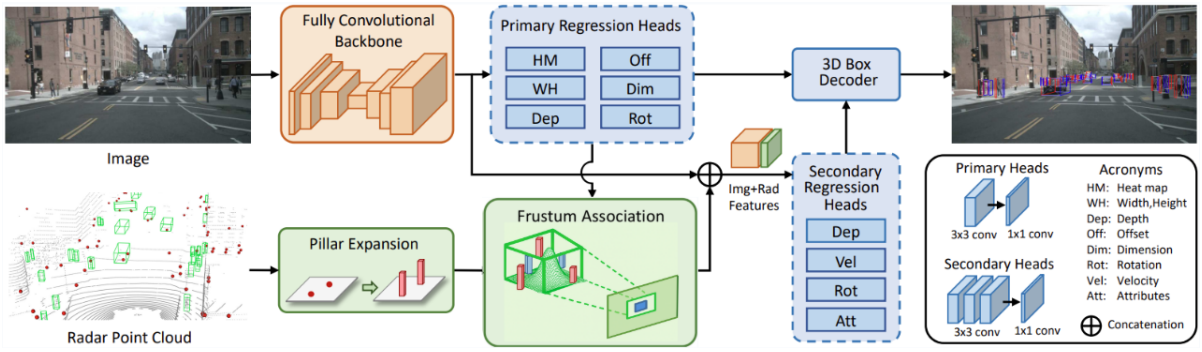


Figure 3.2: Overview of the GaussianFusion architecture.

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_z\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left[\left(\frac{x-\mu_x}{\sigma_x}\right)^2 + \left(\frac{z-\mu_z}{\sigma_z}\right)^2\right]\right) \tag{3.1}$$

The parameters of the distribution are defined as:

- μ_x: horizontal center of the 2D bounding box $(x_{\min} + x_{\max})/2$
- μ_z: predicted depth from the monocular 3D object detection module
- σ_x^2: variance in the x-direction, set to $\alpha \cdot (x_{\max} - x_{\min})^2$ (α is a scale factor)
- σ_z^2: variance in the z-direction, set to $\alpha \cdot (z_{\max} - z_{\min})^2$ (α is a scale factor)
- ρ: correlation coefficient between x and z, set to 0 (assuming independence)

While Equation 3.1 represents the canonical form of a normalised bivariate Gaussian PDF, the implementation for Gaussian-weighting can be simplified by omitting the normalisation constant and computing only the exponential term in the Gaussian [49, 50]:

$$w(x,z) \propto \exp\left(-\frac{1}{2}\left[\left(\frac{x-\mu_x}{\sigma_x}\right)^2 + \left(\frac{z-\mu_z}{\sigma_z}\right)^2\right]\right) \tag{3.2}$$

Next, the weights are normalised by dividing by the total weight:

$$\tilde{w}_i = \frac{w(x_i, z_i)}{\sum_j w(x_j, z_j)} \tag{3.3}$$

Since the weights are normalised in a subsequent step to ensure they sum to 1, the omitted constant does not affect the final weighted result. This approach allows for more efficient computation without sacrificing correctness in terms of radar point influence. This procedure is summarised in Algorithm 2 in Appendix A.1.

An additional advantage of using a Gaussian-weighted fusion strategy in GaussianFusion is that it enables the use of a significantly larger frustum volume during radar association. In CenterFusion, the frustum must remain relatively narrow to avoid incorrect associations caused by selecting the single nearest radar point. On the contrary, GaussianFusion considers all radar points within the frustum. By applying a weighting scheme, an artificial radar point is generated that is more robust to noise, sparsity, and frustum misalignment.

This Gaussian-based approach is not only more robust to noisy radar returns, but also enables the use of a greater expanded frustum without sacrificing association quality. The horizontal expansion is governed by a hyperparameter called ϵ, which enlarges the frustum relative to the bounding box. It is particularly helpful in cluttered or sparse environments, where a broader inclusion zone increases the likelihood of capturing relevant radar measurements. Moreover, it provides resilience to slight temporal misalignment between radar and camera data, which is a common issue when integrating marine radar systems and cameras, as discussed in Section 1.1.

The frustum association mechanism in GaussianFusion can further be tuned using two other key parameters. The first is α, which is a scale factor that determines the variance of the Gaussian distribution. Smaller values concentrate weights near the predicted center, while larger values allow a broader inclusion of surrounding radar points. The second is δ, which controls depth-wise expansion of the frustum.

This simple yet effective modification enables GaussianFusion to leverage radar data more flexibly and robustly than CenterFusion, particularly in scenarios with sparse radar returns or uncertain monocular predictions.

3.2 GaussianFusion++

GaussianFusion++ extends the GaussianFusion architecture by introducing a scene-adaptive radar association mechanism. The key novelty in this architecture is the RGPNet, a lightweight and interpretable network proposed in this thesis. RGPNet replaces the LFANet module used in CenterFusion++, which relies on a black-box radar encoder. Instead of processing raw radar snapshots, RGPNet directly learns the parameters of a bivariate Gaussian distribution from the radar points within each frustum. This enables a more interpretable and uncertainty-aware association strategy that adapts to the spatial structure of each scene.

While GaussianFusion improves robustness by applying fixed Gaussian weighting to radar points within a frustum, its parameters are statically derived from monocular predictions and cannot adjust to variations in radar point distribution or scene complexity. This can result in suboptimal radar–image fusion, particularly in environments with large depth variation or high sensor noise.

To address this limitation, GaussianFusion++ uses RGPNet to estimate Gaussian parameters dynamically from the observed radar points. This enables scene-dependent weighting based on proximity to a predicted mean, improving the model's ability to handle clutter, sparsity, and uncertainty more effectively.

As in GaussianFusion, a frustum enlargement strategy is applied to increase robustness during radar association. The parameter ϵ controls horizontal expansion relative to the 2D bounding box width, while δ governs depth-wise enlargement. These parameters ensure that relevant radar points are captured even under imperfect predictions or temporal misalignment.

3.2.1 Architecture

GaussianFusion++ uses an early fusion strategy in which radar and image features are combined at the input level. This approach allows the backbone network to learn joint representations from the beginning of the pipeline, which improves robustness in environments where visual information may be degraded or ambiguous.

At the core of the radar association mechanism is RGPNet. This network takes as input the radar points within the frustum, defined by the monocular 2D and 3D predictions, and estimates the parameters of a bivariate Gaussian distribution: μ_x, μ_z, σ_x, σ_z, and ρ. These parameters are used to compute a weighted combination of the radar points, producing a single fused radar point. This artificial point is then embedded into a radar feature map and combined with the image feature map. The fused representation is subsequently processed by the regression heads for final object prediction. An overview of the architecture is shown in Figure 3.3.

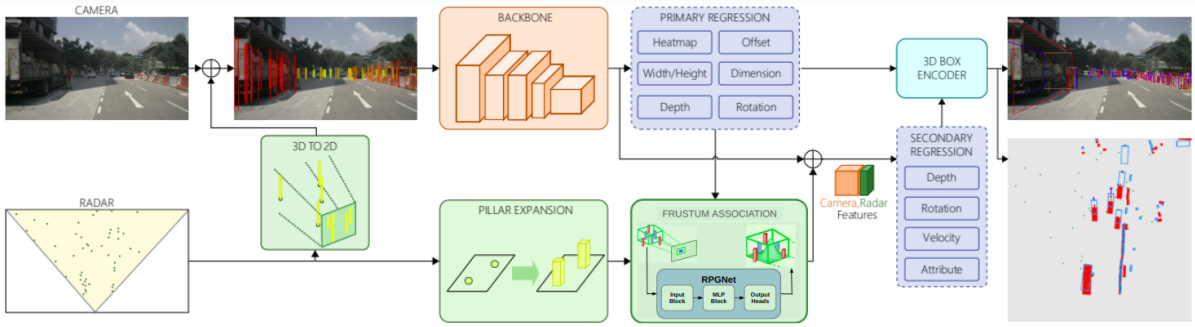


Figure 3.3: Overview of the GaussianFusion++ architecture.

3.2.2 RGPNet

RGPNet is a multilayer perceptron (MLP)-based regression network [51]. Its goal is to estimate bivariate Gaussian parameters from the radar points within a frustum. Two variants were constructed: RGPNet-A and RGPNet-B. Both variants are inspired by point-based architectures, like PointNet [52]. Each radar point is represented by its horizontal coordinate projected in the camera image (x) and real-world depth (z), resulting in a 2-dimensional input channel per radar point. RGPNet-A's architecture, visualised in Figure 3.4, applies a shared MLP independently

to each radar point's feature vector. This channel-mixing structure transforms each point's input features while preserving permutation invariance and enabling the network to process unordered sets of variable size [52]. The shared MLP consists of two fully connected layers and produces a per-point embedding of dimensionality $C = 64$.

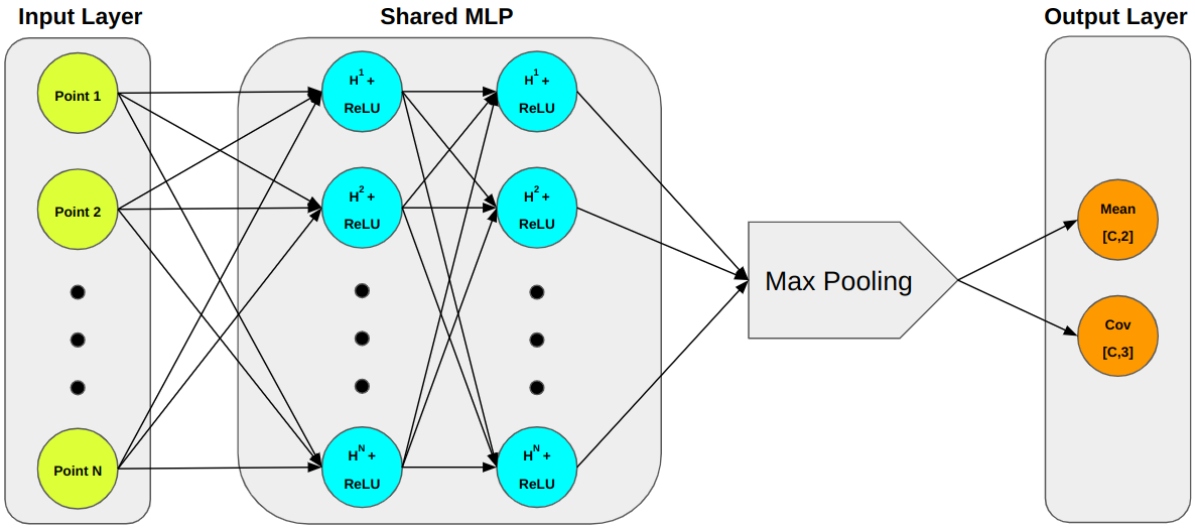


Figure 3.4: Overview of the RGPNet-A architecture. N refers to the variable number of radar points. C refers to the number of channels. H^i refers to the hidden state for radar point i.

Each linear layer is followed by a ReLU activation function [53]. ReLU effectively zeroes out negative values in the feature vectors passed into it. This allows the network to learn complex, non-linear patterns, and improves generalisation and computational efficiency. Furthermore, ReLU is well-suited for Kaiming initialisation and efficient optimisation [54]. Optionally, batch normalisation [55] and dropout [56] can be applied after each linear layer; these are configurable via training parameters. However, they are omitted in the final model analysed in this thesis, which solely includes linear transformations and ReLU activations, to maintain architectural simplicity. Moreover, the effectiveness of batch normalisation has been shown to diminish in shallow networks, where its impact on convergence and generalisation is typically limited.

After point-wise processing, the resulting feature tensor has shape $[N, C]$; where N is the number of radar points in the frustum and $C = 64$ is the dimensionality of the learnt point-wise feature embeddings. A global feature vector is then obtained by applying a max pooling operation across the N points, reducing the representation of the $[N, C]$-dimensional feature tensor to a single C-dimensional feature vector. This global descriptor captures the most salient features across the radar point set and is used as input to two output heads:

- **Mean Head:** Predicts the Gaussian mean (μ_x, μ_z), which represents the estimated center of the radar point distribution within the frustum:
  - μ_x is the predicted horizontal position of the object's center in image coordinates (typically aligned with the center of the 2D bounding box).
  - μ_z is the predicted depth of the object, corresponding to the expected distance from the sensor.
- **Covariance Head:** Predicts $\log(\sigma_x)$, $\log(\sigma_z)$, and $\tanh(\rho)$, which are transformed representations of the 2D covariance matrix parameters:
  - $\log(\sigma_x)$ and $\log(\sigma_z)$ are the natural logarithms of the standard deviations in the horizontal and depth directions, respectively. Predicting the logarithms directly ensures that the corresponding standard deviations σ_x and σ_z are strictly positive, which is required for a valid Gaussian distribution.
  - $\tanh(\rho)$ is the hyperbolic tangent of the correlation coefficient ρ, which constrains its value to the valid range $(-1, 1)$. This helps ensure that the predicted covariance matrix is numerically stable and remains positive semi-definite during training.

RGPNet-B, visualised in Figure 3.5, incorporates a token-mixing mechanism inspired by the MLP-Mixer architecture [57], enabling the model to learn interactions between different radar points within the frustum. To support

this, the number of radar points N per frustum is fixed across all inputs. Token-mixing treats each radar point as a token in a sequence and applies linear transformations across the token dimension. Such operations require a consistent input shape, both for architectural compatibility and for efficient batch processing on modern accelerators such as Graphics Processing Units (GPUs). In this work, $N = 128$. This size balances spatial coverage with computational efficiency. It retains enough points to preserve the geometry of individual objects and the diversity of radar reflections across the frustum, while limiting the influence of densely clustered regions and preventing over-representation of noisy returns.

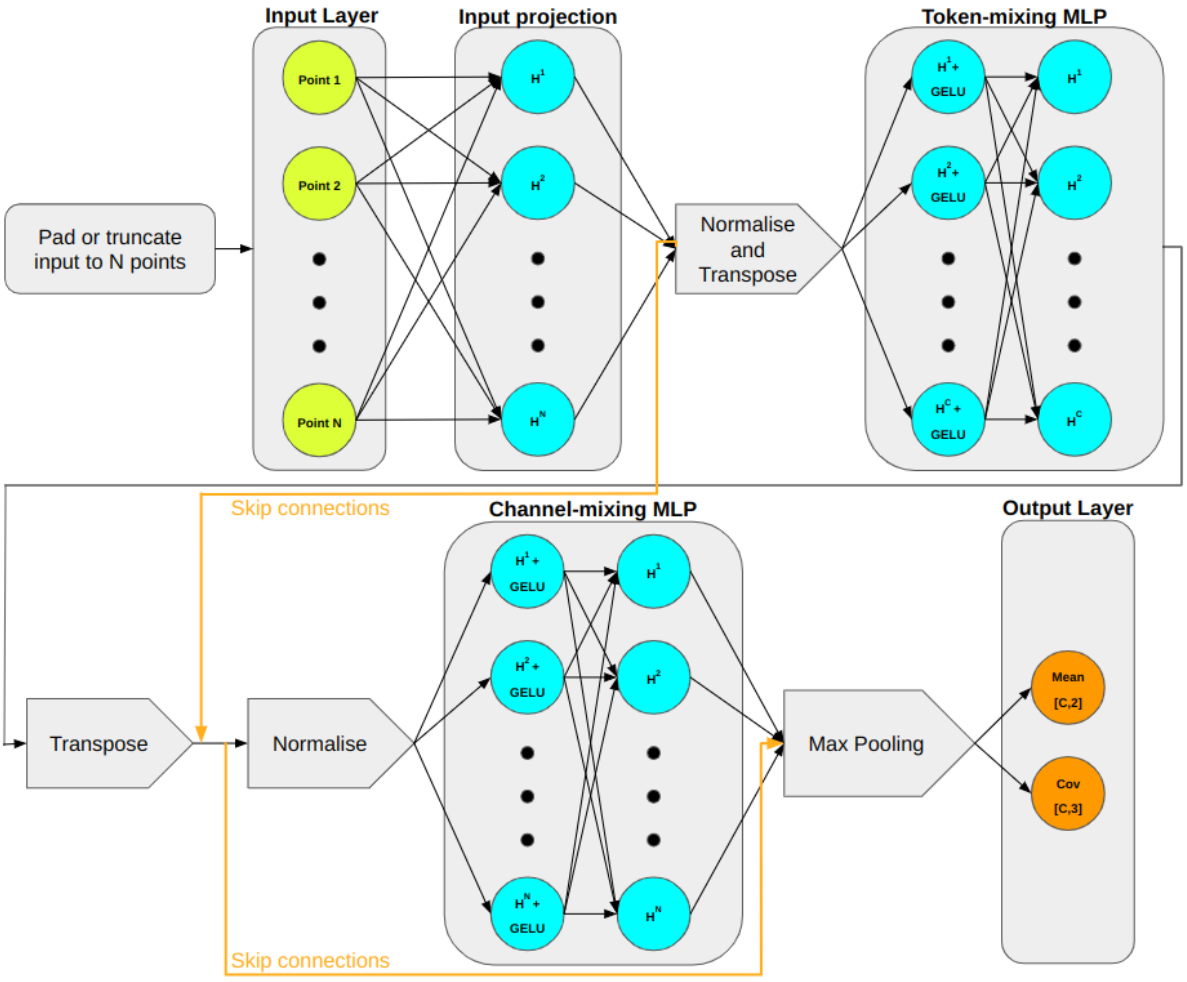


Figure 3.5: Overview of the RGPNet-B architecture. N refers to the fixed number of radar points. C refers to the number of channels. H^i refers to the hidden state for radar point i.

To enforce the fixed size $N = 128$, radar point sets are either truncated or padded to a predefined number of tokens. If fewer than N points are available, zero-padding is applied, and a binary mask records which points are real. If there are more than $N = 128$ points, the set is downsampled using Farthest Point Sampling (FPS). FPS is a deterministic and geometry-aware sampling technique that selects a subset of points such that the selected points are as far apart from each other as possible. This encourages spatial diversity and avoids redundant sampling in dense regions. Compared to random sampling, FPS better preserves the shape and structure of the point cloud, especially along edges, corners, and sparse regions. FPS is commonly used in point cloud processing tasks and was introduced in the PointNet++ architecture [58].

Each radar point's (x, z) coordinate is then projected into a higher-dimensional embedding space using a linear layer, lifting the raw 2D input into a representation suitable for learning complex spatial patterns. The resulting tensor has shape $[N, C]$, where $C = 64$ is the channel dimension. Layer normalisation is applied across the hidden dimension to stabilise training by standardising feature distributions, thereby mitigating internal covariate shift [55].

To perform token-mixing, the tensor is transposed from $[N, C]$ to $[C, N]$, so that linear layers can operate across the token dimension N. The token-mixing MLP comprises two linear layers, each mapping N to N, separated by

a Gaussian Error Linear Unit (GELU) activation [59]. The GELU [59] is a smooth, non-linear activation function that combines the benefits of ReLU and sigmoid activations. By allowing small negative values to pass through and preserving non-linearity, GELU improves learning dynamics in deep MLP-based models such as MLP-Mixer [57]. The token-mixing operation enables the network to learn point-to-point interactions across the frustum, capturing spatial dependencies that are not modelled in channel-mixing architectures such as PointNet [52].

After token-mixing, the tensor is transposed back to $[N, C]$, and layer normalisation is again applied. Each point is processed independently through a channel-mixing MLP, composed of two linear layers that map C to C. These layers are separated by another GELU activation. This enhances each point's representation without introducing cross-point interactions. By combining token- and channel-mixing the expressive capacity of the model is enhanced while remaining computationally lightweight and avoiding the complexity of convolutional or attention-based alternatives.

To further stabilise training and facilitate gradient flow, skip connections are employed in both the token-mixing and channel-mixing stages. These residual connections add the input of each MLP block back to its output, allowing the model to learn perturbations around the identity function. This helps mitigate the problem of vanishing gradients, especially in deeper architectures, and enables the network to retain useful low-level features from earlier layers. Skip connections are a well-established technique in deep learning and were first introduced in the ResNet architecture [60]. In the context of MLP-based architectures, such as MLP-Mixer, residual connections have been shown to be essential for effective training and convergence [57].

To ensure that padded radar points do not influence the final output, a binary mask is used to zero out their contributions. Specifically, after the token- and channel-mixing stages, the masked feature tensor is set to a large negative value (e.g., -10^9) at all padded positions. Identical to RGPNet-A, a global max pooling operation is then applied to reduce the $[N, C]$ tensor to a single C-dimensional feature vector per frustum. This global descriptor is passed to the outputs heads to predict the Gaussian parameters.

Finally, the predicted Gaussian parameters are used to weight radar points in the frustum and compute a fused artificial radar point. This point is embedded in a radar feature heatmap at the object's location and concatenated to the camera features before being used for downstream regression.

Compared to LFANet in CenterFusion++ [38], RGPNet provides a lighter and more interpretable alternative. By explicitly modelling the spatial distribution of radar points within a frustum using a bivariate Gaussian distribution, it enables more robust radar association in cluttered or noisy scenes. Additionally, the use of predicted uncertainty, in the form of covariance parameters, allows the system to adapt its association strategy based on scene complexity, offering improved generalisation over heuristic or black-box methods.

In summary, RGPNet encodes unordered radar point sets into a compact, Gaussian-parameterised representation using MLP layers, global max pooling, and dual-headed output prediction. This design makes it well-suited for downstream sensor fusion and regression tasks in frustum-based 3D perception pipelines.

3.2.3 Loss function

RGPNet is trained using a composite loss function that supervises two objectives: (1) estimating a radar-based feature heatmap encoding object depth, and (2) learning the parameters of a bivariate Gaussian distribution that captures the spatial distribution of radar points within a frustum. The heatmap loss provides indirect supervision to the Gaussian prediction by penalising inconsistencies in the generated radar feature map, while the Gaussian Negative Log-Likelihood (NLL) loss provides direct supervision by explicitly training the predicted distribution to match the annotated object locations. The total loss is defined as:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{NLL}} \cdot \mathcal{L}_{\text{NLL}} + \lambda_{\text{HM}} \cdot \mathcal{L}_{\text{HM}} \tag{3.4}$$

where the weighting factors are set to $\lambda_{\text{HM}} = 0.8$ and $\lambda_{\text{NLL}} = 0.2$. Since RGPNet is ultimately designed to produce accurate radar feature heatmaps for downstream fusion, the heatmap loss is given a heavier weight. It plays a more central role in guiding RGPNet to produce useful spatial features and indirectly supervises the Gaussian parameters by ensuring the resulting radar heatmap is spatially aligned with the ground truth. Additionally, the Gaussian NLL loss tends to be more sensitive and numerically less stable due to operations like matrix inversion and log-determinants. A lower weight helps stabilise training while still allowing the model to learn meaningful uncertainty estimates.

The heatmap produced by RGPNet represents radar-derived features, such as depth or velocity, encoded into a 2D spatial grid aligned with image coordinates. This heatmap serves as an intermediate representation that is fused with image features to improve final 3D object predictions.

The supervision of this radar heatmap follows a similar approach to that used in LFANet [38], where an L1 loss is applied between the predicted heatmap $\hat{H}$ and a ground truth reference H:

$$\mathcal{L}_{\text{HM}} = \frac{1}{N} \sum_{i=1}^{N} \left| \hat{H}_i - H_i \right| \tag{3.5}$$

Here:

- N is the total number of pixels in the heatmap.
- $\hat{H}_i$ and H_i are the predicted and ground truth radar values at pixel i, respectively.

The L1 loss, also known as mean absolute error, encourages sharper and more localised predictions compared to L2 loss, which tends to blur out predictions [61]. This is especially useful in radar settings where the feature maps are sparse and must retain spatial specificity.

Importantly, this loss provides indirect supervision to the Gaussian parameter prediction. In this way, the model is encouraged to learn Gaussian parameters that produce feature maps closely resembling the true spatial structure of radar returns, without explicitly supervising the parameters themselves.

The Gaussian NLL loss directly supervises the predicted Gaussian distribution over radar points within each object's frustum. RGPNet outputs five parameters: mean coordinates (μ_x, μ_z) and the covariance matrix:

$$\Sigma = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_z \\ \rho\sigma_x\sigma_z & \sigma_z^2 \end{bmatrix} \tag{3.6}$$

The ground truth mean $\mathbf{g} = [g_x, g_z]$ is computed from the 2D bounding box center and annotated object depth. The Gaussian NLL is given by:

$$\mathcal{L}_{\text{NLL}} = \frac{1}{2} \left[\log |\Sigma| + (\mathbf{g} - \boldsymbol{\mu})^\top \Sigma^{-1} (\mathbf{g} - \boldsymbol{\mu}) \right] \tag{3.7}$$

Here:

- $\boldsymbol{\mu} = [\mu_x, \mu_z]$: predicted mean of the bivariate Gaussian.
- Σ: predicted covariance matrix encoding uncertainty and correlation.
- $\log |\Sigma|$: penalises high uncertainty (i.e., large variance), encouraging precise distributions.
- The Mahalanobis distance [62] $(\mathbf{g} - \boldsymbol{\mu})^\top \Sigma^{-1} (\mathbf{g} - \boldsymbol{\mu})$: penalises predictions that deviate from the ground truth while taking the predicted uncertainty into account.

A small value ϵ is added to the diagonal of Σ to ensure numerical stability and positive definiteness during inversion.

Each loss serves a distinct but complementary role in training RGPNet:

- **Heatmap Loss (L1)** ensures that the radar heatmap generated from the predicted Gaussian parameters aligns with the spatial structure expected from ground truth. It acts as indirect supervision on the Gaussian parameters, encouraging RGPNet to learn parameters that yield a useful intermediate representation.
- **Gaussian NLL Loss** directly supervises the predicted Gaussian parameters to ensure they meaningfully represent the underlying radar point distribution. This is crucial for uncertainty modelling, where the model learns to express confidence through the predicted variance components of the distribution [63]. Without this supervision, the model might predict degenerate Gaussians, e.g. overly sharp ones that place all mass near the object center or overly diffuse ones that spread uncertainty too broadly. that "cheat" to get plausible-looking heatmaps. These configurations can "cheat" the heatmap loss by producing plausible-looking outputs while failing to faithfully model the underlying uncertainty.

Together, these losses help RGPNet learn both to understand the distribution of radar data and to draw a useful representation of it for downstream fusion. If only one loss is used, the model may either: (1) Produce heatmaps that look plausible but are backed by nonsensical Gaussian parameters (without Gaussian NLL), or (2) Learn Gaussian parameters that are statistically sound but fail to produce useful spatial features for fusion (without heatmap loss). Therefore, combining both losses ensures the predicted Gaussian is both correct in theory (Gaussian NLL) and useful in practice (heatmap).

Crucially, $\mathcal{L}_{\text{NLL}}$ is only computed during the training phase for valid frustums, defined as those where a ground truth object center lies within the frustum. If no such frustums exist in a batch, the total loss is reduced to the heatmap loss alone.

3.3 Dynamic frustum enlargement

In the original CenterFusion model [37], frustum construction is guided by the 2D bounding box, and monocular depth, size and orientation estimation. A frustum is then extended using a fixed depth expansion parameter, denoted as δ. To improve the chance of capturing relevant radar returns, especially under uncertainty in depth predictions and partial temporal misalignment between the sensors, the frustum is horizontally enlarged through a parameter called ϵ. The horizontal enlargement factor is $1 + \epsilon$, and the depth-wise enlargement factor is $1 + \delta$. This simple yet effective strategy increases the likelihood that radar returns corresponding to the detected object fall within the frustum volume, which is crucial for the downstream radar association process.

While this fixed enlargement may improve robustness if used correctly, it applies the same degree of expansion to all objects, regardless of their distance from the sensor. This presents a limitation. Empirically, the error in monocular depth estimation tends to increase with distance [64]. As objects get farther away, the accuracy of the predicted depth diminishes due to lack of visual cues and diminishing pixel resolution. Consequently, a fixed-size frustum may no longer be large enough to include relevant radar points for distant objects, leading to poor association and degraded 3D detection performance. On the other hand, excessively increasing the frustum's size can lead to the inclusion of irrelevant radar returns, increasing false positives and degrading detection precision.

Conversely, for objects that are very close to the sensor, monocular depth estimation is typically more reliable. Using an unnecessarily large frustum in such cases increases the chance of including unrelated radar returns, introducing noise into the fusion process. Hence, there is a strong motivation to design a frustum enlargement strategy that adapts to the estimated object depth.

To address this issue, an adaptive frustum enlargement module, one of the key novelties of this thesis, is proposed. Rather than applying a constant horizontal and depth-wise frustum expansion, this module computes the horizontal and depth-wise frustum expansions as a function of the predicted depth. The goal is to dynamically increase the frustum size for distant objects, where depth predictions are generally less accurate, while keeping it compact for nearby objects to retain precision.

Empirical studies and benchmark analyses, such as those conducted on the KITTI dataset [12], have shown that the error in monocular depth estimation grows superlinearly, and in many cases, approximately quadratically with the actual distance to the object [65]. This behaviour stems from the geometric properties of monocular vision systems, where small inaccuracies in pixel space translate into increasingly large metric errors as depth increases. Furthermore, the scarcity of texture, reduced pixel resolution, and diminished parallax cues at long ranges all contribute to compounding depth uncertainty. To account for this non-linear degradation in accuracy, the adaptive frustum enlargement function is defined to grow quadratically with the predicted depth, as shown in Equation 3.8:

$$\delta = \delta_{base} + k \cdot depth^2 \tag{3.8}$$

The same function is used for horizontal and depth-wise frustum enlargement, though the scaling factor k may be tuned separately depending on the axis. This formulation ensures minimal frustum expansion for close-range objects, where depth predictions are generally reliable, while aggressively enlarging the frustum for distant objects, where greater tolerance is needed to accommodate uncertainty and ensure successful radar-point association. The scaling parameter k controls how rapidly the frustum expands with depth and must be chosen carefully to prevent excessive enlargement at mid-range distances. In practice, k should be very small, typically around 10^{-6}, to ensure that the frustum expansion does not grow uncontrollably at large depths. This preserves precision at short range while still accommodating the increased uncertainty typical of distant objects.

Chapter 4
Experimental setup

This chapter outlines the experimental setup used to train and evaluate the proposed 3D object detection models. Given the limited size and domain specificity of the maritime dataset, careful attention was paid to both the training strategy and the evaluation protocol. The first section details the multi-phase training procedure, including transfer learning, staged learning, and multi-task optimisation. The second section presents the evaluation metrics used to assess model performance in both 2D and 3D domains. Together, these components form the foundation for reproducible and meaningful comparison of model variants under realistic maritime conditions.

4.1 Training strategy

To effectively train the proposed 3D object detection models in a data-scarce maritime setting, a carefully designed multi-phase training strategy is necessary. This strategy consists of three key components: pretraining on a custom 2D dataset [18], staged learning [41], and multi-task co-adaptation [21]. Each step is motivated by the unique challenges associated with maritime perception and the structural dependencies between 2D detection and 3D regression.

Given the limited size of the custom 3D dataset, pretraining was essential to initialise the backbone and 2D detection heads with strong feature representations. To this end, a CenterNet model with a DLA-34 backbone was trained on a custom 2D dataset provided by RH Marine [39]. This dataset exclusively contains images of maritime objects, such as vessels, acquired in realistic settings. It comprises 5624 images with 15214 annotations, split into a training set with 3831 images and 9097 annotations, and a validation set with 1793 images and 6117 annotations.

Before training on this dataset, the CenterNet model was initialised using weights from a version of CenterNet pretrained on the COCO dataset [40]. From this pretrained model, the DLA-34 backbone and the output heads relevant for detecting boats were extracted and reused. This transfer learning approach provides a strong initialisation not only for low- and mid-level visual features in the backbone, but also for the task-specific outputs of the detection heads. Since the COCO dataset includes a wide variety of object categories, including watercraft, this allowed the model to start from weights already attuned to maritime object characteristics, thereby accelerating convergence and improving performance on the custom ship-focused dataset.

CenterNet was trained using early stopping [66] based on validation loss, and the final weights of its backbone and 2D detection heads were subsequently used to initialise the CenterFusion [37], GaussianFusion, CenterFusion++ [38], and GaussianFusion++ models. The purpose of this pretraining stage was not to maximise performance on the 2D dataset itself, but rather to ensure that the initialised model had learned to extract meaningful visual features. This is critical because the downstream 3D detection heads rely on robust 2D detections to guide learning 3D characteristics, such as depth, orientation, size and velocity [17].

However, pretraining alone is not sufficient. The backbone must ultimately learn to extract features that are jointly useful for both 2D and 3D tasks. To achieve this, a staged learning scheme was employed, which gradually shifts the model's focus from 2D to 3D. In the first phase, the backbone is frozen for 5 epochs and the model concentrates on refining the 2D heads. During this phase, the weights of the 2D loss terms are increased by a factor of 2, while the weights of the 3D loss terms are decreased by a factor of 10. This allows the model to fine-tune 2D performance on the 3D dataset before introducing the complexity of 3D regression.

After 5 epochs, the second training phase begins, encompassing the next 15 epochs. Here, the backbone is unfrozen, and the model starts learning shared representations for both tasks. The 2D losses are reduced by a factor of 2 compared to phase one, and the 3D losses are increased by a factor of 5. Although the backbone is now updated by both tasks, the emphasis remains on 2D detection to ensure that spatial localisation in the image remains strong.

In the final phase, which begins after epoch 20, the training emphasis shifts toward 3D feature learning. The 3D losses are increased by a factor of 4 compared to phase two, and the 2D losses are halved again. At this stage, the model focuses primarily on refining its 3D predictions. This phased approach ensures that the model first masters the simpler task of 2D detection, which provides the foundation for learning more complex spatial representations in 3D.

Throughout all stages, a multi-task co-adaptation training paradigm is applied [21]. The model is trained end-to-end, with both the 2D and 3D loss gradients contributing to the optimisation of shared layers. This encourages the network to develop representations that are jointly informative for both types of tasks, improving robustness and efficiency.

Finally, early stopping, based on the nuScenes Detection Score (NDS) metric, is applied in the final phase to prevent overfitting. NDS is a comprehensive metric that balances detection accuracy and localisation quality, making it well suited for multi-task models in complex, real-world environments such as ports [14].

To validate the importance of the proposed multi-phase training strategy, CenterFusion was trained for 40 epochs both with and without it. When trained without pretraining or staged learning, the model achieved an mean Average Precision (mAP) of only 0.1405. In comparison, training CenterFusion with the full strategy, which includes pretraining on a maritime 2D dataset and staged loss weighting, resulted in a significantly higher mAP of 0.4165. Since mAP reflects the model's ability to detect objects accurately and consistently, achieving a high value is essential for robust obstacle detection. This substantial difference in performance confirms the critical role of the proposed training strategy in data-scarce maritime environments.

4.2 Evaluation metrics

The performance of the proposed models is evaluated both qualitatively and quantitatively. The qualitative evaluation consists of visualising the predictions by overlaying 3D bounding boxes on the original camera images and rendering the predicted objects in BEV. This allows for a visual assessment of spatial accuracy and can reveal systematic failure cases that are not evident from numerical scores alone.

The quantitative evaluation includes both 2D and 3D metrics. For the 2D task, Average Precision (AP) is reported [67]. This metric summarises the detector's precision and recall performance across a range of confidence thresholds. Each predicted object is assigned a confidence score between 0 and 1, representing the model's certainty that an object is present. By varying this threshold, one can trade off between precision (how many detections are correct) and recall (how many ground truth objects are detected).

To compute these metrics, we define three quantities: true positives (TPs), false positives (FPs), and false negatives (FNs). A predicted bounding box is considered a TP if its Intersection over Union (IoU) with a ground truth bounding box exceeds a given threshold. Predictions not matched to any ground truth, based on this threshold, are counted as FPs. Ground truth objects without any matched predictions are counted as FNs. The IoU threshold used to define TPs can vary, typically ranging from 0.5 to 0.95 in steps of 0.05 when computing COCO-style AP. Precision and recall are defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.2}$$

To compute AP, the model's predictions are first sorted by decreasing confidence score. The precision and recall are then calculated at each threshold, forming a precision–recall curve. AP is defined as the area under this curve, providing a single scalar value that reflects the model's detection performance across all operating points:

$$\text{AP} = \int_0^1 \text{Precision}(r)\, dr \tag{4.3}$$

In this work, AP is reported following the COCO convention [40], averaging AP over multiple IoU thresholds ranging from 0.50 to 0.95 in steps of 0.05. This provides a more rigorous evaluation of localisation quality than using a single fixed threshold. Since the custom dataset involves a single object class, this averaged AP value is equivalent to mAP. It is important to note that this definition of mAP differs from the one used in the official nuScenes benchmark [14]. The nuScenes mAP is calculated using center distance thresholds in BEV and averaged over multiple object classes. Therefore, this work's mAP is not directly comparable to the nuScenes benchmark, although both aim to capture detection quality.

For the 3D task, four TP metrics are reported: Avergae Translation Error (ATE), Average Scale Error (ASE), Average Orientation Error (AOE), and Average Velocity Error (AVE). These are calculated only for matched detections. If a detection is assigned to a ground truth bounding box with an IoU of at least 0.5, then these are considered a match. Matching is performed greedily, meaning that predictions are sorted by confidence score and each is matched to the highest IoU ground truth bounding box that has not already been matched, provided

the IoU is above the threshold. This ensures a one-to-one correspondence between predictions and ground truths. ATE measures the unbounded Euclidean distance between predicted and ground truth centers. ASE quantifies the relative error in predicted object size after aligning orientation and translation ($1 - 3$D IoU). AOE captures the yaw angle error in radians, and AVE measures the unbounded Euclidean difference in velocity vectors. The Average Attribute Error (AAE), typically included in the nuScenes benchmark, is excluded here since the model does not predict object attributes.

To evaluate how detection performance varies with distance, the TP metrics are stratified across three depth intervals: 0–100 m, 100–250 m, and beyond 250 m. This stratification provides insight into how detection quality changes with distance, particularly when adjusting frustum enlargement parameters.

As an overall performance measure, NDS is used, which aggregates detection accuracy and localisation quality into a single value [14]. In the nuScenes definition, NDS combines mAP with the mean of the five 3D metrics over all classes, described as:

$$\text{NDS} = \frac{1}{10}\left[5 \cdot \text{mAP} + \sum_{\text{mTP} \in \mathbb{TP}} (1 - \min(1, \text{mTP}))\right] \tag{4.4}$$

Here, mAP refers to the nuScenes mAP and $\mathbb{TP} = \{\text{mATE}, \text{mASE}, \text{mAOE}, \text{mAVE}, \text{mAAE}\}$ includes all TP metrics averaged over all classes, used in the nuScenes benchmark. For calculating NDS, all TP metrics are bounded between 0 and 1 before being subtracted from 1. Half of NDS is based on the detection performance and the other half is based on the quality of the TP metrics.

Since this work excludes attribute classification and solely consists of single class objects, an adjusted NDS formula is computed:

$$\text{NDS} = \frac{1}{8}\left[4 \cdot \text{mAP} + \sum_{\text{TP} \in \mathbb{TP}} (1 - \min(1, \text{TP}))\right] \tag{4.5}$$

Here, mAP refers to this work's mAP, and $\mathbb{TP} = \{\text{ATE}, \text{ASE}, \text{AOE}, \text{AVE}\}$ includes the four TP metrics used in this work. To ensure comparability within the NDS calculation, all TP metrics are bounded between 0 and 1 before being subtracted from 1. However, the normalisation procedure differs slightly from the original nuScenes benchmark. ATE can become much larger compared to the nuScenes benchmark due to the extended maximum range of 1654.8 m and the superlinear degradation of depth estimation at greater distances [42, 65]. Therefore, ATE is clipped to the range [0, 100] and then divided by 100 to keep it within [0, 1]. AVE is normalised by dividing by the 9.4, which is the maximum velocity in the dataset. Similarly, AOE, which ranges from 0 to π radians, is normalised by dividing by π. Half of the adjusted NDS is based on the detection performance, and the other half reflects the quality of the TP metrics. This adjusted NDS captures a model's combined performance on object detection and 3D localisation quality, while remaining consistent with the IoU-based matching protocol.

While all the above metrics provide valuable insights into different aspects of detection quality, mAP is the most important metric used in this thesis. This is because it directly reflects whether objects are detected at all. If objects are not detected in the first place, then accurate localisation, scale estimation, orientation prediction, or velocity estimation becomes irrelevant. As such, mAP is used as the primary basis for comparison across methods. Nonetheless, the other metrics are also analysed where they reveal meaningful patterns or help explain performance differences not captured by mAP alone.

Chapter 5
Experiments

This chapter presents a series of controlled experiments aimed at evaluating the influence of key architectural and methodological modifications introduced in this work. Each experiment is designed to isolate a specific aspect of the detection pipeline and assess its impact on overall system performance under realistic maritime conditions. The evaluation is conducted using the same annotated dataset, metrics, and training procedures described in Chapters 2 and 4. In every experiment, the models were trained for 40 epochs.

The first experiment investigates the effect of fixed frustum enlargement parameters on radar–camera fusion performance. By training the four models under different configurations of horizontal and depth-wise enlargement, this experiment examines how increasing the frustum volume influences detection and localisation accuracy. Additionally, two ablation studies are conducted to investigated the effects of horizontal frustum enlargement.

The second experiment extends the frustum enlargement approach by introducing a dynamic module that adjusts the size of the frustum as a function of the predicted object depth. This depth-adaptive strategy is motivated by the observation that monocular depth estimates tend to degrade superlinearly with distance [65]. This means that it is highly probable that different depths have different optimal frustum enlargement values. A quadratic enlargement function, shown in Equation 3.8, is applied to both horizontal and depth axes, with the aim of increasing robustness to localisation errors at long range. All models are retrained with this dynamic module to evaluate whether adaptive frustum sizing leads to improved detection performance across different depth intervals.

The third experiment focuses on the role of uncertainty modelling in the Gaussian weighting process. Specifically, it compares the performance of GaussianFusion when trained with different values for scale factor α, which determines the variance of the Gaussian distribution within a frustum. A larger α results in a broader, flatter distribution, allowing radar points that are further from the predicted object center to contribute more to the construction of the artificial radar point. In contrast, a smaller α concentrates the weights more tightly around the predicted object center. By comparing different settings for α, this experiment seeks to understand how different levels of uncertainty affect localisation performance.

Finally, the fourth experiment explores the impact of introducing token-mixing in RGPNet, the radar-based parameter estimation network used in GaussianFusion++. Two variants are considered: RGPNet-A, which applies independent point-wise transformations, and RGPNet-B, which includes token-mixing layers to allow structured interactions between radar points within the frustum. The goal is to evaluate whether modelling inter-point dependencies improves the estimation of Gaussian parameters and contributes to more effective fusion and downstream prediction.

For all experiments, the models are trained for 40 epochs using the same pre-initialised CenterNet [39] backbone and the training strategy described in Section 4.1. Evaluation is performed on the fixed validation set using both aggregated and stratified metrics. Performance is tracked across three distance intervals: near ($< 100\,\text{m}$), mid (100–250 m), and far ($> 250\,\text{m}$). The evaluation metrics include mAP, ATE, ASE, AOE, AVE, and NDS, all defined in Section 4.2.

In addition to standard detection metrics, a threshold sweep analysis is conducted for the best-performing configuration of GaussianFusion++. This procedure evaluates all confidence thresholds from 0.01 to 0.99 to identify the threshold that maximises the F1 score, which balances precision (positive prediction accuracy) and recall (detection completeness). The corresponding optimal precision, recall, and F1 score values are reported to support intuitive interpretation and practical decision-making.

In Experiments 1 and 2 GaussianFusion++ is configured with the RGPNet-A variant, which does not incorporate token-mixing. This decision allows an initial assessment of whether the simpler point-wise architecture can adequately learn to estimate the Gaussian parameters for constructing the artificial radar point. The impact of introducing token-mixing in RGPNet-B is investigated separately in Experiment 4 to determine whether modelling inter-point dependencies further enhances the performance of GaussianFusion++.

Together, these experiments provide a structured and comprehensive analysis of the design decisions made in this thesis, offering empirical evidence for the effectiveness of proposed techniques in multi-modal 3D object detection for autonomous surface vessels.

5.1 Experiment 1: Static frustum enlargement

This experiment evaluates the impact of fixed frustum enlargement parameters on the performance of the four radar-camera fusion models: CenterFusion [37], GaussianFusion, CenterFusion++ [38], and GaussianFusion++. Frustum enlargement refers to expanding the spatial volume within which radar points are associated with image-based detections. As described in Chapter 3, this is controlled by two parameters: ϵ for horizontal enlargement and δ for depth-wise enlargement.

In addition to these parameters, a fixed scale factor $\alpha = 0.1$ is set to configure the variances used in the bivariate Gaussian weighting. Specifically, the variances are computed as $\sigma_x^2 = 0.1 \times (x_{\max} - x_{\min})^2$ and $\sigma_z^2 = 0.1 \times (z_{\max} - z_{\min})^2$. The chosen scale factor $\alpha = 0.1$ balances the trade-off between weighting radar points tightly near the predicted object center and allowing sufficient spread to tolerate moderate monocular prediction errors and sensor misalignments, which is critical for robust radar-camera association in maritime conditions.

The motivation for this experiment arises from two key observations. First, fixed-size frustums may fail to capture relevant radar points when monocular predictions are imprecise, particularly at long range. Second, in the custom dataset, temporal misalignment between radar and camera frames introduce spatial discrepancies. Horizontally widening the frustum may mitigate this effect by increasing the likelihood of capturing valid radar returns despite such misalignment.

To investigate these effects, each model was trained twice under four static frustum configurations:

- **Baseline:** $\epsilon = 0$, $\delta = 0$ (no enlargement);
- **Mild enlargement:** $\epsilon = 0.5$, $\delta = 1$;
- **Moderate enlargement:** $\epsilon = 2$, $\delta = 3$;
- **Strong enlargement:** $\epsilon = 5$, $\delta = 8$.

Additionally, two targeted ablation studies were conducted for GaussianFusion++ to better understand the independent effects of horizontal and depth-wise enlargement. The first ablation uses $\epsilon = 2$, $\delta = 0$ to isolate the role of horizontal widening alone, verifying whether lateral expansion meaningfully mitigates radar–camera misalignment. The second ablation uses $\epsilon = 2$, $\delta = 8$, matching the strongest depth enlargement while employing a moderate horizontal value instead of the extreme $\epsilon = 5$. This helps determine whether the highest horizontal expansion is excessive and provides guidance for setting balanced parameters in the dynamic frustum enlargement experiment.

This experiment is designed to test the hypothesis that GaussianFusion and GaussianFusion++, which use probabilistic radar point weighting, are more robust to excessive frustum enlargement and radar noise than CenterFusion and CenterFusion++.

5.1.1 Aggregated results

Figure 6.1 presents the detection performance across different static frustum enlargement settings, evaluated using mAP and NDS. For CenterFusion and GaussianFusion, the results remain relatively stable across all configurations, with minor variation in both mean values and standard deviations. This suggests that these models are less sensitive to static frustum enlargement in terms of detection accuracy. In contrast, CenterFusion++ and GaussianFusion++ exhibit more pronounced changes. Notably, GaussianFusion++ shows increased variance in mAP, particularly at the baseline and strong enlargement settings, indicating a higher sensitivity to frustum configuration in those regimes.

Figure 6.2 reports the TP metrics (ATE, ASE, AOE, AVE) across all static frustum enlargement configurations. The ATE results indicate that CenterFusion and CenterFusion++ are more sensitive to the choice of frustum parameters. For CenterFusion, the mean ATE remains relatively stable, but its standard deviation increases noticeably with stronger enlargement. CenterFusion++, by comparison, shows substantial variation in mean ATE across different configurations. GaussianFusion and GaussianFusion++, on the other hand, display greater robustness. In particular, GaussianFusion++ maintains a stable ATE mean across all settings.

The ASE metric displays a consistent trend across all configurations: CenterFusion and GaussianFusion perform similarly and less favourably, while CenterFusion++ and GaussianFusion++ consistently achieve lower ASE values. This suggests that the models with learnable frustum association mechanisms offer better alignment between predicted and ground truth box shapes, regardless of frustum size.

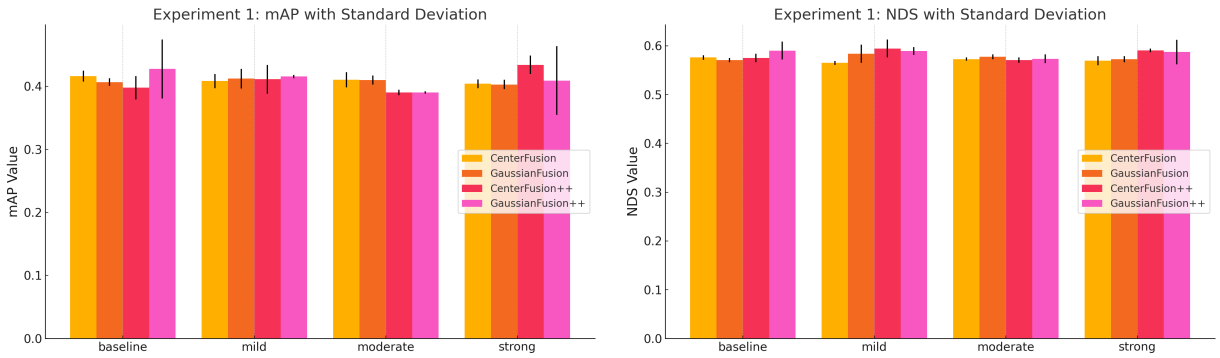


Figure 5.1: Aggregated detection performance metrics (mAP, NDS) for all models under different static frustum enlargement configurations (Baseline, Mild, Moderate, Strong). Each subplot visualises the trend for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++.

In contrast, the AOE and AVE metrics do not exhibit any clear or interpretable patterns in response to frustum enlargement. The observed fluctuations appear to be model-specific and lack consistent trends, implying that frustum size has a limited or unpredictable effect on orientation and velocity estimation in this setup.

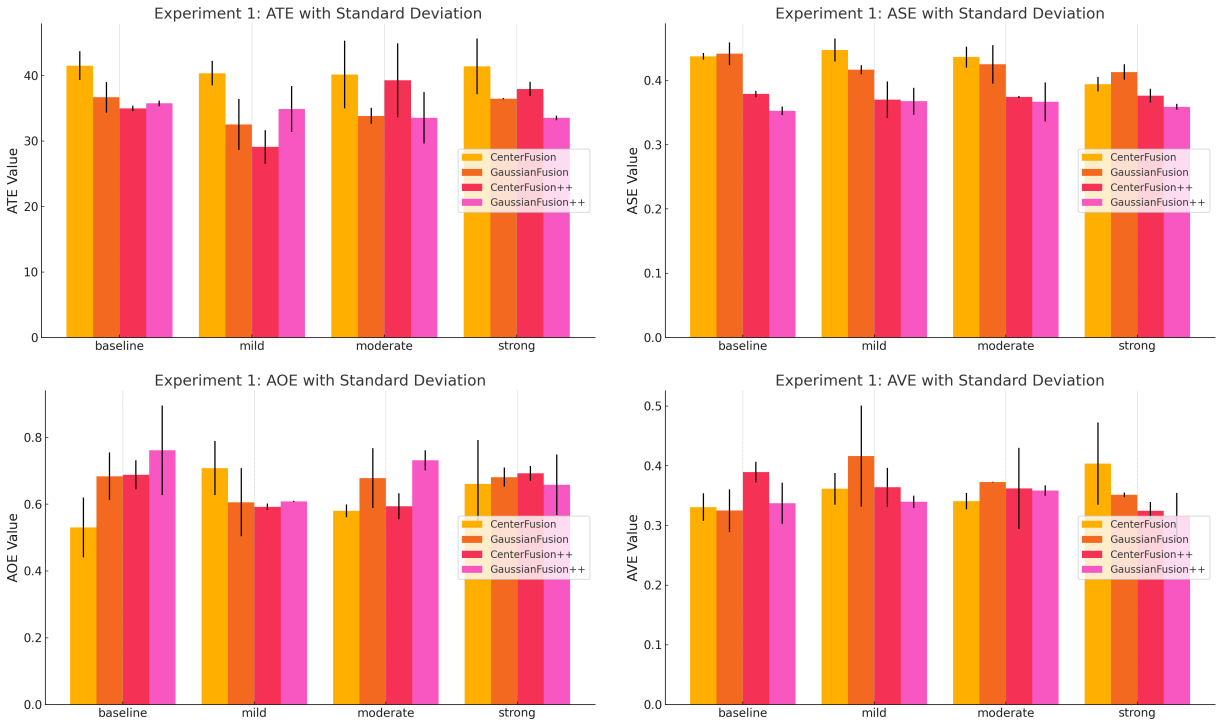


Figure 5.2: Aggregated TP metrics (ATE, ASE, AOE, AVE) for all models under different static frustum enlargement configurations.

The exact numerical values for each model and setting are reported in Tables A.2.1 to A.2.4 in Appendix A.2.1. Qualitative results are shown in Figures A.2.2 and A.2.9.

5.1.2 Stratified results

Table 5.1 shows the stratified ATE for each frustum setting. For the baseline ($\epsilon = 0$, $\delta = 0$), CenterFusion++ achieves the lowest ATE across all ranges. With mild enlargement ($\epsilon = 0.5$, $\delta = 1$), GaussianFusion has the lowest near ATE (11.60), while GaussianFusion++ achieves the lowest mid ATE (25.30). CenterFusion++ achieves the lowest far ATE (56.85). At moderate enlargement ($\epsilon = 2$, $\delta = 3$), CenterFusion++ achieves the lowest near ATE (12.67), while GaussianFusion++ achieves the lowest mid ATE (26.76). GaussianFusion has the lowest far

ATE (71.35). For strong enlargement ($\epsilon = 5$, $\delta = 8$), CenterFusion shows the lowest near ATE (11.60), while GaussianFusion++ achieves the lowest mid ATE (23.14) and far ATE (74.63).

Table 5.1: Stratified ATE of all four models under different static frustum enlargement settings. Metrics are stratified by distance: near (< 100 m), mid (100–250 m), and far (> 250 m).

ϵ	δ	CenterFusion ATE ↓			GaussianFusion ATE ↓			CenterFusion++ ATE ↓			GaussianFusion++ ATE ↓		
		Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
0.0	0	16.3046 ± 0.8533	31.0006 ± 1.5908	92.7482 ± 5.7646	15.5870 ± 1.3361	29.3846 ± 2.9892	76.7594 ± 6.1452	**12.5640** ± 0.6896	**27.3747** ± 1.6799	**76.1477** ± 3.4650	13.0389 ± 0.6715	27.4766 ± 4.8702	78.8957 ± 3.2615
0.5	1	16.8668 ± 3.4663	27.7418 ± 0.5417	94.4933 ± 13.8371	**11.5989** ± 0.2850	27.5696 ± 1.9105	67.7507 ± 12.4417	13.2966 ± 2.3926	25.4198 ± 0.4057	**56.8536** ± 6.7222	15.1290 ± 0.0395	**25.3007** ± 4.6962	76.4312 ± 9.6272
2.0	3	14.8869 ± 2.2026	33.1271 ± 0.9757	85.6156 ± 13.8920	13.9287 ± 2.3296	27.4767 ± 0.6802	**71.3455** ± 4.2708	**12.6705** ± 0.9315	31.8922 ± 6.9502	83.1106 ± 13.5347	12.7511 ± 0.0954	**26.7614** ± 0.0760	72.1311 ± 13.2548
5.0	8	**11.5971** ± 0.0489	33.8632 ± 0.2012	88.5907 ± 13.6119	15.0350 ± 1.3511	29.4297 ± 3.5187	75.3748 ± 1.1483	14.4180 ± 3.1719	28.6653 ± 0.2714	85.0196 ± 1.8801	15.0784 ± 2.3797	**23.1434** ± 1.1509	**74.6340** ± 0.3419

As visualised in Figure 6.3, CenterFusion consistently exhibits the highest ATE at far range, indicating the poorest localisation performance in that distance interval. At moderate and strong enlargement levels, GaussianFusion++ outperforms all other models in the mid- and far-range categories. In contrast, for the baseline and mild enlargement settings, CenterFusion++ generally achieves the lowest ATE across almost all distance intervals.

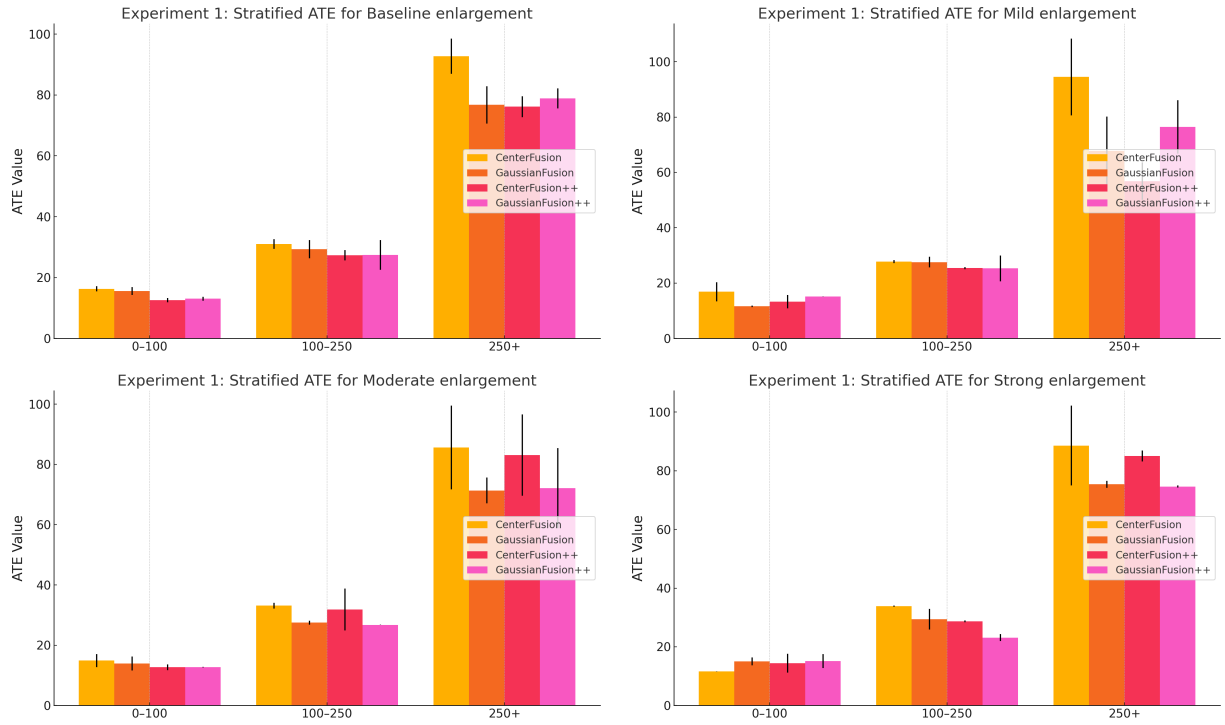


Figure 5.3: Stratified ATE results for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++ under different static frustum enlargement configurations (Baseline, Mild, Moderate, Strong). Each plot shows the ATE grouped by depth ranges: 0–100 m, 100–250 m, and beyond 250 m.

Table 5.2 presents the stratified ASE for all static frustum enlargement configurations. GaussianFusion++ consistently achieves the lowest ASE across all distance ranges and settings, with a single exception at mid-range under mild enlargement, where CenterFusion++ attains the best performance in scale estimation. In general, the ASE values for all models tend to decrease with increasing range, indicating better alignment between predicted and ground truth box shapes at farther distances. These stratified results reinforce the trend observed in the aggregated ASE histogram in Figure 6.2, where CenterFusion++ and GaussianFusion++ demonstrate overall superior performance in scale estimation. The stratified histograms in Figure 6.4 visualise these findings.

Table 5.2: Stratified ASE of all four models under different static frustum enlargement settings. Metrics are stratified by distance: near (< 100 m), mid (100–250 m), and far (> 250 m).

ϵ	δ	CenterFusion ASE ↓			GaussianFusion ASE ↓			CenterFusion++ ASE ↓			GaussianFusion++ ASE ↓		
		Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
0.0	0.0	0.4944 ± 0.0172	0.4700 ± 0.0016	0.3074 ± 0.0080	0.5086 ± 0.0061	0.4634 ± 0.0250	0.3163 ± 0.0139	0.4401 ± 0.0325	0.3862 ± 0.0148	0.2851 ± 0.0072	**0.4246** ± 0.0043	**0.3717** ± 0.0041	**0.2258** ± 0.0155
0.5	1.0	0.5293 ± 0.0363	0.4386 ± 0.0118	0.3352 ± 0.0045	0.4950 ± 0.0271	0.4350 ± 0.0013	0.2865 ± 0.0044	0.4425 ± 0.0259	**0.3828** ± 0.0371	0.2482 ± 0.0204	**0.4371** ± 0.0109	0.3875 ± 0.0202	**0.2410** ± 0.0282
2.0	3.0	0.4886 ± 0.0327	0.4718 ± 0.0441	0.3082 ± 0.0443	0.4954 ± 0.0158	0.4330 ± 0.0308	0.3142 ± 0.0437	0.4538 ± 0.0094	0.3838 ± 0.0261	0.2574 ± 0.0156	**0.4408** ± 0.0315	**0.3755** ± 0.0385	**0.2500** ± 0.0220
5.0	8.0	0.4489 ± 0.0222	0.4108 ± 0.0064	0.3040 ± 0.0064	0.4848 ± 0.0378	0.4295 ± 0.0038	0.2932 ± 0.0077	0.4442 ± 0.0032	0.3977 ± 0.0079	0.2484 ± 0.0203	**0.4359** ± 0.0100	**0.3660** ± 0.0015	**0.2376** ± 0.0008

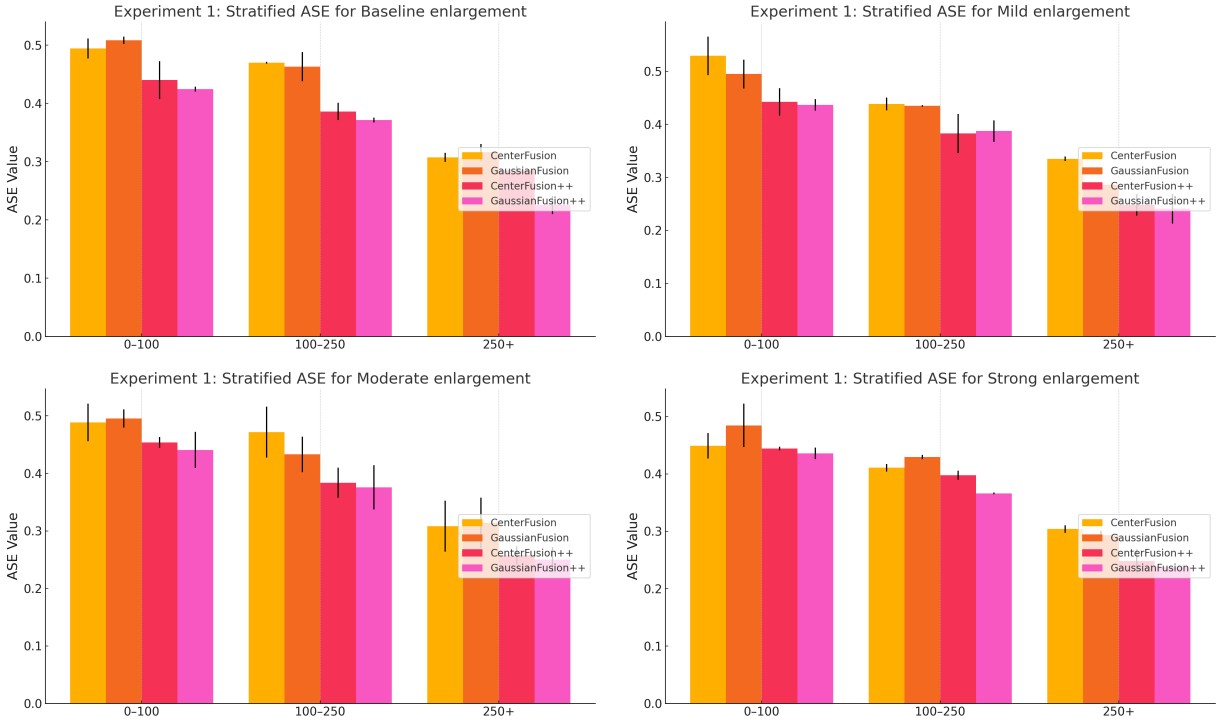


Figure 5.4: Stratified ASE results for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++ under different static frustum enlargement configurations (Baseline, Mild, Moderate, Strong). Each plot shows the ASE grouped by depth ranges: 0–100 m, 100–250 m, and beyond 250 m.

The stratified results for AOE and AVE follow the same trend observed in the aggregated analysis: no clear or interpretable patterns emerge as a consequence of using different frustum enlargement settings. The results appear highly variable and model-dependent, without consistent behaviour across distance ranges or configurations. As these findings offer limited insight into the effects of static frustum settings on orientation and velocity estimation, the corresponding figures are presented in Appendix A.2.1 for completeness.

5.1.3 Ablation study: The effect of horizontal frustum enlargement

To better understand the influence of horizontal frustum enlargement in isolation, an ablation study was conducted for GaussianFusion++. Specifically, two pairwise comparisons were performed: one to assess the effect of adding horizontal expansion alone, and another to investigate whether very strong horizontal expansion is necessary when combined with strong depth-wise enlargement.

Table 5.3 presents the aggregated performance metrics for these ablation settings. Comparing the baseline configuration ($\epsilon = 0, \delta = 0$) with moderate horizontal expansion only ($\epsilon = 2, \delta = 0$) shows that the addition of

horizontal enlargement results in a higher mAP (0.4411 vs. 0.4276) and a lower ATE (32.51 m vs. 35.75 m). Small improvements are also observed for ASE, AOE, and AVE, and the NDS increases from 0.5902 to 0.6074.

For the second pairwise comparison, the configuration with very strong horizontal and depth-wise expansion ($\epsilon = 5, \delta = 8$) is compared to the setting with moderate horizontal and strong depth-wise expansion ($\epsilon = 2, \delta = 8$). In this case, the moderate horizontal expansion achieves a slightly lower mAP (0.4068 vs. 0.4092) and a slightly lower ATE (31.90 m vs. 33.78 m), along with a higher NDS (0.5941 vs. 0.5874).

Table 5.3: Aggregated performance metrics for GaussianFusion++ comparing ablation settings with standard static frustum settings.

ϵ	δ	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
$\epsilon = 0$	$\delta = 0$	0.4276	35.7586	0.3527	0.7614	0.3372	0.5902
$\epsilon = 2$	$\delta = 0$	**0.4411**	**32.5102**	**0.3454**	**0.6371**	**0.2987**	**0.6074**
$\epsilon = 5$	$\delta = 8$	**0.4092**	33.5373	0.3590	0.6580	**0.3154**	0.5874
$\epsilon = 2$	$\delta = 8$	0.4068	**31.9007**	**0.3586**	**0.4976**	0.3573	**0.5941**

Table 5.4 shows the same results stratified by distance intervals. For the baseline versus moderate horizontal expansion ($\epsilon = 0$ vs. $\epsilon = 2$, both with $\delta = 0$), lower ATE values are observed across near and far distance ranges when moderate horizontal enlargement is applied. Minor improvements in ASE and AVE are also visible.

In the second comparison ($\epsilon = 5$ vs. $\epsilon = 2$ with $\delta = 8$), moderate horizontal expansion yields lower ATE across all depth ranges, as well as slightly lower ASE and AOE values.

Table 5.4: Stratified performance metrics for GaussianFusion++ comparing ablation settings with standard static frustum settings. Metrics are stratified by distance: near (< 100 m), mid (100–250 m), and far (> 250 m).

ϵ	δ	ATE ↓			ASE ↓			AOE ↓			AVE ↓		
		Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
0.0	0	13.0389	**27.4766**	78.8957	0.4246	0.3717	**0.2258**	0.7691	0.7632	0.7472	0.1698	0.3244	0.5914
2.0	0	**9.9349**	29.9278	**68.4272**	**0.4167**	**0.3516**	0.2347	**0.6007**	**0.5999**	**0.7385**	**0.1487**	**0.2900**	**0.5261**
5.0	8	15.0784	23.1434	74.6340	0.4359	0.3660	**0.2376**	0.5885	0.6877	0.7190	**0.1482**	**0.2942**	**0.5880**
2.0	8	**12.8828**	**22.2018**	**68.6582**	**0.4328**	**0.3582**	0.2639	**0.4381**	**0.5667**	**0.4853**	0.1796	0.3169	0.6364

Based on these results, it is evident that adding horizontal frustum enlargement does improve localisation accuracy, especially at farther ranges. However, excessive horizontal expansion introduces irrelevant radar points and increases noise, thereby diminishing overall performance. Consequently, horizontal enlargement can be beneficial within certain threshold boundaries.

5.2 Experiment 2: Dynamic frustum enlargement

The goal of this experiment is to evaluate the effect of a dynamic frustum enlargement module on the performance of the four radar–camera fusion models. This module implements a depth-adaptive frustum expansion based on the quadratic function defined in Equation 3.8, allowing the frustum size to increase with the predicted object depth.

The motivation for introducing a dynamic frustum enlargement module stems from a key limitation observed with fixed enlargement settings: when a model is trained with a fixed frustum configuration, it optimises its parameters specifically for that setting. This behaviour can lead to suboptimal performance because the optimal frustum size likely depends on the object's distance. Due to the nature of monocular depth estimation, prediction errors increase superlinearly with distance [65]; hence, using a fixed enlargement does not provide adequate tolerance for distant objects and may include unnecessary noise for near objects. To overcome this, a dynamic module aims to adapt the frustum size automatically, allocating smaller frustums for closer, more accurately predicted objects, and larger frustums for distant objects where greater uncertainty must be accommodated.

To investigate this effect, all four models are trained using the same dynamic frustum module with the following parameters: $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. These values are selected based on the quantitative and qualitative results from Experiment 1, ensuring a stable yet flexible enlargement that grows quadratically with depth. Additionally, informed by the ablation study findings, the horizontal frustum enlargement parameter ϵ is explicitly bounded to prevent excessive enlargement at greater depths. Specifically, ϵ is computed as

$\epsilon = \min\left(\epsilon_{base} + k \cdot \text{depth}^2,\ 2\right)$, which ensures that horizontal enlargement remains within the empirically determined optimal range while still being depth-dependent. Visualisations of the resulting depth-wise and horizontal enlargement functions are provided in Figure 5.5.

As an additional verification step, two ablation studies are conducted. The first focuses on the GaussianFusion model and includes two targeted configurations. First, the horizontal enlargement bound is removed to re-validate the ablation findings from Experiment 1 and to assess whether restricting horizontal growth remains necessary under dynamic enlargement. Second, the horizontal enlargement is again bounded at a maximum of 2, but the quadratic scaling factor is increased to $k = 7.5 \times 10^{-6}$ to enable faster frustum growth with depth. This ablation isolates the sensitivity of GaussianFusion to both the shape constraints and the growth rate of the frustum under dynamic expansion.

The second ablation study examines whether different radar–camera fusion models benefit from different dynamic enlargement rates. Specifically, CenterFusion++ and GaussianFusion++ are retrained using an increased quadratic scaling factor of $k = 1 \times 10^{-5}$, while all other enlargement parameters remain fixed. These two models are selected because, unlike CenterFusion and GaussianFusion, they incorporate neural network modules for frustum-level radar association. By comparing their performance under this more aggressive frustum growth, this ablation aims to determine whether models that rely on learnable radar association strategies exhibit different optimal dynamic enlargement parameters.

This experiment is designed to test the hypothesis that a depth-adaptive frustum enlargement improves the robustness and overall localisation performance of all models across various depth ranges, compared to static frustum configurations.

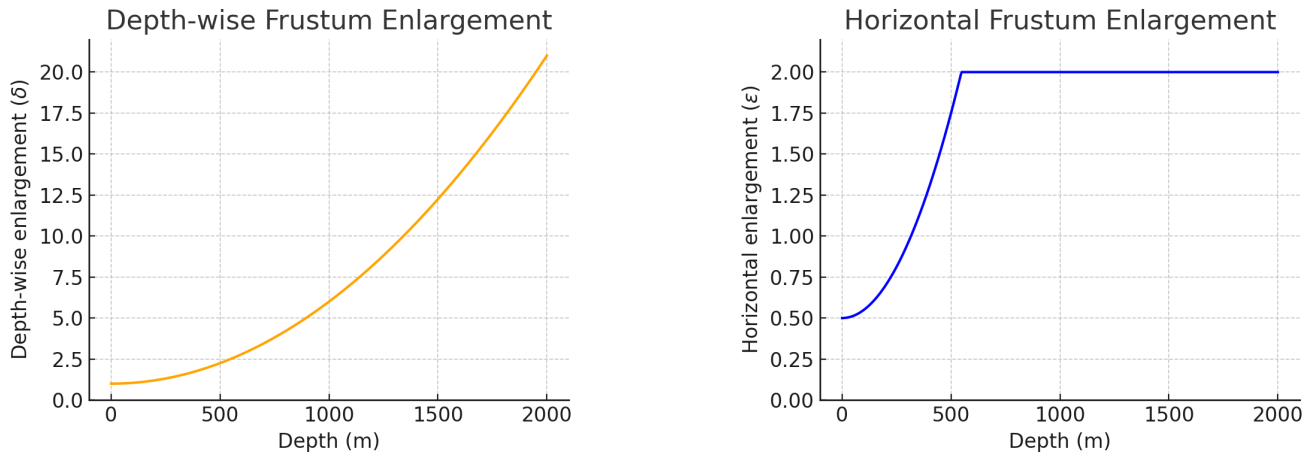


Figure 5.5: Dynamic frustum enlargement functions for depth-wise (left) and horizontal (right) directions, illustrating the depth-adaptive behaviour used in the proposed module. Depth-wise enlargement δ starts at 1. Horizontal enlargement ϵ starts at 0.5 and is capped at 2.

5.2.1 Aggregated results

Table 5.5 presents the aggregated performance metrics for all four models using the dynamic frustum enlargement configuration with $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. Under this setting, GaussianFusion++ achieves the highest mAP (0.4204) and the lowest ASE (0.3643), while CenterFusion++ obtains the lowest ATE (27.2527) and the lowest AVE (0.3462). CenterFusion records the lowest AOE (0.5597). Representative qualitative results for each model under this configuration are provided in Figures A.2.12 and A.2.13 in Appendix A.2.2. Aggregated histograms comparing the dynamic enlargement configuration are shown in Figure A.2.14 in Appendix A.2.2.

Table 5.5: Aggregated performance metrics for all four models with the dynamic frustum enlargement module, shown in Equation 3.8, using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$.

Model	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
CenterFusion	0.4080	35.5027	0.4313	**0.5597**	0.3829	0.5784
GaussianFusion	0.3923	34.8743	0.4364	0.5711	0.3533	0.5706
CenterFusion++	0.4009	**27.2527**	0.3735	0.8286	**0.3462**	0.5821
GaussianFusion++	**0.4204**	32.1465	**0.3643**	0.6017	0.3940	**0.5953**

5.2.2 Stratified results

Table 5.6 reports the stratified metrics grouped by distance intervals. GaussianFusion++ achieves the lowest mid and far-range ASE (0.3704 and 0.2327, respectively) and the lowest near-range AVE (0.1634). CenterFusion++ achieves the lowest near-range ATE (10.0967) and far-range ATE (50.4306), while CenterFusion achieves the lowest near and far AOE. These results confirm that the depth-adaptive module improves detection consistency across distance bands, addressing the increased uncertainty at longer ranges. Stratified histograms illustrating these trends are presented in Figure A.2.15 in Appendix A.2.2.

Table 5.6: Stratified performance metrics for all four models with the dynamic frustum enlargement module, shown in Equation 3.8, using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. Metrics are stratified by distance: near ($< 100\,\text{m}$), mid (100–250 m), and far ($> 250\,\text{m}$).

Model	ATE ↓			ASE ↓			AOE ↓			AVE ↓		
	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
CenterFusion	14.2234	27.1027	75.7867	0.4842	0.4448	0.3409	**0.5376**	0.5746	**0.5706**	0.2155	**0.3045**	0.7161
GaussianFusion	14.0913	27.8162	70.8275	0.5007	0.4554	0.3285	0.6064	**0.5017**	0.6102	0.1857	0.3310	0.6007
CenterFusion++	**10.0967**	28.9267	**50.4306**	**0.4359**	0.3788	0.2741	0.7177	0.9262	0.8653	0.1642	0.3888	**0.5597**
GaussianFusion++	13.1166	**25.6693**	69.8268	0.4458	**0.3704**	**0.2327**	0.5624	0.6145	0.6434	**0.1634**	0.3680	0.7782

5.2.3 Ablation study: Varying dynamic enlargement settings

The first ablation study is conducted on the GaussianFusion model and investigates the impact of both bounding the horizontal enlargement and adjusting the frustum growth rate. Table 5.7 presents the aggregated performance metrics for GaussianFusion under different dynamic frustum configurations. Removing the horizontal enlargement bound while keeping $k = 5 \times 10^{-6}$ results in an increase in mAP, a slight gain in NDS, and a small decrease in ASE. However, it does lead to a significantly higher ATE (39.7939 vs. 34.8743), a small increase in AOE, and a larger AVE. Increasing k to 7.5×10^{-6} while retaining the horizontal bound slightly lowers ASE and AOE but negatively impact all other metrics.

Table 5.7: Aggregated performance metrics for GaussianFusion with the dynamic frustum enlargement module, shown in Equation 3.8, using different configurations.

$k \times 10^{-6}$	ϵ bounded	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
5	Yes	0.3923	**34.8743**	0.4364	0.5711	**0.3533**	0.5706
5	No	**0.4084**	39.7939	0.4268	0.5782	0.3784	**0.5731**
7.5	Yes	0.3798	35.2186	**0.4057**	**0.5059**	0.3839	0.5699

Table 5.8 provides the same ablation results stratified by distance intervals. The configuration with an unbounded horizontal enlargement parameter exhibits lower near-range ATE but substantially higher far-range ATE (87.8874) compared to the bounded setting (70.8275), confirming that unrestrained widening particularly affects localisation at longer ranges where depth errors compound. For near-range localisation, the bounded and unbounded configurations produce similar enlargement sizes, since the predicted depths are low and the quadratic term remains below the maximum cap. The small observed difference in near-range ATE is therefore attributed to minor training variation rather than an actual effect of the bound.

For the configuration with increased k to 7.5×10^{-6}, the stratified results show slightly lower ATE and ASE at far range compared to the standard setting, indicating that a larger frustum can be beneficial for distant objects where higher uncertainty must be tolerated. However, performance at near and mid ranges shows minor degradation, and the differences across all intervals remain small. This suggests that increasing the quadratic scaling factor may provide marginal gains in localisation at greater depths but does not consistently improve localisation overall.

Table 5.8: Stratified performance metrics for GaussianFusion with the dynamic frustum enlargement module, shown in Equation 3.8, using different configurations. Metrics are stratified by distance: near ($< 100\,\text{m}$), mid (100–250 m), and far ($> 250\,\text{m}$).

k	ϵ	ATE ↓			ASE ↓			AOE ↓			AVE ↓		
$\times 10^{-6}$	bounded	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
5	Yes	14.0913	**27.8162**	70.8275	0.5007	0.4554	0.3285	0.6064	0.5017	0.6102	0.1857	0.3310	**0.6007**
5	No	**12.3327**	29.8492	87.8874	**0.4600**	0.4633	0.3374	0.5948	**0.4927**	0.6669	0.1786	0.3255	0.7031
7.5	Yes	16.5912	31.6379	**67.1150**	0.4892	**0.4134**	**0.2750**	**0.4658**	0.5416	**0.5103**	**0.1732**	**0.3030**	0.8043

In summary, the results of the first ablation reinforce the conclusion from Experiment 1 that bounding the horizontal enlargement parameter is essential for maintaining robust localisation at greater depths, as unbounded widening introduces excessive noise that degrades far-range accuracy. The second ablation indicates that using a slightly larger quadratic scaling factor can provide marginal improvements in localisation performance for distant objects by allowing more aggressive frustum growth. However, this benefit comes at the cost of slightly reduced performance at near and mid ranges. These findings highlight the importance of carefully balancing frustum growth to accommodate increasing uncertainty with depth while minimising the inclusion of irrelevant radar points.

The second ablation study investigates whether CenterFusion++ and GaussianFusion++ exhibit different optimal frustum scaling factors by increasing the dynamic enlargement rate to $k = 1 \times 10^{-5}$ while keeping all other parameters fixed. Table 5.9 presents the aggregated performance metrics for both models under this configuration. Compared to the baseline setting with $k = 5 \times 10^{-6}$, GaussianFusion++ improves in mAP (from 0.4204 to 0.4273), ATE (from 32.1465 to 29.6389), and ASE (from 0.3643 to 0.3494), indicating that it benefits from a more aggressive frustum expansion. CenterFusion++, on the other hand, records slightly worse ATE (rising from 27.2527 to 29.9651) and ASE (from 0.3735 to 0.3485), though it shows modest gains in mAP (from 0.4009 to 0.4307), AOE (from 0.8286 to 0.6789), and NDS (from 0.5821 to 0.6029). The limited degradation in localisation performance for CenterFusion++ may be attributed to the fact that the majority of training samples lie in depth regions where the effective enlargement remains close to the base parameters of $\epsilon = 0.5$ and $\delta = 1$.

Table 5.9: Aggregated performance metrics for CenterFusion++ and GaussianFusion++ using a higher frustum scaling factor of $k = 1 \times 10^{-5}$.

Model ($k = 1 \times 10^{-5}$)	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
CenterFusion++	**0.4307**	29.9651	**0.3485**	**0.6789**	**0.3318**	**0.6029**
GaussianFusion++	0.4273	**29.6389**	0.3494	0.7343	0.3347	0.5993

Stratified metrics in Table 5.10 further support these trends. GaussianFusion++ achieves lower ATE at all ranges, with improvements at near (from 13.1166 to 11.3251,m), mid (from 25.6693 to 19.7690,m), and far (from 69.8268 to 63.6721,m). It also records lower ASE at near (from 0.4458 to 0.4067), mid (from 0.3704 to 0.3891), and far (from 0.2327 to 0.2324), and improved AVE at near (from 0.1634 to 0.1573), mid (from 0.3680 to 0.3306), and far (from 0.7782 to 0.5538). AOE also slightly improves at near (from 0.5624 to 0.6132) and mid (from 0.6145 to 0.7676), but worsens at far (from 0.6434 to 0.8402). In contrast, CenterFusion++ shows a mixed response. ATE increases at near (from 10.0967 to 13.4347,m) and far (from 50.4306 to 53.6903,m), while slightly improving at mid range (from 28.9267 to 30.6326,m). ASE increases at near (from 0.4359 to 0.4407) and decreases at mid (from 0.3788 to 0.3552) and far (from 0.2741 to 0.2024). A similar pattern is observed for AVE, which improves at mid (from 0.3888 to 0.3091) but worsens at near (from 0.1642 to 0.1845) and far (from 0.5597 to 0.5818). These results indicate that GaussianFusion++ consistently benefits from the increased scaling factor, while CenterFusion++ shows a less uniform response with minor performance degradation at near and far distances.

Table 5.10: Stratified performance metrics for CenterFusion++ and GaussianFusion++ using $k = 1 \times 10^{-5}$. Metrics are stratified by distance: near ($< 100\,\text{m}$), mid (100–250 m), and far ($> 250\,\text{m}$).

Model	ATE ↓			ASE ↓			AOE ↓			AVE ↓		
$k = 1 \times 10^{-5}$	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
CenterFusion++	13.4347	30.6326	**53.6903**	0.4407	**0.3552**	**0.2024**	0.6170	0.7956	**0.6147**	0.1845	**0.3091**	0.5818
GaussianFusion++	**11.3251**	**19.7690**	63.6721	**0.4067**	0.3891	0.2324	**0.6132**	**0.7676**	0.8402	**0.1573**	0.3306	**0.5538**

In summary, this ablation study demonstrates that radar–camera fusion models with similar architectural en-

hancements can respond differently to the same frustum scaling strategy. GaussianFusion++ shows robust gains from a higher dynamic enlargement rate, leveraging the increased radar context to improve localisation and detection across depth bands. In contrast, CenterFusion++ exhibits more constrained improvements and a degree of deterioration in localisation accuracy, highlighting the importance of tuning dynamic enlargement parameters to the characteristics of each model's radar association mechanism.

5.3 Experiment 3: Uncertainty scaling

This experiment investigates how varying the scale factor α affects the performance of GaussianFusion when using dynamic frustum enlargement. As described in Chapter 3, the scale factor α determines the variances of the bivariate Gaussian distribution used to weight radar points within each frustum. Specifically, the variances are computed as $\sigma_x^2 = \alpha \cdot (x_{\max} - x_{\min})^2$ and $\sigma_z^2 = \alpha \cdot (z_{\max} - z_{\min})^2$. The values of x and z depend on the frustum dimensions, which themselves increase with predicted object depth due to the dynamic enlargement strategy. Consequently, the absolute uncertainty also grows with depth.

The motivation for this experiment stems from the fact that the influence of the scale factor α on the detection performance remains unknown and must be systematically investigated. Intuitively, a smaller α tightens the weighting around the predicted object center, potentially filtering out noise but risking the exclusion of relevant radar points when monocular predictions are inaccurate. Conversely, a larger α allows a broader inclusion of radar points, which may enhance robustness at greater depths but could introduce additional noise at close range.

To explore this, GaussianFusion is trained with three different scale factor settings while using the same dynamic frustum enlargement parameters as in Experiment 5.2 ($\epsilon_{base} = 0.5$, $\delta_{base} = 1$, $k = 5 \times 10^{-6}$):

- **Baseline:** $\alpha = 0.1$;
- **Low uncertainty:** $\alpha = 0.05$;
- **High uncertainty:** $\alpha = 0.2$.

This experiment tests the hypothesis that a higher α may positively influence performance at greater depths, where the frustum and thus the inherent uncertainty are already larger, while a lower α may be beneficial for improving performance at closer ranges by tightly focusing the radar point weighting.

5.3.1 Aggregated results

Table 5.11 shows the aggregated performance metrics for GaussianFusion with three different scale factor settings α, using the dynamic frustum enlargement configuration. The baseline configuration with $\alpha = 0.1$ achieves a mAP of 0.3923 and an NDS of 0.5706. The low uncertainty setting ($\alpha = 0.05$) results in a slightly lower mAP of 0.3829 and an NDS of 0.5607. The high uncertainty setting ($\alpha = 0.2$) achieves the highest mAP of 0.4173 and an NDS of 0.5745.

The ATE increases from 34.8743 at baseline to 36.2807 for the low uncertainty setting and 40.6204 for the high uncertainty setting. The ASE decreases slightly with increasing α, from 0.4364 at baseline to 0.4162 at high uncertainty. The AOE is lowest for the baseline (0.5711), increasing to 0.7177 for low uncertainty and 0.6626 for high uncertainty. The AVE is lowest for the low uncertainty setting (0.3261), compared to 0.3533 at baseline and 0.3720 for the high uncertainty setting. Representative qualitative results for each model under this configuration are presented in Figures A.2.16 and A.2.17 in Appendix A.2.3. Aggregated histograms are shown in Figure A.2.18 in Appendix A.2.3.

Table 5.11: Aggregated performance metrics of GaussianFusion using different scale factor settings α and the dynamic frustum enlargement module (Equation 3.8), with $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$.

α	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
0.1 (Baseline)	0.3923	**34.8743**	0.4364	**0.5711**	0.3533	0.5706
0.05 (Low)	0.3829	36.2807	0.4206	0.7177	**0.3261**	0.5607
0.2 (High)	**0.4173**	40.6204	**0.4162**	0.6626	0.3720	**0.5745**

5.3.2 Stratified results

Table 5.12 presents the stratified metrics by distance for each α setting. For ATE, the low uncertainty configuration ($\alpha = 0.05$) shows the lowest values for near and mid ranges (11.9129 and 26.3541), while the baseline setting shows the lowest far-range ATE (70.8275). The high uncertainty setting shows higher ATE across all ranges, particularly at far range (95.4180).

For ASE, the low uncertainty setting achieves slightly lower values for near and far ranges compared to the other settings. The high uncertainty setting has the lowest mid-range ASE (0.4008). Regarding AOE, the baseline shows the lowest near and mid-range values (0.6064 and 0.5017), while the low uncertainty setting shows the lowest far-range AOE (0.7290). The high uncertainty setting yields higher AOE values across all ranges. For AVE, the high uncertainty setting shows the lowest values for near and mid ranges (0.1414 and 0.2888), while the low uncertainty setting achieves the lowest far-range AVE (0.5686). To complement the numerical results in Table 5.12, stratified histograms are presented in Figure A.2.19 in Appendix A.2.3.

Table 5.12: Stratified performance metrics of GaussianFusion using different scale factor settings α and the dynamic frustum enlargement module (Equation 3.8), with $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. Metrics are stratified by distance: near (< 100 m), mid (100–250 m), and far (> 250 m).

α	ATE ↓			ASE ↓			AOE ↓			AVE ↓		
	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
0.1 (Baseline)	14.0913	27.8162	**70.8275**	0.5007	0.4554	0.3285	**0.6064**	**0.5017**	**0.6102**	0.1857	0.3310	0.6007
0.05 (Low)	**11.9129**	**26.3541**	83.465	**0.4950**	0.4394	**0.2926**	0.7044	0.7225	0.7290	0.1765	0.2990	**0.5686**
0.2 (High)	15.0089	27.5348	95.4180	0.5043	**0.4008**	0.3082	0.6727	0.5523	0.7950	**0.1414**	**0.2888**	0.8193

5.4 Experiment 4: RGPNet-A vs. RGPNet-B

This experiment investigates the impact of token-mixing within RGPNet for GaussianFusion++. As detailed in Section 3.2.2, two variants of RGPNet are considered: RGPNet-A, which relies solely on independent point-wise transformations (channel-mixing only), and RGPNet-B, which augments this architecture with token-mixing layers to explicitly model structured interactions between radar points within each frustum.

The motivation for this experiment stems from the insight that radar returns within a frustum often exhibit local spatial dependencies and geometric patterns, particularly in cluttered maritime environments where reflections can cluster along object edges or water surfaces. While RGPNet-A treats each radar point independently before global pooling, it cannot capture such inter-point relationships. By contrast, RGPNet-B introduces token-mixing, allowing the network to learn spatial interactions across the radar point set. This additional capacity could help the network generate more reliable and context-aware Gaussian parameter estimates, especially in complex or noisy scenes.

In this experiment, GaussianFusion++ is trained twice under identical conditions, varying only the RGPNet variant used for radar parameter estimation. Both configurations employ the same dynamic frustum enlargement module and training strategy described in previous experiments, ensuring a controlled comparison. The evaluation focuses on whether token-mixing in RGPNet-B improves overall detection accuracy, localisation, and robustness compared to the simpler RGPNet-A design.

Additionally, an ablation study is conducted to investigate two specific factors. First, the effect of varying the maximum number of tokens N used in RGPNet-B is examined. While Chapter 3.2.2 motivates the choice of $N = 128$ as a trade-off between geometric coverage and computational efficiency, this ablation empirically evaluates whether alternative values lead to better performance. Second, RGPNet is replaced with the more complex and widely-used PointNet++ architecture [58] to determine whether general-purpose point cloud networks offer an advantage over the specialised, lightweight design of RGPNet. In both cases, GaussianFusion++ is retrained for 40 epochs under otherwise identical conditions to ensure comparability.

Finally, to assess the practical detection performance of the best-performing configuration identified in this experiment, a threshold sweep analysis is conducted. This analysis evaluates GaussianFusion++ the best learnable frustum association mechanism across a range of confidence thresholds to determine the setting that maximises the F1 score. By examining the precision–recall trade-off, this evaluation provides a complementary perspective on detection quality beyond standard metrics such as mAP and ATE, and supports a more deployment-oriented assessment of the model.

This experiment tests the hypothesis that explicitly modelling point-to-point dependencies enhances the quality of the estimated radar feature representations, resulting in more accurate radar–camera fusion and improved

localisation performance.

5.4.1 Aggregated results

Table 5.13 shows the aggregated performance metrics for GaussianFusion++ with the two RGPNet variants using the dynamic frustum enlargement configuration. RGPNet-B achieves a higher mAP of 0.4563 compared to 0.4204 for RGPNet-A. The NDS is also slightly higher for RGPNet-B (0.5993) than for RGPNet-A (0.5953).

The ATE is lower for RGPNet-A (32.1465) compared to RGPNet-B (32.8684). The ASE is slightly lower for RGPNet-B (0.3573) than for RGPNet-A (0.3643). The AOE increases from 0.6017 for RGPNet-A to 0.9846 for RGPNet-B. The AVE decreases from 0.3940 for RGPNet-A to 0.3008 for RGPNet-B. Representative qualitative results for each model under this configuration are provided in Figures A.2.20 and A.2.21 in Appendix A.2.4. Aggregated histograms are shown in Figure A.2.22 in Appendix A.2.4.

Table 5.13: Aggregated performance metrics of GaussianFusion++ using RGPNet-A and RGPNet-B with the dynamic frustum enlargement module (Equation 3.8), using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$.

Encoder ($k = 5 \times 10^{-6}$)	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
RGPNet-A	0.4204	**32.1465**	0.3643	**0.6017**	0.3940	0.5953
RGPNet-B	**0.4563**	32.8684	**0.3573**	0.9846	**0.3008**	**0.5993**

5.4.2 Stratified results

Table 5.14 presents the stratified performance metrics for near, mid, and far ranges. For ATE, RGPNet-B achieves lower values in the near and far ranges (10.3864 and 64.7392) compared to RGPNet-A (13.1166 and 69.8268), while the mid-range ATE is higher for RGPNet-B (33.4393) than for RGPNet-A (25.6693).

The ASE is slightly lower for RGPNet-B across all ranges. The AOE is higher for RGPNet-B in all distance intervals. The AVE is lower for RGPNet-B in the near, mid, and far ranges compared to RGPNet-A. The results presented in Table 5.14 are visualised as stratified histograms in Figure A.2.23 in Appendix A.2.4.

Table 5.14: Stratified performance metrics of GaussianFusion++ using RGPNet-A and RGPNet-B with the dynamic frustum enlargement module (Equation 3.8), using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. Metrics are stratified by distance: near (< 100 m), mid (100–250 m), and far (> 250 m).

Encoder	ATE ↓			ASE ↓			AOE ↓			AVE ↓		
$k = 5 \times 10^{-6}$	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
RGPNet-A	13.1166	**25.6693**	69.8268	0.4458	0.3704	0.2327	**0.5624**	**0.6145**	**0.6434**	0.1634	0.3680	0.7782
RGPNet-B	**10.3864**	33.4393	**64.7392**	**0.4426**	**0.3658**	**0.2221**	0.8537	1.1279	0.9861	**0.1520**	**0.2636**	**0.5656**

5.4.3 Ablation study: RGPNet-B with varying token count N vs. PointNet++

This ablation study investigates the influence of varying the fixed radar token count N within the RGPNet-B encoder on the overall performance of GaussianFusion++. The design of RGPNet-B assumes that $N = 128$ strikes a balance between preserving spatial detail and computational efficiency, as described in Section 3.2.2. To validate this choice, RGPNet-B is also evaluated with $N = 64$ and $N = 256$. Additionally, PointNet++ [58] is included as a comparative baseline to assess the performance of a widely used, general-purpose radar encoder within the same learnable frustum association framework.

PointNet++ is a hierarchical point-based network designed to model local structures and inter-point relationships using multi-scale grouping, a technique used to extract meaningful geometric patterns from a point cloud. It has been applied in various 3D vision tasks, including radar and LiDAR point cloud encoding. Its inclusion in this study serves to assess whether a general-purpose point cloud network can outperform the proposed RGPNet-B encoder in the context of estimating Gaussian parameters for GaussianFusion++.

As shown in Table 5.15, GaussianFusion++ with RGPNet-B and $N = 128$ achieves the highest mAP (0.4563) and NDS (0.5993), confirming the effectiveness of this configuration. In comparison, the same architecture with $N = 64$ or $N = 256$ results in lower mAP and NDS, suggesting that both under- and over-sampling of radar points reduce performance. Notably, using $N = 64$ results in the lowest ATE (38.67) among the RGPNet-B variants, indicating that fewer radar points may reduce noise at the cost of detection performance. On the other hand, using PointNet++ leads to slightly better ATE (31.96) than any RGPNet-B variant. This suggests that

PointNet++ enables slightly more precise localisation on matched predictions, but its mAP is significantly lower (0.3854), indicating a reduced ability to detect objects in the first place.

While PointNet++ achieves lower ATE, it detects fewer objects overall, which is reflected in its low mAP (0.3854). Since ATE is computed only on TPs, this implies that the precision of PointNet++-enabled detections may be higher, but the overall recall is lower. In contrast, RGPNet-B with $N = 128$ achieves significantly higher mAP, indicating that it enables more consistent detection across all objects. As object detection is the primary task of GaussianFusion++, the improvement in mAP is more meaningful than the small gains in ATE offered by PointNet++.

Table 5.15: Aggregated performance metrics of GaussianFusion++ using different radar encoders with the dynamic frustum enlargement module (Equation 3.8), using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$.

Encoder ($k = 5 \times 10^{-6}$)	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
RGPNet-B ($N = 128$)	**0.4563**	32.8684	**0.3573**	0.9846	**0.3008**	**0.5993**
RGPNet-B ($N = 64$)	0.4238	38.6683	0.3765	**0.6858**	0.3080	0.5851
RGPNet-B ($N = 256$)	0.4190	36.2651	0.3766	**0.8449**	0.3437	0.5789
PointNet++	0.3854	**31.9574**	0.3604	**0.9095**	0.3589	0.5667

Table 5.16 provides additional insights by stratifying performance across distance intervals. RGPNet-B with $N = 128$ performs best at near and far distances, with the lowest ATE in both ranges (10.39 and 64.74 respectively), suggesting that token-mixing at this resolution enables the model to capture both local radar structure and long-range spatial patterns effectively. However, its ATE at mid-range (33.44) is substantially higher than all other configurations, which negatively impacts its aggregated ATE. This explains why the overall ATE of RGPNet-B ($N = 128$) is slightly worse than PointNet++ despite superior near and far performance.

Table 5.16: Stratified performance metrics of GaussianFusion++ using different radar encoders with the dynamic frustum enlargement module (Equation 3.8), using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. Metrics are stratified by distance: near (< 100 m), mid (100–250 m), and far (> 250 m).

Encoder	ATE ↓			ASE ↓			AOE ↓			AVE ↓		
$k = 5 \times 10^{-6}$	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
RGPNet-B ($N = 128$)	**10.3864**	33.4393	**64.7392**	0.4426	**0.3658**	**0.2221**	0.8537	1.1279	0.9861	**0.1520**	**0.2636**	0.5656
RGPNet-B ($N = 64$)	14.5295	**30.0078**	82.2455	0.4549	0.3909	0.2520	**0.6034**	**0.8318**	**0.6161**	0.1588	0.2875	**0.5361**
RGPNet-B ($N = 256$)	11.4883	**27.4877**	81.2495	0.4713	0.3859	0.2355	0.9849	**0.8055**	**0.7033**	0.1770	0.2882	0.6418
PointNet++	10.4797	**25.3838**	65.3347	**0.4275**	0.3787	0.2587	0.8665	**0.9174**	**0.9525**	0.1607	0.3418	0.6174

These findings demonstrate that while RGPNet-B with $N = 128$ exhibits slightly higher mid-range localisation error, it outperforms all other configurations in near- and far-range localisation and achieves the highest overall mAP and NDS. As mAP reflects the ability to detect objects reliably, regardless of how precisely they are localised, the performance gains in detection outweigh the minor localisation trade-off. Therefore, RGPNet-B with $N = 128$ is concluded to be the most effective radar encoder for the learnable frustum association module in GaussianFusion++.

5.4.4 Threshold selection

This subsection presents a threshold analysis for the best-performing configuration of GaussianFusion++, which uses the RGPNet-B encoder with $N = 128$. The objective is to identify the confidence threshold that yields the optimal balance between precision and recall, as measured by the F1 score. The analysis is conducted using a fixed IoU threshold of 0.5 for determining true positives.

To this end, the detection confidence threshold is swept from 0.01 to 0.99 in steps of 0.01. At each step, the corresponding precision, recall, and F1 score are computed on the validation set. These results are visualised using two complementary plots:

- A precision–recall curve, shown in Figure 5.6, which characterises the trade-off between detection accuracy and completeness across thresholds.
- An F1 score vs. confidence threshold curve, shown in Figure 5.7, which identifies the threshold that maximises the F1 score.

The analysis reveals that the best F1 score of 0.8284 is achieved at a confidence threshold of 0.39. This threshold represents the optimal operating point for deployment when precision and recall are equally important. It is marked in red on both figures for reference.

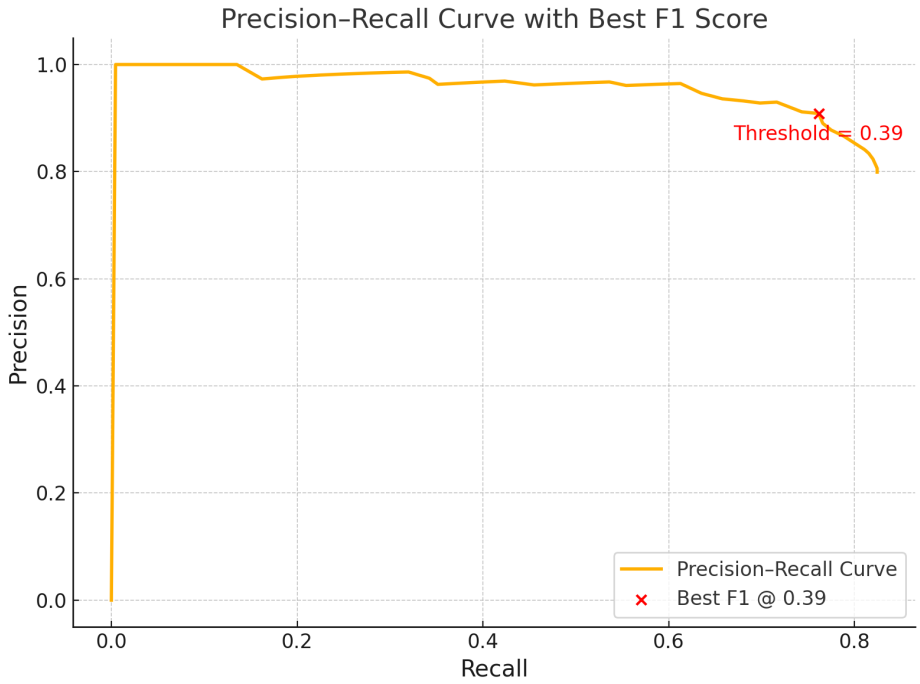


Figure 5.6: Precision–recall curve for GaussianFusion++ (RGPNet-B, $N = 128$). The red marker indicates the confidence threshold (0.39) that achieves the best F1 score.

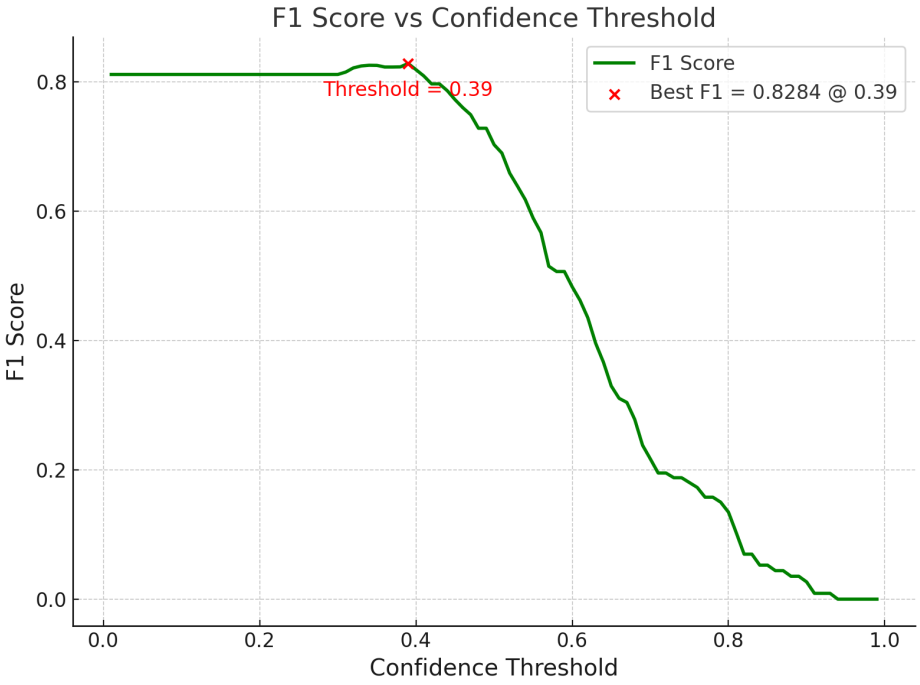


Figure 5.7: F1 score as a function of detection confidence threshold. The red marker indicates the threshold (0.39) that maximises the F1 score.

Chapter 6
Discussion

This chapter provides a critical analysis of the experimental results in relation to the main objective of this thesis: designing a novel multi-modal 3D object detection model using radar and camera data to complement RH Marine's CAS. The discussion examines how specific architectural modifications and parameter adjustments influence model behaviour and task-specific performance under realistic maritime conditions. The evaluation metrics are both aggregated and stratified. This stratification splits the TP metrics (ATE, ASE, AOE, and AVE) into near- (0-100 m), mid- (100-250 m), and far-range (250+ m) depth intervals. The discussion further considers how the challenges identified in Section 1.1 can be addressed through the proposed methods and experimental insights. By highlighting strengths, limitations, and practical trade-offs, this chapter clarifies the contribution of the developed approaches to improving situational awareness for autonomous surface vessels and lays the groundwork for the concluding reflections and recommendations for future research.

6.1 Impact of frustum enlargement strategies

Experiment 1 examined how different configurations of frustum enlargement, defined by the horizontal (ϵ) and depth-wise (δ) expansion parameters, affect the performance of the four radar-camera fusion models. The aim was to determine how varying the frustum volume influences the inclusion of relevant radar points and the resulting detection accuracy in complex maritime scenarios. Performance was tracked using the task-specific metrics mAP, ATE, ASE, AOE, and AVE, which are combined to generate NDS. It is anticipated that GaussianFusion and GaussianFusion++, which rely on probabilistic radar point weighting, would handle larger frustums more robustly than CenterFusion [37] and CenterFusion++ [38], due to their capacity to mitigate the impact of irrelevant or noisy radar returns. This section analyses how these expectations align with the experimental outcomes and highlights key trends observed across different enlargement settings.

The aggregated histograms in Figure 6.1 show that mAP remains stable across different frustum enlargement levels for CenterFusion and GaussianFusion. CenterFusion++ exhibits a reduction at moderate enlargement, while GaussianFusion++ exhibits a reduction at strong enlargement. The NDS indicates that CenterFusion's overall performance is largely unaffected by frustum size, while the other models achieve their highest NDS at mild enlargement ($\epsilon = 0.5$, $\delta = 1$).

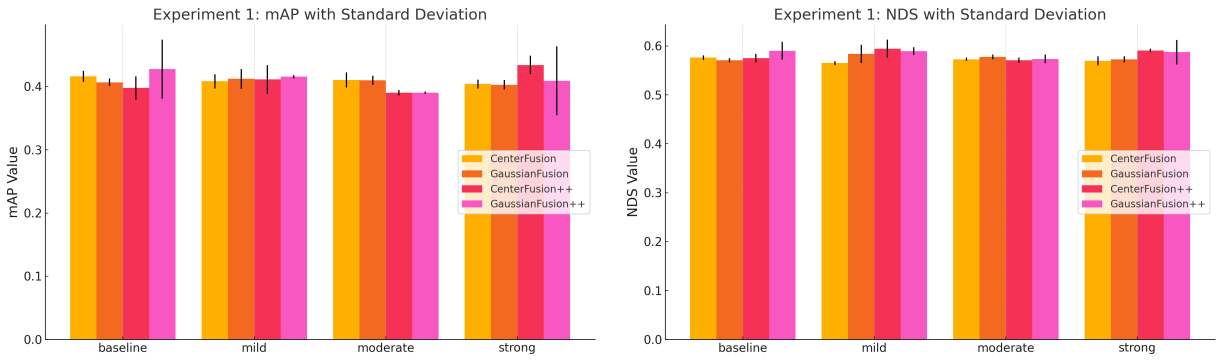


Figure 6.1: Aggregated detection performance metrics (mAP, NDS) for all models under different static frustum enlargement configurations (Baseline, Mild, Moderate, Strong). Each subplot visualises the trend for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++.

The ATE results, visualised in Figure 6.2, demonstrate that the level of frustum enlargement directly affects depth estimation accuracy, which aligns with the fact that radar depth fusion primarily influences translation error. CenterFusion and GaussianFusion++ reach the lowest ATE at moderate enlargement, whereas GaussianFusion and CenterFusion++ perform best at mild enlargement. For this setting, the parameters are $\epsilon = 0.5$ and $\delta = 1$.

This suggests that probabilistic radar weighting enables more robust depth estimation when the frustum includes additional radar returns.

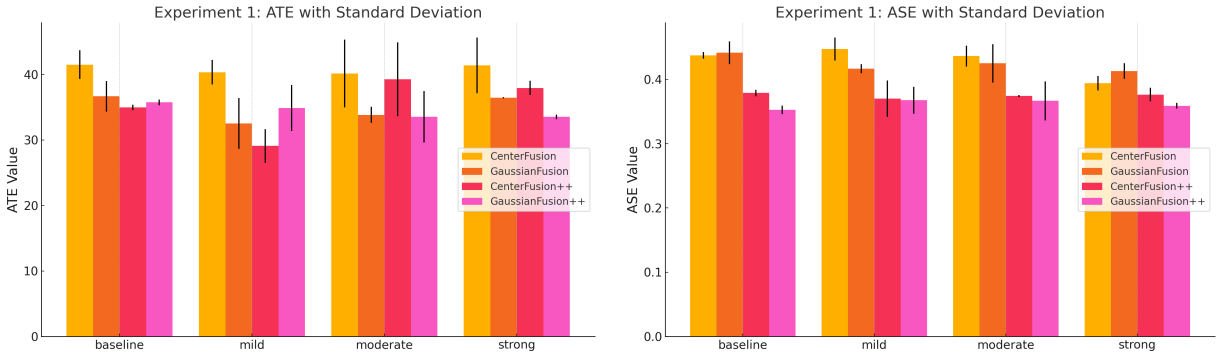


Figure 6.2: Aggregated TP metrics (ATE and ASE) for all models under different static frustum enlargement configurations.

A consistent pattern is observed in the ASE across all levels of frustum enlargement, suggesting that ASE is model-specific. In particular, models employing a learnable frustum association mechanism, CenterFusion++ and GaussianFusion++,consistently outperform those relying on traditional association methods, CenterFusion and GaussianFusion. This trend holds regardless of frustum size, reinforcing the conclusion that scale estimation accuracy is largely model-dependent. Intuitively, one might expect better depth estimation to correlate with improved scale estimation, as both are spatially related and may benefit from shared features. However, this hypothesis is not clearly supported by the experimental results visualised in Figure 6.2.

No systematic trends are observed for AOE and AVE, which is expected since the radar provides only depth and not velocity information. Consequently, no significant benefit for orientation and velocity estimation is obtained from radar fusion in this configuration. Including velocity information from the radar data would likely improve both AOE and AVE, as demonstrated in the original CenterFusion study [37].

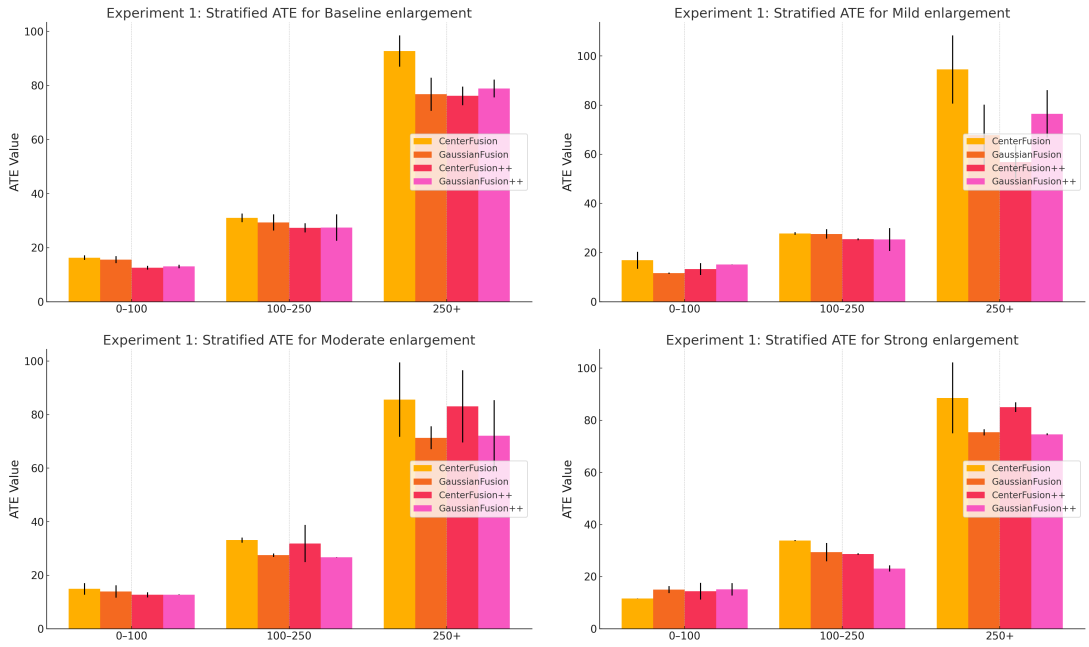


Figure 6.3: Stratified ATE results for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++ under different static frustum enlargement configurations (Baseline, Mild, Moderate, Strong). Each plot shows the ATE grouped by depth ranges: 0–100 m, 100–250 m, and beyond 250 m.

The stratified ATE in Figure 6.3 confirms that CenterFusion's depth accuracy degrades most rapidly at long ranges. This can be observed by CenterFusion consistently producing the highest error beyond 250 meters for

the baseline, mild, and moderate enlargement settings. GaussianFusion and GaussianFusion++ generally maintain lower ATE at greater distances, indicating increased robustness to the noise introduced by larger frustums. CenterFusion++ achieves a notably low overall ATE as well as a surprisingly low stratified far-range ATE of 56.8536 at mild enlargement ($\epsilon = 0.5$ and $\delta = 1$). However, since such a small frustum enlargement is generally insufficient to consistently include the radar points of distant objects, this outcome can be considered coincidental and is best attributed to the way the model was trained at this configuration rather than to effective radar association. As the frustum size increases further, CenterFusion++ shows higher ATE again, suggesting that it is more sensitive to clutter and noise as a result of frustum enlargement than the probabilistic weighting methods used in GaussianFusion and GaussianFusion++.

The stratified ASE, shown in Figure 6.4, exhibits a decreasing trend with increasing depth range. This likely reflects the fact that, at far distances, the dataset contains mostly similar types of vessels, resulting in more consistent object sizes and thus lower scale estimation error. In contrast, at close range, the variety in vessel sizes is much larger, making accurate scale prediction more challenging. For stratified AOE and AVE, shown in Figures A.2.10 and A.2.11, no clear or consistent trends are observed, indicating these metrics are less influenced by object distance in the current setup.

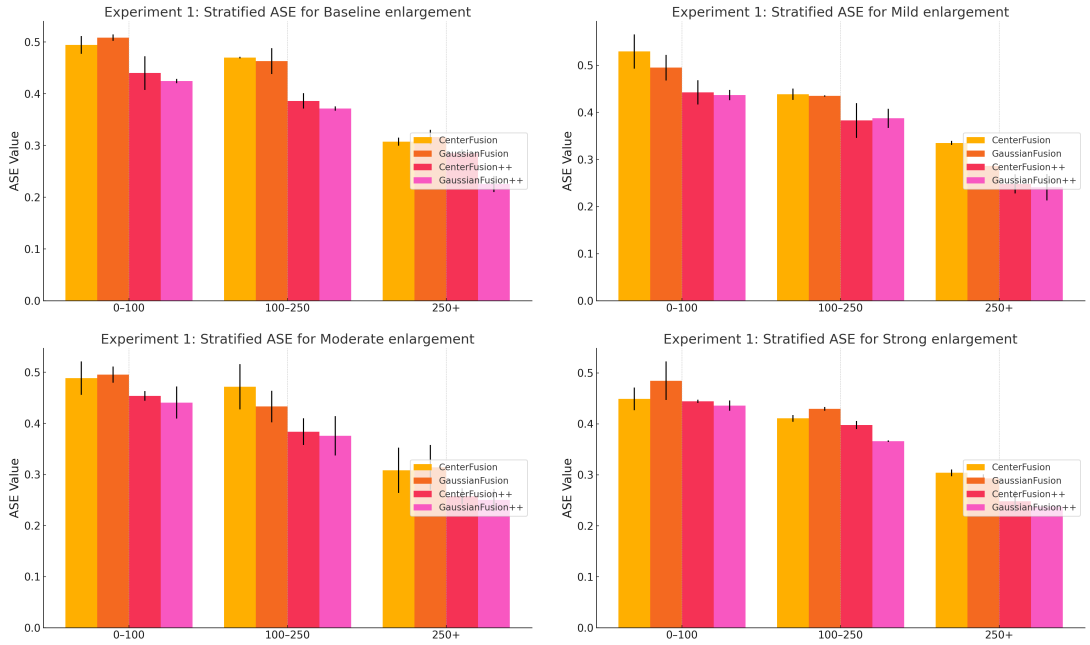


Figure 6.4: Stratified ASE results for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++ under different static frustum enlargement configurations (Baseline, Mild, Moderate, Strong). Each plot shows the ASE grouped by depth ranges: 0–100 m, 100–250 m, and beyond 250 m.

The qualitative results in Appendix A.2.1 illustrate these findings in more detail. For CenterFusion, accurate localisation is achieved when the initial monocular depth estimate is reliable. This can be seen for the cluttered nearby objects in Figure A.2.4 (mild enlargement) and for the distant object in Figure A.2.2 (baseline). When the frustum is enlarged too much, however, the inclusion of additional noise reduces depth accuracy, pulling the detection away from the ground truth as visible in Figure A.2.7 (moderate enlargement).

GaussianFusion shows greater robustness to clutter. In Figure A.2.8 (strong enlargement), one of the two boats near the center of the BEV image is estimated closer to the true position, although the other remains less accurate. In Figure A.2.9, GaussianFusion accurately locates the distant moving vessel at the top-centre of the image.

CenterFusion++ does not exhibit clear robustness to noise but delivers more stable and visually cleaner predictions than CenterFusion on average.

GaussianFusion++ achieves the highest level of resilience to noise among all models. Figures A.2.4, A.2.6, and A.2.8 show that the pair of boats located on the right side of the image is detected closest to the ground truth, even in scenes with significant clutter.

Overall, these findings indicate that larger frustums are beneficial for capturing radar points when monocular depth is unreliable but require robust association to limit the impact of clutter. Probabilistic radar point weighting contributes to more stable depth estimation under complex and cluttered conditions.

6.2 Benefits of depth-adaptive frustum enlargement

Experiment 2 examined the impact of applying a depth-adaptive frustum enlargement module on the performance of the four radar–camera fusion models. This experiment addresses the limitation of static frustum enlargement, which cannot adapt to the increasing uncertainty in monocular depth estimation at larger distances. The dynamic module uses a quadratic scaling function to adjust both the horizontal and depth-wise frustum dimensions based on the predicted object depth, maintaining tight frustums for near objects and enlarging them for distant ones where depth estimates are less reliable. This strategy is expected to improve radar association quality and yield more robust localisation performance, in the form of a lower ATE, across varying depth intervals for all models.

The aggregated results in Figure A.2.14 show that the mAP for CenterFusion and GaussianFusion remains fairly consistent across both static and dynamic frustum configurations, indicating limited sensitivity to moderate changes in frustum size. For CenterFusion++, the mAP under dynamic enlargement is comparable to that of mild static enlargement, which aligns with the fact that the dynamic module parameters ($\epsilon_{base} = 0.5$, $\delta_{base} = 1$) produce a frustum size that closely matches the mild static setting for depths up to approximately 250 meters, where most objects in the custom dataset are located. The mAP for GaussianFusion++ with the dynamic frustum configuration is slightly lower than its mAP at the static baseline configuration.

The NDS results reflect this distinction more clearly. CenterFusion shows minimal variation across configurations. For GaussianFusion and CenterFusion++, the highest NDS is obtained with mild static enlargement, and a lower but still competitive value is observed with the dynamic frustum enlargement configuration. In contrast, GaussianFusion++ achieves its highest NDS under dynamic frustum enlargement, indicating that the combination of probabilistic radar weighting and depth-adaptive frustum sizing improves localisation consistency across different depth intervals for this model in particular.

The ATE results indicate that CenterFusion and CenterFusion++ benefit the most from the dynamic frustum enlargement module with respect to localisation. For GaussianFusion, the lowest ATE is observed at mild static enlargement, and for GaussianFusion++, the best ATE occurs at dynamic enlargement. Among all models, CenterFusion++ achieves the lowest aggregated ATE. This outcome is primarily due to the fact that the dynamic frustum enlargement closely resembles the mild static enlargement within the depth range that contains the majority of training objects. Specifically, using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$ results in an enlargement factor that increases only slightly compared to mild static settings: by approximately 0.05 at 100 meters and up to about 0.31 at 250 meters. Beyond 250 meters, the quadratic term leads to a progressively larger frustum, but as most objects are within 250 meters, the aggregated ATE remains similar between the dynamic and mild static configurations.

The stratified ATE in Figure A.2.15 further clarifies these effects for CenterFusion++. For near-range targets, the dynamic frustum produces a near-range ATE approximately 31.7% lower than that of mild static enlargement, possibly due to very small enlargement (0.05 at 100 m) only just including important radar points for LFANet. In the mid-range, the slight additional enlargement (ranging from 0.05 to 0.3125) compared to the mild static frustum increases the inclusion of radar clutter, resulting in an approximate 12.1% increase in mid-range ATE. At far distances beyond 250 meters, the quadratic term enlarges the frustum substantially, allowing the model to capture radar points that would otherwise be excluded, leading to an improvement in far-range ATE of about 12.7% relative to mild static enlargement. This trade-off illustrates how dynamic frustum enlargement balances reliable near-range learning with increased radar coverage at long ranges, while introducing some performance degradation at mid-range ATE for CenterFusion++ due to additional clutter.

For both GaussianFusion and GaussianFusion++, the stratified ATE results show only minor variations across configurations. GaussianFusion localises better under the static mild configuration compared to the dynamic one, though the difference is relatively small. In the case of GaussianFusion++, the dynamic configuration does not yield the best stratified ATE at near or mid ranges. The lowest near-range ATE is observed with the static moderate configuration, while the mid-range performance of the dynamic configuration is comparable to that of the static mild setup. This is expected, as the frustum enlargements in these configurations are very similar, and the model's probabilistic radar association mechanism, which continuously weighs radar points based on spatial proximity and learned uncertainty, effectively mitigates the impact of minor increases in noise. Notably, GaussianFusion++ achieves its best far-range ATE with the dynamic configuration. Overall, the relatively small variation in ATE across frustum enlargement configurations for both Gaussian-based models underscores their robustness to larger frustum sizes and noise levels.

An example highlighting the robustness of GaussianFusion++ compared to CenterFusion++ is shown in Figure 6.5. A single vessel appears near the center of the BEV image, and radar noise is present directly behind it. The radar noise is spatially close to the vessel's true location. CenterFusion++ uses a deterministic frustum asso-

ciation mechanism that creates an artificial radar point to represent the object. In this case, it incorrectly selects a spurious radar return caused by background clutter, leading to a mislocalisation. In contrast, GaussianFusion++ applies a probabilistic association strategy to create the artificial radar point by learning a bivariate Gaussian over all radar points within the frustum. This allows it to down-weight noisy returns and emphasise points more likely to correspond to the true object location. As a result, GaussianFusion++ accurately localises the vessel. This case highlights the advantage of learning probabilistic radar associations, which improves robustness to clutter and misaligned radar signals.

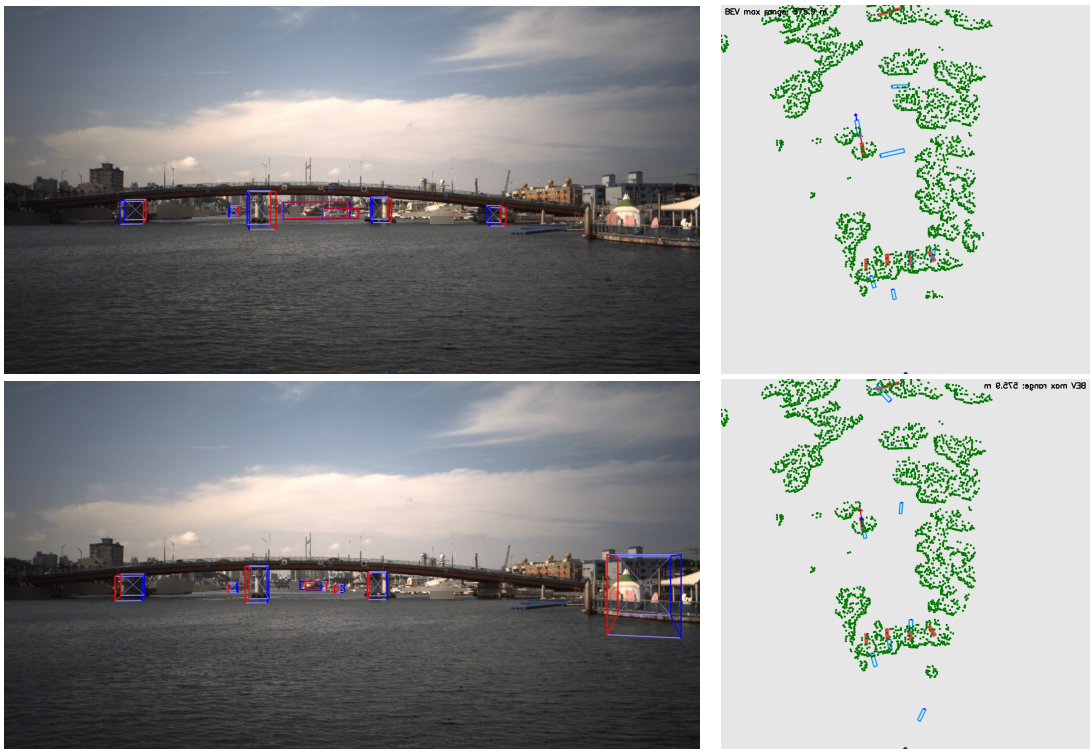


Figure 6.5: Qualitative detection results in a unique scene for CenterFusion++ (first row) and GaussianFusion++ (second row). Each row shows the camera view (left) and the corresponding BEV (right).

The stratified ASE results follow the same pattern as observed with static frustum enlargement settings. CenterFusion++ and GaussianFusion++ achieve better size estimation performance than the other models, but dynamic frustum enlargement does not provide a clear benefit for ASE. For stratified AOE and AVE, no consistent trends are visible. The higher AOE of CenterFusion++ remains apparent and would likely decrease if trained on a larger and more diverse dataset.

Overall, Experiment 2 shows that introducing a depth-adaptive frustum enlargement module can improve localisation performance and maintain robust detection across varying distances. For localisation, CenterFusion and CenterFusion++ benefit the most from dynamic frustum enlargement, as evidenced by their lower aggregated ATE compared to all static configurations. The Gaussian-based models, on the other hand, show minimal differences in performance compared to their best static configurations. This indicates that their probabilistic radar association mechanisms already provide strong robustness to radar noise and clutter, making them less sensitive to changes in frustum size than CenterFusion and CenterFusion++.

The depth-adaptive enlargement function used in this experiment was designed based on the assumption that monocular depth estimation error grows superlinearly with distance [65], and its parameters were selected based on the results from Experiment 1. Although the chosen quadratic form with the current parameters provides a good balance between near- and far-range association, further optimisation could refine this balance and enhance performance. Additionally, different models may benefit from different enlargement functions or parameter settings, as demonstrated by the ablation study in Section 5.2.3, which showed that each model responds differently to changes in dynamic frustum scaling. These functions could be quadratic, linear, or follow another suitable form, depending on the model's radar association mechanism and sensitivity to clutter.

6.3 Influence of uncertainty in GaussianFusion

Experiment 3 examined the impact of varying the scale factor α on the performance of GaussianFusion in combination with dynamic frustum enlargement. The scale factor α governs the spread of the bivariate Gaussian weighting function. Lower values constrain the weighting tightly around the predicted object center, whereas higher values broaden the distribution, allowing radar points at greater distances to contribute more substantially.

The aggregated results in Table 5.11 show that increasing α from the baseline (0.1) to a higher uncertainty setting (0.2) leads to a higher mAP and a slightly increased NDS. This increase in NDS is primarily due to the stronger influence of mAP in its computation. Other metrics indicate that a higher uncertainty does not improve localisation. The ATE, which directly reflects the contribution of depth information from radar, is significantly lower for the baseline setting than for high uncertainty and slightly lower than for low uncertainty. The AOE for the baseline configuration is also notably better, although further improvements are expected once radar velocity information is included. The AVE shows a minor benefit for low uncertainty but remains less conclusive due to the absence of velocity data.

The stratified results in Table 5.12 provide additional context. For ATE, the low uncertainty setting achieves the lowest errors at near and mid ranges, where monocular depth estimates are generally more reliable and tighter weighting is effective at suppressing noise. At greater distances, where depth predictions are less accurate, the baseline uncertainty outperforms both low and high uncertainty settings. The high uncertainty setting does not yield improvements at far range and generally results in higher translation errors.

These findings are consistent with the qualitative examples in Figure A.2.17. In this figure, the distant vessel near the center of the BEV is more accurately localised using the baseline uncertainty ($\alpha = 0.1$) compared to the low uncertainty setting. Conversely, the near and mid-range objects appear slightly better aligned for the low uncertainty configuration, supporting the stratified results.

Overall, the results demonstrate a trade-off between noise suppression and sufficient radar point inclusion. A low α (low uncertainty) can overly limit the association of valid radar points for distant objects, while a high α (high uncertainty) broadens the weighting excessively, introducing clutter and increasing translation errors without notable benefit at larger distances. For the current setup, the baseline uncertainty setting ($\alpha = 0.1$) offers the best overall balance for depth estimation across all ranges. The observed increase in mAP for $\alpha = 0.2$ does not translate to improved depth localisation and mainly affects the NDS due to its heavy weighting for detection accuracy. Selecting an appropriate α is therefore essential for robust radar–camera fusion in maritime environments. Future research could explore dynamically adjusting α according to estimated object depth or detection confidence to further enhance robustness against monocular depth uncertainty.

6.4 Contribution of token-mixing in RGPNet

Experiment 4 investigated the impact of introducing token-mixing in the RGPNet architecture by comparing two variants: RGPNet-A and RGPNet-B. Both models are used within the GaussianFusion++ pipeline and were trained under identical conditions using the dynamic frustum enlargement configuration from Experiment 2. The goal was to evaluate whether structured inter-point reasoning improves the estimation of Gaussian parameters for radar–camera fusion, and consequently enhances depth-based localisation performance.

RGPNet-A follows a shallow architecture that processes radar points independently using two fully connected layers, followed by a global max pooling operation. It captures only channel-wise features and ignores spatial relationships between radar points. RGPNet-B extends this design with a token-mixing MLP block that enables inter-point communication by operating across a fixed-length radar point sequence. This added depth and complexity allows the model to learn structured spatial patterns within the radar frustum.

The aggregated results in Table 5.13 show that RGPNet-B slightly outperforms RGPNet-A in both mAP and NDS, indicating improved overall detection quality. More importantly, the stratified ATE results in Table 5.14 reveal that RGPNet-B yields lower localisation error for both near and far-range targets. These improvements align with the expectation that modelling inter-point structure helps resolve ambiguity when the radar return is either dense, at near-range, or sparse, at far-range. The qualitative results in Figures A.2.20 and A.2.21 further support this, as RGPNet-B produces bounding boxes that more accurately align with the ground truth for both nearby and distant objects.

Interestingly, RGPNet-B performs worse than RGPNet-A at mid-range, where it records a higher ATE. This suggests that in scenarios where monocular depth estimates are relatively strong, the added complexity of token-mixing may not provide additional benefit and may even lead to overfitting. This is consistent with established

findings that deeper networks, such as RGPNet-B, are more prone to overfitting in low-data regimes, while shallower architectures like RGPNet-A often generalise more reliably [68].

The results of ASE, AOE, and AVE do not exhibit a consistent or theoretically grounded pattern across the two RGPNet variants and are not expected to be directly influenced by the radar-based depth fusion mechanism. As such, they are not considered relevant for evaluating the contribution of token-mixing in this context.

Beyond this direct comparison, additional ablation studies provide further insight into the role of token count and architecture design. Varying the number of radar tokens N in RGPNet-B revealed that $N = 128$ offers the best balance between geometric detail and computational efficiency. Smaller values (e.g., $N = 64$) led to under-representation of spatial structure, while larger values (e.g., $N = 256$) introduced unnecessary complexity and noise. Furthermore, substituting RGPNet-B with a PointNet++ backbone slightly reduced ATE but at the cost of significantly lower mAP, suggesting that while PointNet++ can localise well when it detects, it does so less consistently than RGPNet-B.

Finally, a threshold sweep analysis confirmed the practical benefit of the best-performing configuration. GaussianFusion++ with RGPNet-B achieved an optimal F1 score of 0.8284 at a detection confidence threshold of 0.39. This balance between precision and recall reinforces the model's suitability for deployment in safety-critical maritime applications, where both accurate and consistent detection are essential.

In summary, RGPNet-B demonstrates advantages for nearby and far-away object localisation, while RGPNet-A excels in the mid-range. On the current benchmark dataset, the two variants perform comparably overall, but the structured reasoning enabled by token-mixing may prove increasingly beneficial if trained on a larger and more diverse dataset. The token-mixing design of RGPNet-B, especially when configured with $N = 128$, strikes a strong balance between expressiveness and stability, and shows promising potential for deployment-oriented robustness.

Chapter 7
Conclusion

This thesis investigated methods to enhance situational awareness for ASVs through improved multi-modal 3D object detection using radar and camera data. The work addressed the unique challenges posed by maritime environments, including broad depth variation, radar–camera misalignment, sensor noise, a lack of annotated training data, and degradation of monocular depth estimation with increasing range.

A fully annotated benchmark dataset was constructed using the Pohang Canal Dataset [42] as a foundation, enabling supervised learning and evaluation of detection models under realistic maritime conditions. The raw dataset included synchronised radar and camera images, LiDAR point clouds, and navigation data. These sensor modalities were used to manually generate 3D object annotations for static and moving maritime objects across varying distances.

Two novel fusion architectures were proposed. GaussianFusion introduced a bivariate Gaussian weighting mechanism for radar association, offering improved robustness to sparse and noisy radar returns. GaussianFusion++ extended this with a learnable radar association network, RGPNet, which estimates the Gaussian parameters based on radar point sets within a frustum. Two variants of RGPNet were evaluated: a lightweight point-wise version (RGPNet-A) and a token-mixing version (RGPNet-B) capable of modelling inter-point dependencies.

To improve radar–camera association at long range, a depth-adaptive frustum enlargement strategy was introduced. This method dynamically adjusts the frustum volume in proportion to predicted object depth, enhancing the inclusion of radar returns when monocular depth estimation becomes less reliable.

Four experiments were conducted to evaluate the proposed contributions. The results demonstrated that GaussianFusion and GaussianFusion++ achieved more robust depth estimation under large, fixed frustum settings and reduced sensitivity to clutter compared to existing methods. Depth-adaptive frustum enlargement further improved localisation at long distances while maintaining near-field precision. The use of token-mixing in RGPNet-B was shown to slightly enhance performance for near and far objects, with slightly reduced accuracy at mid-range, likely due to data sparsity and reduced radar resolution within that depth interval.

In summary, this thesis demonstrated that probabilistic radar fusion and depth-aware frustum association can significantly improve 3D object detection in complex maritime settings. The methods proposed here contribute to more robust and interpretable perception systems for ASVs, and form a basis for future work on uncertainty-aware and data-efficient multi-modal fusion in real-world autonomous navigation.

7.1 Future work & recommendations

Although the developed models and methods demonstrate clear improvements for maritime 3D object detection, several aspects remain open for further investigation and optimisation. These potential research directions include model refinements, improved frustum enlargement and uncertainty handling, enhanced data quality, and alternative fusion strategies.

One area for improvement is the radar association mechanism in GaussianFusion. Instead of assuming independence in the bivariate Gaussian weighting, the predicted object orientation could be incorporated to estimate a non-zero correlation coefficient (ρ). Furthermore, a dynamic scale factor α that adjusts according to object distance could help the model represent uncertainty more realistically, potentially improving depth estimation at greater ranges.

RGPNet in GaussianFusion++ could benefit from additional refinements. Pretraining RGPNet on point cloud data with full annotations may accelerate convergence and produce more reliable Gaussian parameters. A curriculum learning scheme that gradually increases the influence of the NLL loss could lead to more stable training and more robust uncertainty modelling. Providing additional inputs such as the predicted object size and orientation might further improve RGPNet's ability to detect relevant radar point clusters and estimate the Gaussian parameters more effectively.

Furthermore, investigating the maximum range at which the camera remains a useful modality is an important next step. Extending the dynamic depth enlargement approach to cover such long-range scenarios, for example up to three kilometers, could help identify the point at which radar should take precedence due to diminishing camera performance. Moreover, the frustum enlargement strategy could be made more adaptive by analysing how

the predicted object orientation and scale affect the optimal frustum shape. Specifically, future methods should incorporate predicted object scale into the enlargement function so that smaller objects receive proportionally greater enlargement, ensuring that frustum size remains contextually appropriate. Introducing fuzzy logic into this dynamic frustum control could further refine enlargement transitions across distance intervals.

Beyond frustum tuning, more principled methods for compensating temporal misalignment should be explored. Rather than relying solely on horizontal frustum enlargement, future approaches could directly adjust the radar data by applying transformations derived from the OS's velocity, heading, and angular motion. This would enable different sections of the radar scan to be temporally corrected relative to the camera frame, reducing misalignment across the entire sweep and improving the consistency of radar–camera fusion. Eliminating horizontal frustum enlargement altogether would also help minimise the inclusion of noise, further enhancing the precision of the fused detections.

In addition to radar and camera, incorporating LiDAR into the multi-modal framework could be an interesting direction for future research. LiDAR provides far more detailed spatial measurements than radar but can be more sensitive to adverse weather conditions. Evaluating the effect of a probabilistic weighting scheme for frustum association could prove highly beneficial for robust sensor fusion in harsh weather conditions when LiDAR data is included. Moreover, integrating Automatic Identification System (AIS) data into the detection process could offer an initial estimate of object positions, orientations, and velocities at specific moments, which, when combined with early fusion strategies, might further improve detection accuracy and monocular 3D object detection. However, since AIS message frequency is typically low, each model must remain robust when AIS data is unavailable or be able to reconcile the timestamp of the AIS message with the OS's position and heading at that time, along with its current navigation state.

Different sensor fusion techniques should also be considered. Incorporating attention mechanisms or explicit modality interaction modules may help exploit complementary radar, camera, LiDAR, and AIS information more effectively, particularly in cluttered or visually challenging maritime conditions.

Improving the training dataset remains an essential step towards developing more robust and generalisable models. Collecting additional data with a broader variety of scenes, weather conditions, traffic scenarios, and greater distance diversity would enhance the model's ability to generalise to different operational environments. In particular, covering a wider range of distances would enable stratified performance evaluation at finer intervals, supporting a more precise understanding of how to adjust dynamic frustum enlargement. Addressing the current imbalance by significantly increasing the proportion of moving objects is crucial for improving motion estimation and velocity prediction accuracy. Expanding the dataset to cover multiple object classes would enable more fine-grained detection, which is important for safe navigation in complex port areas. Furthermore, embedding velocity information and RCS directly into the radar data would provide more informative features for radar association and fusion. A richer and more balanced dataset would also support systematic determination of optimal frustum enlargement parameters for different depth intervals, allowing more precise radar–camera fusion across a wide range of distances.

Finally, the effect of dynamic uncertainty scaling on far-range ATE should be studied in greater detail. Analysing how increased predicted uncertainty influences localisation at long distances could help optimise depth-adaptive frustum strategies.

By pursuing these research directions, future work can build upon the current framework to develop a more flexible, reliable, and generalisable multi-modal detection system, thereby improving situational awareness and operational safety for autonomous surface vessels in complex maritime environments.

Appendix

A.1 Algorithms

Algorithm 1 Radar image to pseudo point cloud conversion

```
procedure RadarToPointCloud(radar_img)
    img ← load_grayscale(radar_img)
    center ← image center
    img ← blacken circular region around center
    blobs ← connected components in img
    for all blob ∈ blobs do
        centroid ← mean(blob)
        blob_size ← size(blob)
        has_white ← any(pixel = 255)
        for all pixel ∈ blob do
            if random() < chance_of_being_chosen(pixel, blob_size, centroid, has_white) then
                if check_distance_to_existing_points(pixel) then
                    intensity ← pixel intensity or local average
                    rcs ← estimate RCS from intensity and blob size
                    (x, y) ← pixel-to-metric-coords(pixel, center)
                    add (x, y, 0, rcs) to point cloud
                end if
            end if
        end for
    end for
    return formatted point cloud with x, y, z, rcs, and empty fields for nuScenes format
end procedure
```

Algorithm 2 Radar-point fusion using a bivariate Gaussian distribution

procedure GAUSSIANFUSEDPOINT($pc\_dep, bbox, dep, dist\_thresh, nonzero\_inds, opt$)
  $(h\_idxs, w\_idxs) \leftarrow nonzero\_inds$
  $values \leftarrow pc\_dep[h\_idxs, w\_idxs]$
  $valid \leftarrow (values < dep + dist\_thresh) \wedge (values > \max(0, dep - dist\_thresh))$
  if $\text{sum}(valid) < 1$ **then**
    return 0.0
  end if
  $values \leftarrow values[valid]$
  $xs \leftarrow w\_idxs[valid]$
  $coords \leftarrow \text{stack}(xs, values)$
  $mean\_x \leftarrow \frac{bbox[0]+bbox[2]}{2}$
  $mean\_z \leftarrow dep$
  $mean \leftarrow [mean\_x, mean\_z]$
  $w \leftarrow bbox[2] - bbox[0]$
  if *opt* has *gaussian_cov_scale* **then**
    $\alpha \leftarrow opt.gaussian\_cov\_scale$
  else
    $\alpha \leftarrow 0.1$
  end if
  $cov \leftarrow \begin{bmatrix} \alpha \cdot w^2 & 0 \\ 0 & \alpha \cdot (2 \cdot dist\_thresh)^2 \end{bmatrix}$
  $inv\_cov \leftarrow \text{inverse}(cov)$
  $diff \leftarrow coords - mean$
  $exponent \leftarrow -0.5 \cdot \text{sum}(diff \cdot inv\_cov \cdot diff)$
  $weights \leftarrow \exp(exponent)$
  $weights \leftarrow weights/(\text{sum}(weights) + 1e{-}6)$
  return $\text{sum}(values \cdot weights)$
end procedure

A.2 Extra results

Figure A.2.1 provides comprehensive ground truth data for two representative samples from the validation set. The left column corresponds to sample 1, and the right column to sample 2. Each column shows, from top to bottom, the raw camera image without annotations, the same camera image with ground truth 3D bounding boxes projected onto it, and the corresponding BEV representation with point cloud data and annotated objects. These examples serve as a consistent reference for visually assessing the quality of the detection results presented in the subsequent sections of this appendix for all experiments.

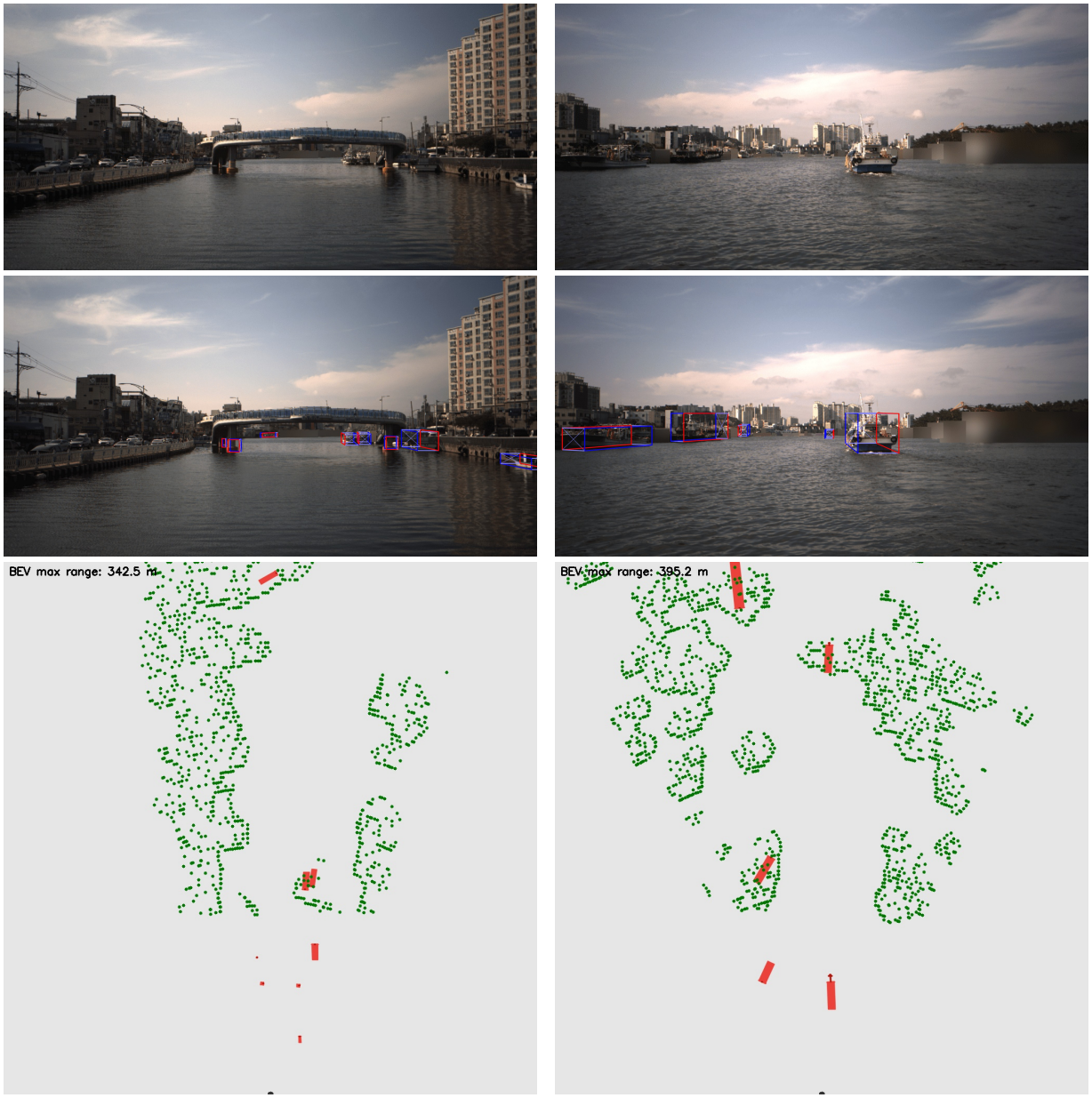


Figure A.2.1: Qualitative ground truth data for two scenarios: the left column shows sample 1 and the right column shows sample 2. Each column includes (from top to bottom) the raw camera image, the annotated camera image with ground truth 3D bounding boxes, and the corresponding BEV with point cloud and annotations.

A.2.1 Experiment 1

Predictions using baseline enlargement

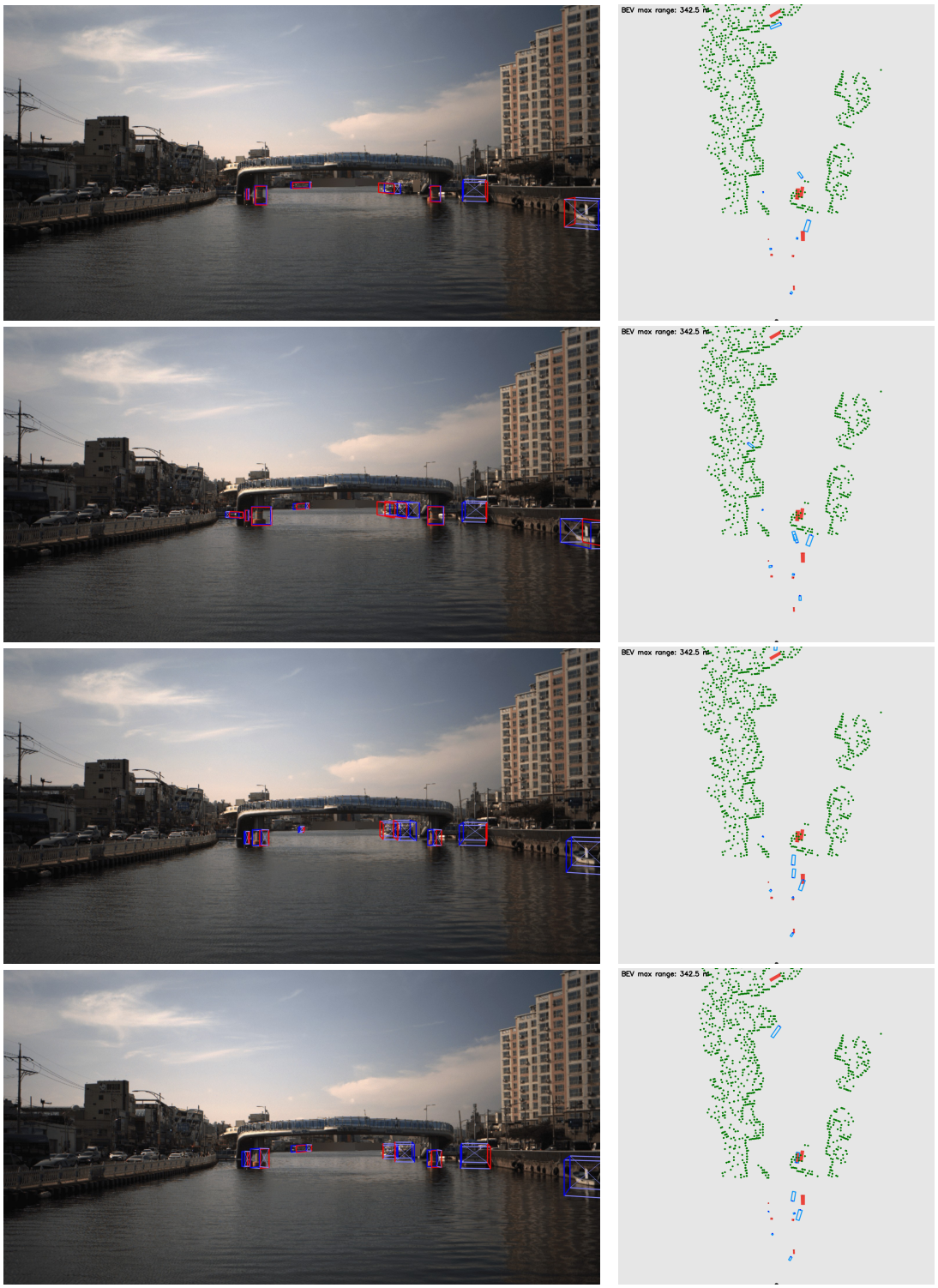


Figure A.2.2: Qualitative detection results on sample 1 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under $\epsilon = 0$ and $\delta = 0$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

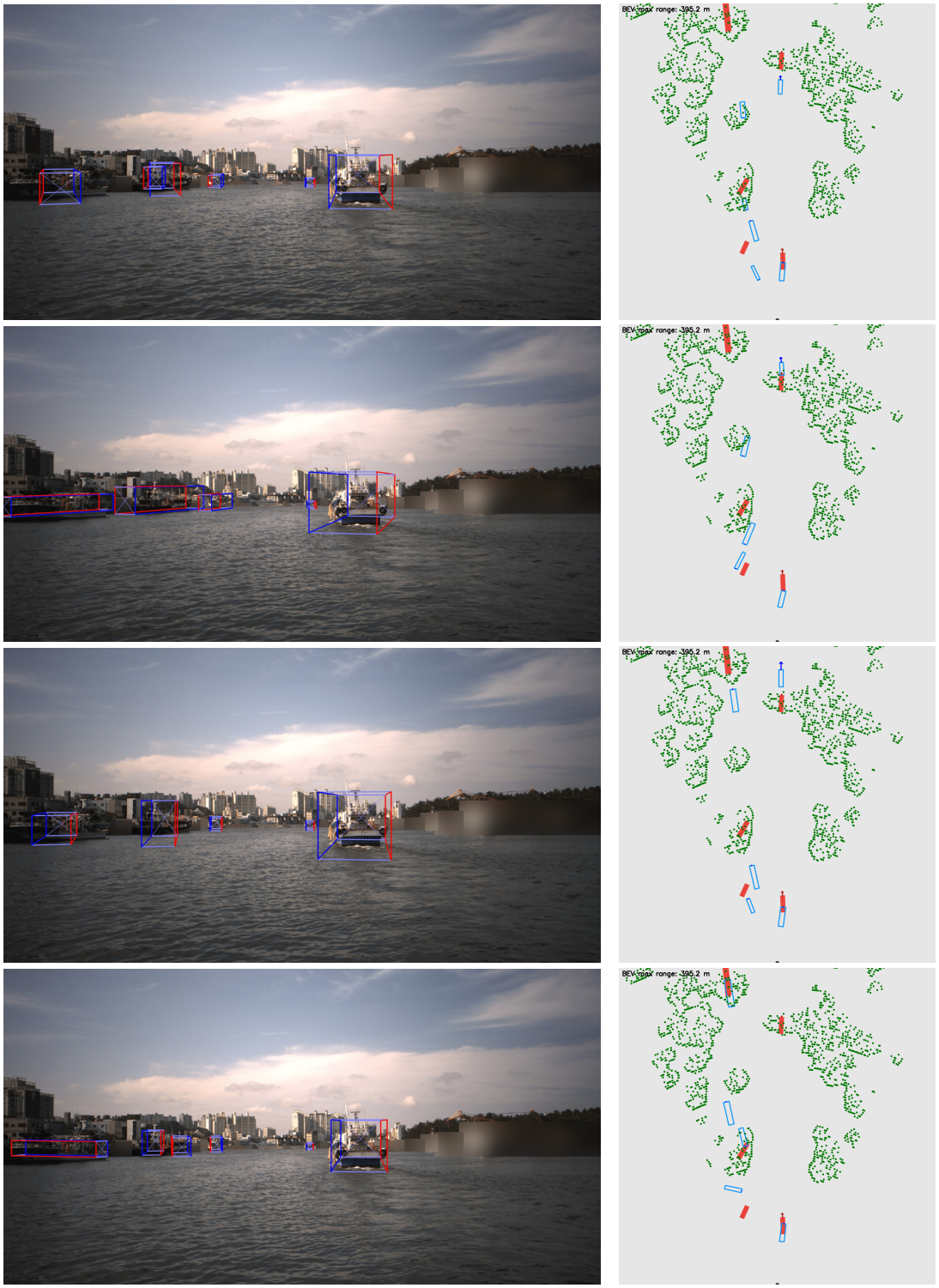


Figure A.2.3: Qualitative detection results on sample 2 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under $\epsilon = 0$ and $\delta = 0$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

Click **here** to jump back to Table A.2.1.

Predictions using mild enlargement

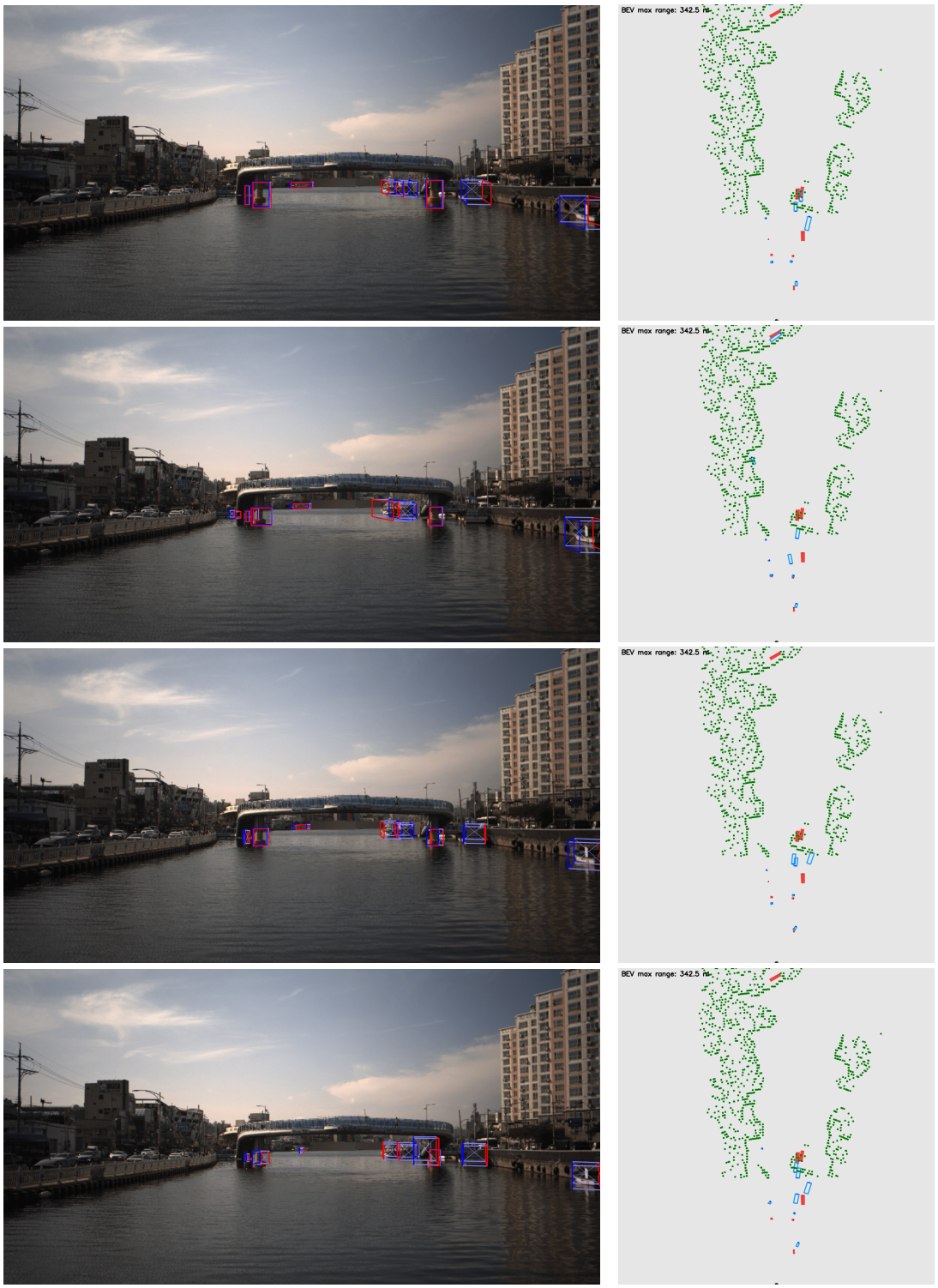


Figure A.2.4: Qualitative detection results on sample 1 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under $\epsilon = 0.5$ and $\delta = 1$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

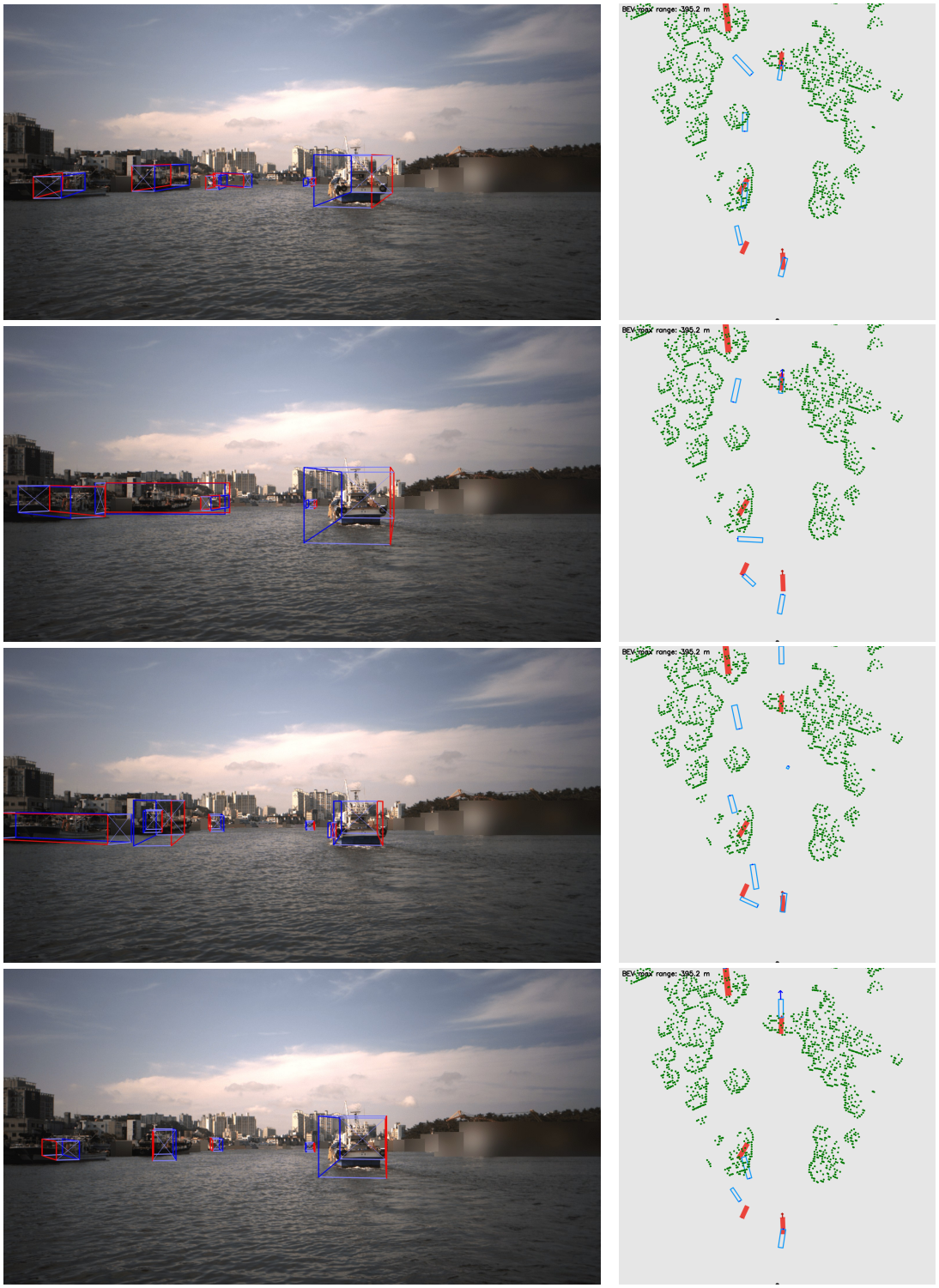


Figure A.2.5: Qualitative detection results on sample 2 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under $\epsilon = 0.5$ and $\delta = 1$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

Click **here** to jump back to Table A.2.2.

Predictions using moderate enlargement

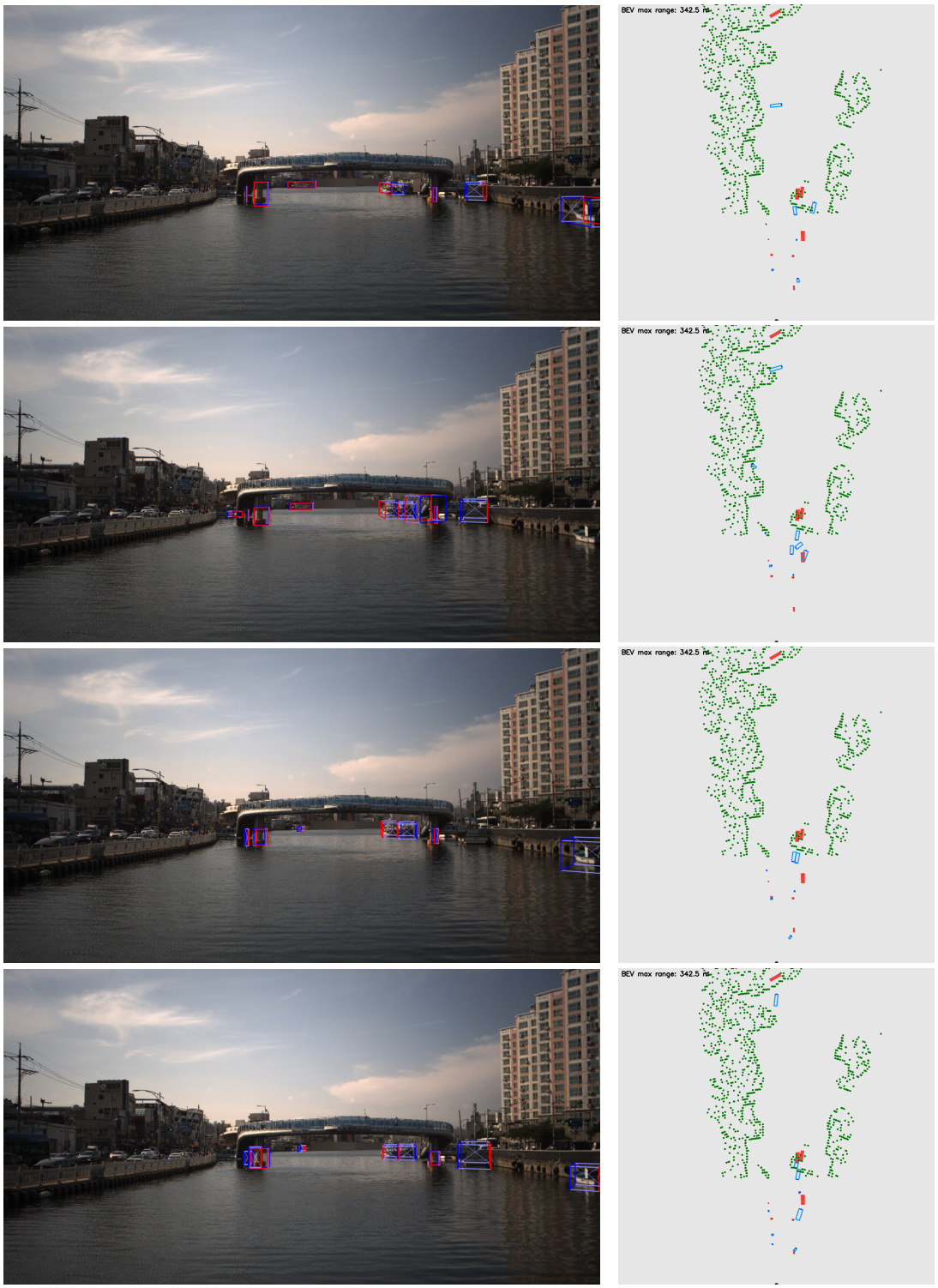


Figure A.2.6: Qualitative detection results on sample 1 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under $\epsilon = 2$ and $\delta = 3$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

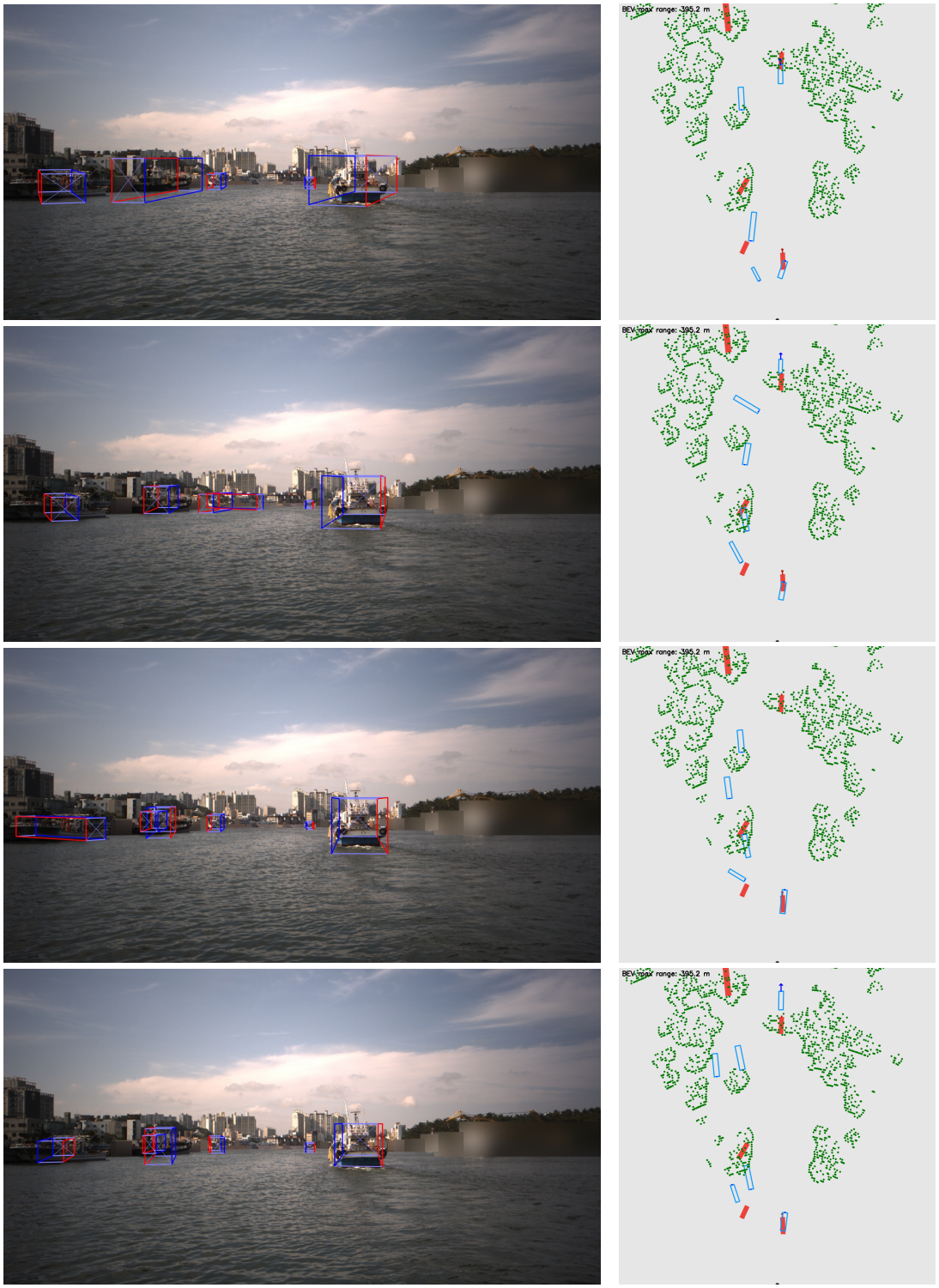


Figure A.2.7: Qualitative detection results on sample 2 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under $\epsilon = 2$ and $\delta = 3$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

Click **here** to jump back to Table A.2.3.

Predictions using strong enlargement

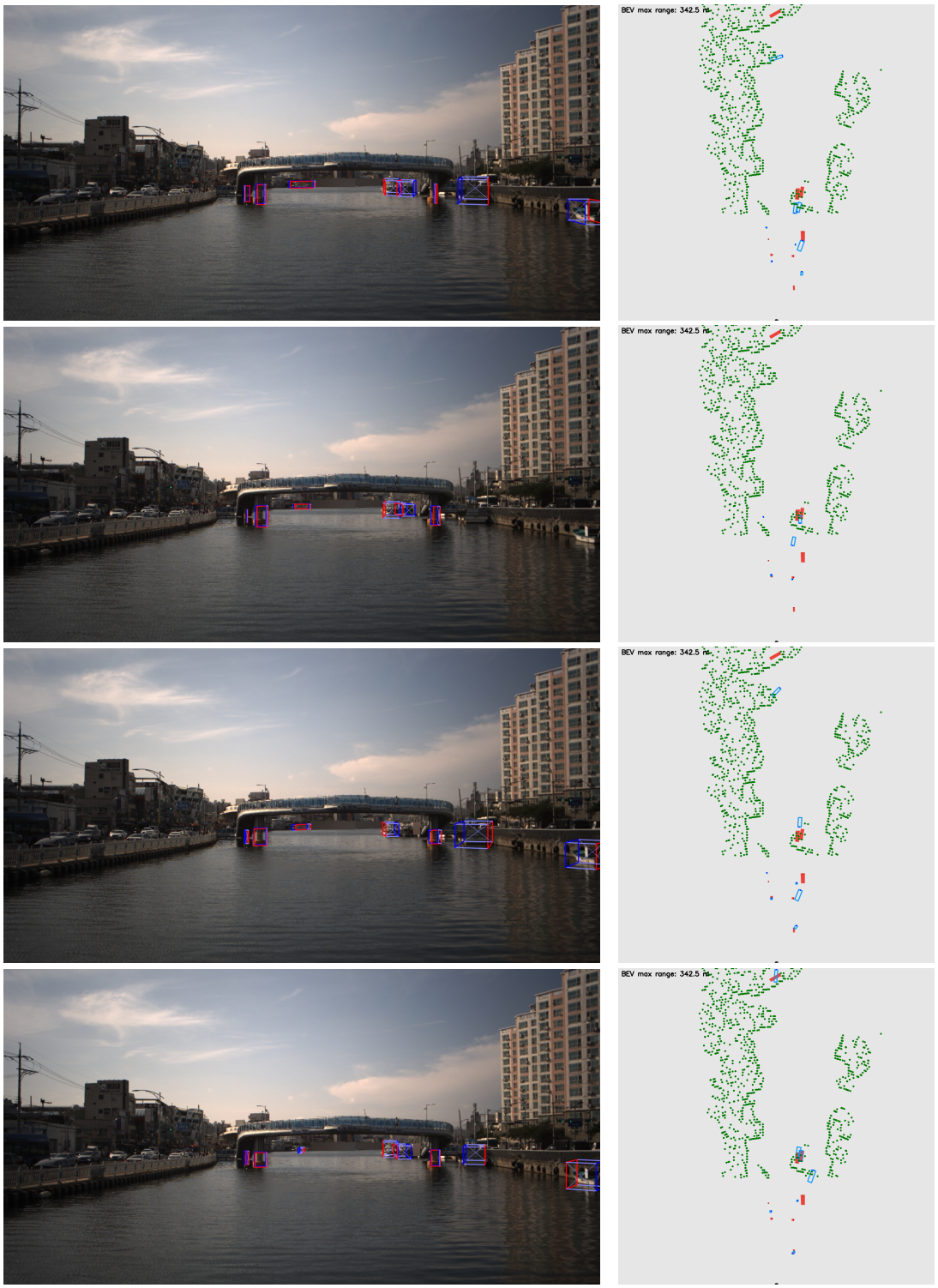


Figure A.2.8: Qualitative detection results on sample 1 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under $\epsilon = 5$ and $\delta = 8$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

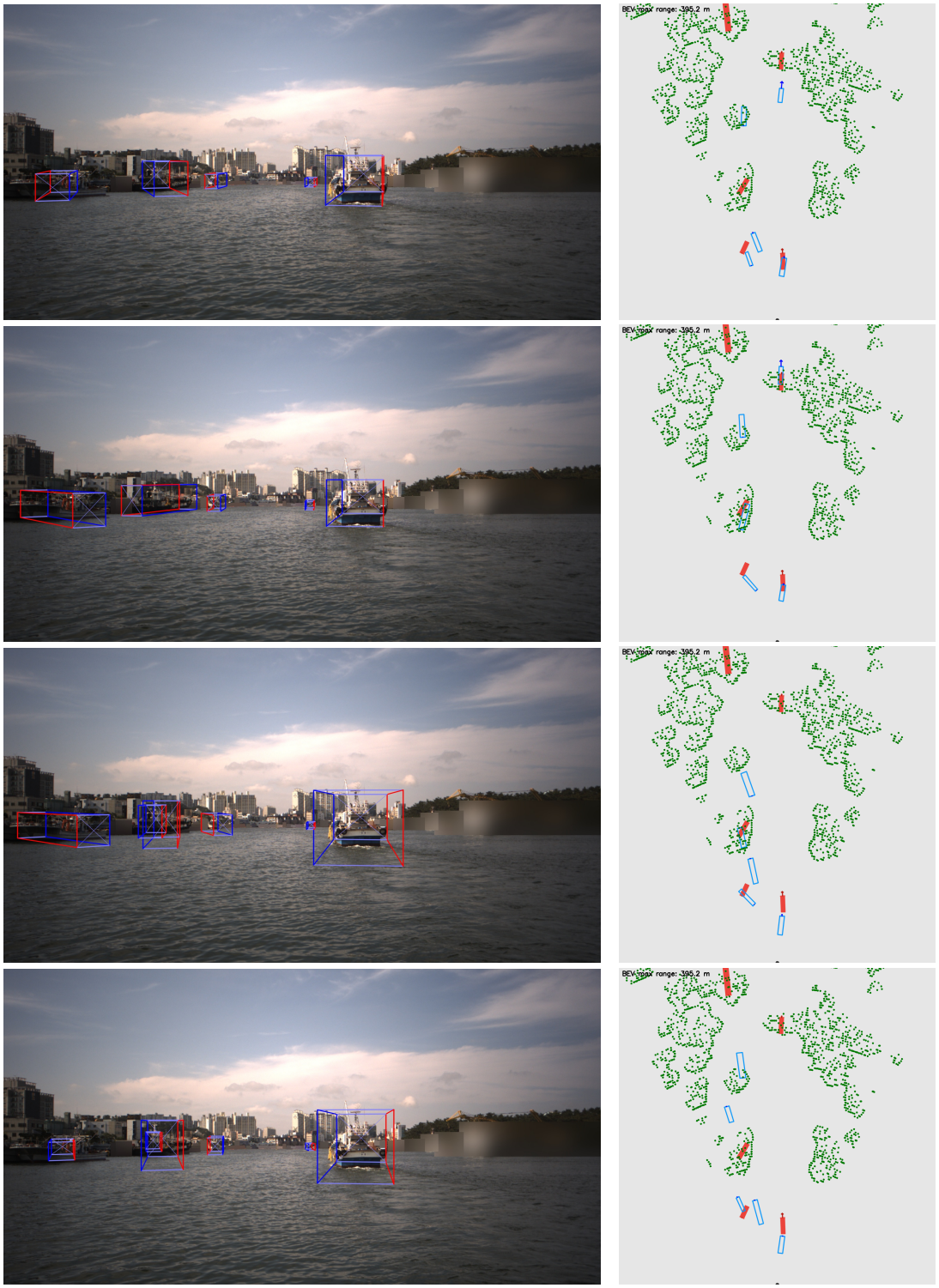


Figure A.2.9: Qualitative detection results on sample 2 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under $\epsilon = 5$ and $\delta = 8$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

Click **here** to jump back to Table A.2.4.

Tables of aggregated results

Table A.2.1 shows the aggregated performance metrics for all four models without any frustum enlargement ($\epsilon = 0$, $\delta = 0$). In this setting, GaussianFusion++ achieves the highest mAP and NDS as well as the lowest ASE. CenterFusion++ achieves the lowest ATE, while CenterFusion and GaussianFusion achieve the lowest AOE and AVE, respectively. Graphical representations of these results can be found in Figures A.2.2 and A.2.3.

Table A.2.1: Aggregated performance metrics for all four models with static frustum parameters $\epsilon = 0$ and $\delta = 0$.

Model	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
CenterFusion	0.4164 ± 0.0087	41.5052 ± 2.1905	0.4373 ± 0.0052	**0.5304** ± 0.0894	0.3308 ± 0.0231	0.5761 ± 0.0048
GaussianFusion	0.4068 ± 0.0061	36.6838 ± 2.3324	0.4415 ± 0.0178	0.6838 ± 0.0709	**0.3248** ± 0.0357	0.5708 ± 0.0045
CenterFusion++	0.3978 ± 0.0186	**34.9797** ± 0.4250	0.3790 ± 0.0049	0.6881 ± 0.0436	0.3894 ± 0.0173	0.5752 ± 0.0085
GaussianFusion++	**0.4276** ± 0.0467	35.7586 ± 0.4400	**0.3527** ± 0.0066	0.7614 ± 0.1339	0.3372 ± 0.0344	**0.5902** ± 0.0182

Table A.2.2 shows the aggregated performance metrics for all four models with mild frustum enlargement ($\epsilon = 0.5$, $\delta = 1$). In this configuration, GaussianFusion++ achieves the highest mAP, and the lowest ASE and AVE. CenterFusion++ achieves the highest NDS, and the lowest ATE and AOE. Graphical representations of these results can be found in Figures A.2.4 and A.2.5.

Table A.2.2: Aggregated performance metrics for all four models with static frustum parameters $\epsilon = 0.5$ and $\delta = 1$.

Model	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
CenterFusion	0.4084 ± 0.0114	40.3542 ± 1.8636	0.4474 ± 0.0178	0.7080 ± 0.0812	0.3614 ± 0.0265	0.5649 ± 0.0040
GaussianFusion	0.4122 ± 0.0156	32.5332 ± 3.8991	0.4168 ± 0.0070	0.6058 ± 0.1016	0.4162 ± 0.0847	0.5837 ± 0.0187
CenterFusion++	0.4113 ± 0.0229	**29.0912** ± 2.5761	0.3700 ± 0.0282	**0.5920** ± 0.0095	0.3641 ± 0.0325	**0.5946** ± 0.0183
GaussianFusion++	**0.4156** ± 0.0025	34.8968 ± 3.5184	**0.3676** ± 0.0209	0.6078 ± 0.0018	**0.3395** ± 0.0104	0.5895 ± 0.0082

Table A.2.3 displays the aggregated metrics for all four models with moderate frustum enlargement ($\epsilon = 2$, $\delta = 3$). In this setting, CenterFusion reaches the highest mAP, and the lowest AOE and AVE. GaussianFusion achieves the highest NDS. GaussianFusion++ achieves the lowest ATE and ASE. Graphical representations of these results can be found in Figures A.2.6 and A.2.7.

Table A.2.3: Aggregated performance metrics for all four models with static frustum parameters $\epsilon = 2$ and $\delta = 3$.

Model	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
CenterFusion	**0.4104** ± 0.0120	40.1490 ± 5.1830	0.4364 ± 0.0162	**0.5802** ± 0.0189	**0.3408** ± 0.0137	0.5728 ± 0.0034
GaussianFusion	0.4100 ± 0.0074	33.8420 ± 1.2236	0.4251 ± 0.0298	0.6782 ± 0.0898	0.3727 ± 0.0004	**0.5776** ± 0.0051
CenterFusion++	0.3904 ± 0.0045	39.2856 ± 5.6265	0.3742 ± 0.0015	0.5936 ± 0.0391	0.3620 ± 0.0679	0.5709 ± 0.0057
GaussianFusion++	0.3903 ± 0.0020	**33.5362** ± 3.9145	**0.3668** ± 0.0304	0.7316 ± 0.0297	0.3585 ± 0.0088	0.5735 ± 0.0091

Table A.2.4 presents the aggregated performance metrics for all four models with strong frustum enlargement ($\epsilon = 5$, $\delta = 8$). CenterFusion++ shows the highest mAP and the highest NDS in this configuration. GaussianFusion++ achieves the lowest ATE, ASE, AOE, and AVE. Graphical representations of these results can be found in Figures A.2.8 and A.2.9.

Table A.2.4: Aggregated performance metrics for all four models with static frustum parameters $\epsilon = 5$ and $\delta = 8$.

Model	mAP ↑	ATE ↓	ASE ↓	AOE ↓	AVE ↓	NDS ↑
CenterFusion	0.4041 ± 0.0068	41.3911 ± 4.2547	0.3943 ± 0.0113	0.6610 ± 0.1310	0.4036 ± 0.0688	0.5694 ± 0.0094
GaussianFusion	0.4030 ± 0.0073	36.4467 ± 0.0804	0.4132 ± 0.0120	0.6811 ± 0.0283	0.3513 ± 0.0038	0.5726 ± 0.0062
CenterFusion++	**0.4342** ± 0.0147	37.9459 ± 1.0767	0.3764 ± 0.0107	0.6922 ± 0.0221	0.3244 ± 0.0148	**0.5908** ± 0.0036
GaussianFusion++	0.4092 ± 0.0546	**33.5373** ± 0.3370	**0.3590** ± 0.0046	**0.6580** ± 0.0902	**0.3154** ± 0.0388	0.5874 ± 0.0252

The results presented in Tables A.2.1 to A.2.4 are visualised as histograms in Figures 6.1 and 6.2 in Experiment 1.

Stratified AOE results

Table A.2.5: Stratified AOE of all four models under different static frustum enlargement settings. Metrics are stratified by distance: near (< 100 m), mid (100–250 m), and far (> 250 m).

ϵ	δ	CenterFusion AOE ↓			GaussianFusion AOE ↓			CenterFusion++ AOE ↓			GaussianFusion++ AOE ↓		
		Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
0.0	0.0	**0.5612** ± 0.1624	**0.4618** ± 0.0414	**0.5874** ± 0.0622	0.7607 ± 0.1191	0.5284 ± 0.0545	0.7931 ± 0.0396	0.6530 ± 0.0040	0.7267 ± 0.0670	0.6826 ± 0.0646	0.7691 ± 0.0365	0.7632 ± 0.1768	0.7472 ± 0.2134
0.5	1.0	0.7082 ± 0.0248	0.6671 ± 0.0911	0.7666 ± 0.1539	0.7256 ± 0.0886	**0.5424** ± 0.1404	**0.5294** ± 0.0897	**0.5589** ± 0.0106	0.5713 ± 0.0341	0.6655 ± 0.0957	0.5893 ± 0.0214	0.5966 ± 0.0029	0.6488 ± 0.0177
2.0	3.0	**0.4978** ± 0.0073	0.5957 ± 0.1879	0.6566 ± 0.2125	0.6769 ± 0.0404	**0.5763** ± 0.1670	0.8218 ± 0.0626	0.5549 ± 0.1394	0.6324 ± 0.0144	**0.5976** ± 0.0641	0.6279 ± 0.0604	0.8053 ± 0.1059	0.7772 ± 0.0507
5.0	8.0	0.6518 ± 0.0637	0.6637 ± 0.2773	**0.6730** ± 0.1966	0.7934 ± 0.1119	**0.5692** ± 0.0578	0.6836 ± 0.1269	0.6680 ± 0.1597	0.6638 ± 0.0610	0.7670 ± 0.0595	**0.5885** ± 0.0523	0.6877 ± 0.1144	0.7190 ± 0.1136

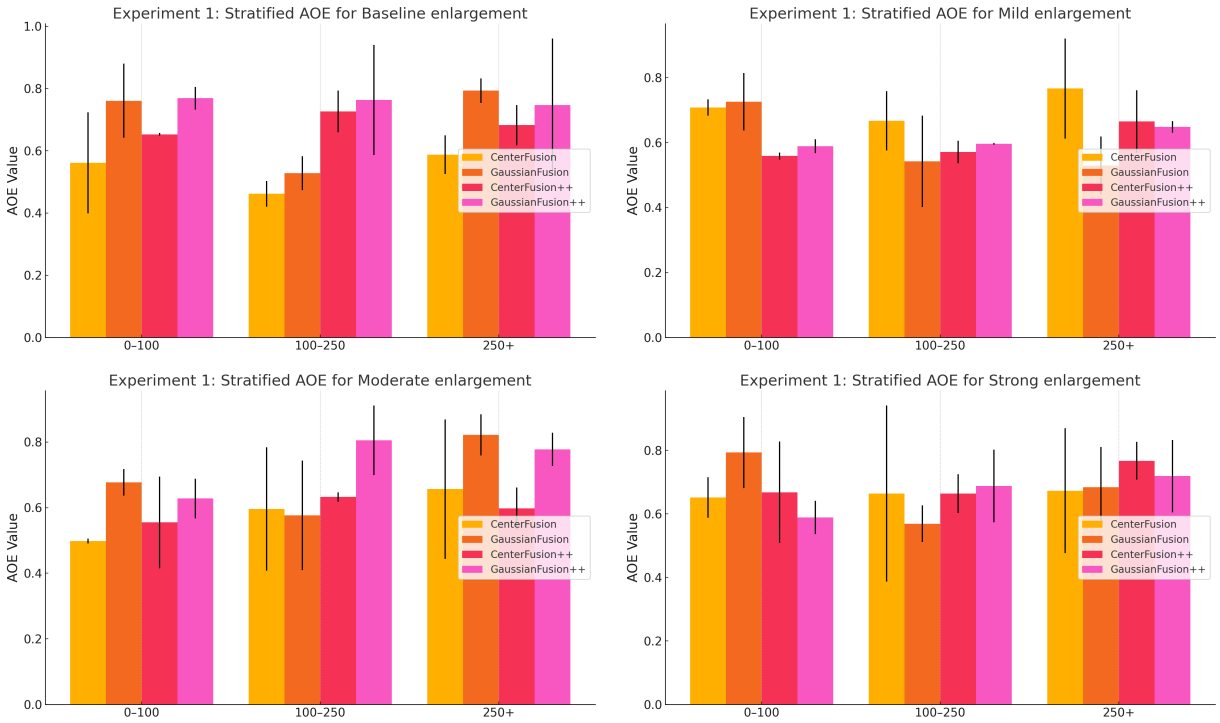


Figure A.2.10: Stratified AOE results for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++ under different static frustum enlargement configurations (Baseline, Mild, Moderate, Strong). Each plot shows the AOE grouped by depth ranges: 0–100 m, 100–250 m, and beyond 250 m.

Click **here** to jump back to Experiment 1.

Stratified AVE results

Table A.2.6: Stratified AVE of all four models under different static frustum enlargement settings. Metrics are stratified by distance: near (< 100 m), mid (100–250 m), and far (> 250 m).

ϵ	δ	CenterFusion AVE ↓			GaussianFusion AVE ↓			CenterFusion++ AVE ↓			GaussianFusion++ AVE ↓		
		Near	Mid	Far	Near	Mid	Far	Near	Mid	Far	Near	Mid	Far
0.0	0.0	**0.1250** ± 0.0506	**0.2780** ± 0.0030	0.7090 ± 0.1944	0.1268 ± 0.0658	0.3077 ± 0.0103	0.6265 ± 0.0338	0.1899 ± 0.0272	0.4090 ± 0.0230	0.6357 ± 0.0555	0.1698 ± 0.0002	0.3244 ± 0.0372	**0.5914** ± 0.1839
0.5	1.0	**0.1524** ± 0.0091	**0.2999** ± 0.0123	0.7688 ± 0.1143	0.2037 ± 0.0414	0.3502 ± 0.0822	0.7953 ± 0.1014	0.1700 ± 0.0090	0.3666 ± 0.0414	0.6402 ± 0.0499	0.1761 ± 0.0133	0.3682 ± 0.0180	**0.5380** ± 0.0303
2.0	3.0	**0.1556** ± 0.0122	**0.2978** ± 0.0214	0.6638 ± 0.0532	0.1596 ± 0.0624	0.2981 ± 0.0076	0.7844 ± 0.1193	0.1716 ± 0.0042	0.3580 ± 0.1760	0.6188 ± 0.0296	0.2015 ± 0.0240	0.3840 ± 0.0148	**0.5463** ± 0.0120
5.0	8.0	0.1748 ± 0.0528	0.3396 ± 0.0490	0.7750 ± 0.1013	0.1574 ± 0.0319	0.2992 ± 0.0141	0.6874 ± 0.0356	**0.1371** ± 0.0332	**0.2781** ± 0.0018	0.6611 ± 0.0195	0.1482 ± 0.0010	0.2942 ± 0.0202	**0.5880** ± 0.1194

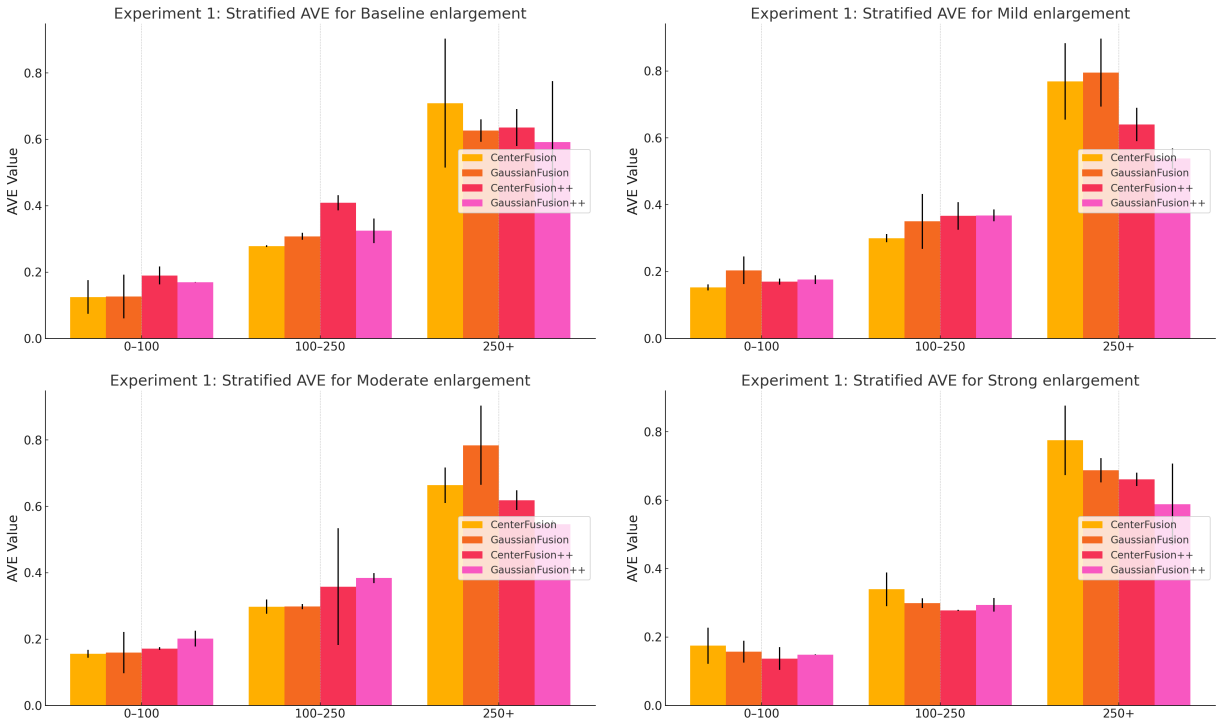


Figure A.2.11: Stratified AVE results for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++ under different static frustum enlargement configurations (Baseline, Mild, Moderate, Strong). Each plot shows the AVE grouped by depth ranges: 0–100 m, 100–250 m, and beyond 250 m.

Click **here** to jump back to Experiment 1.

A.2.2 Experiment 2

Predictions using dynamic frustum enlargement

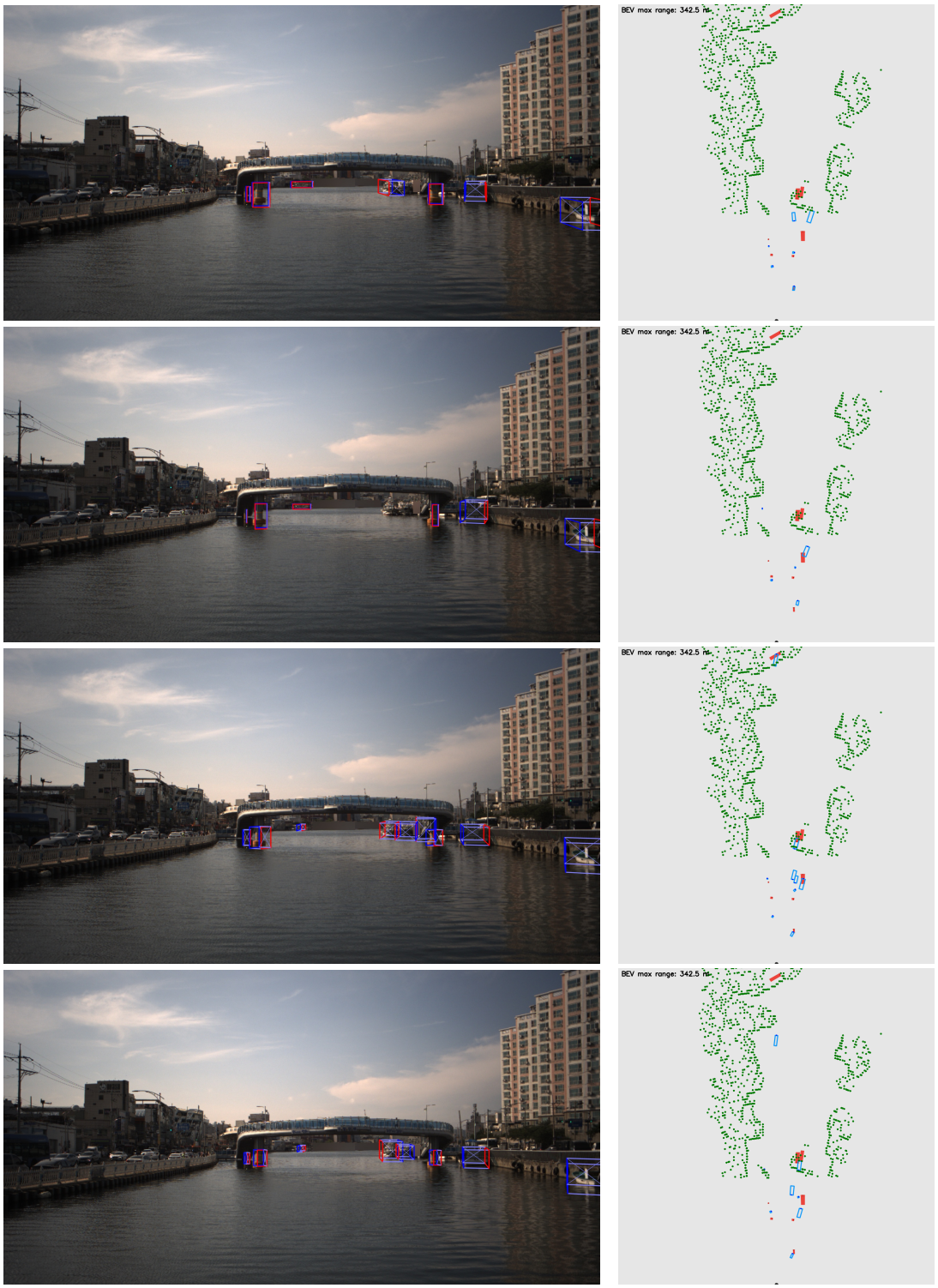


Figure A.2.12: Qualitative detection results on sample 1 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under dynamic frustum enlargement using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

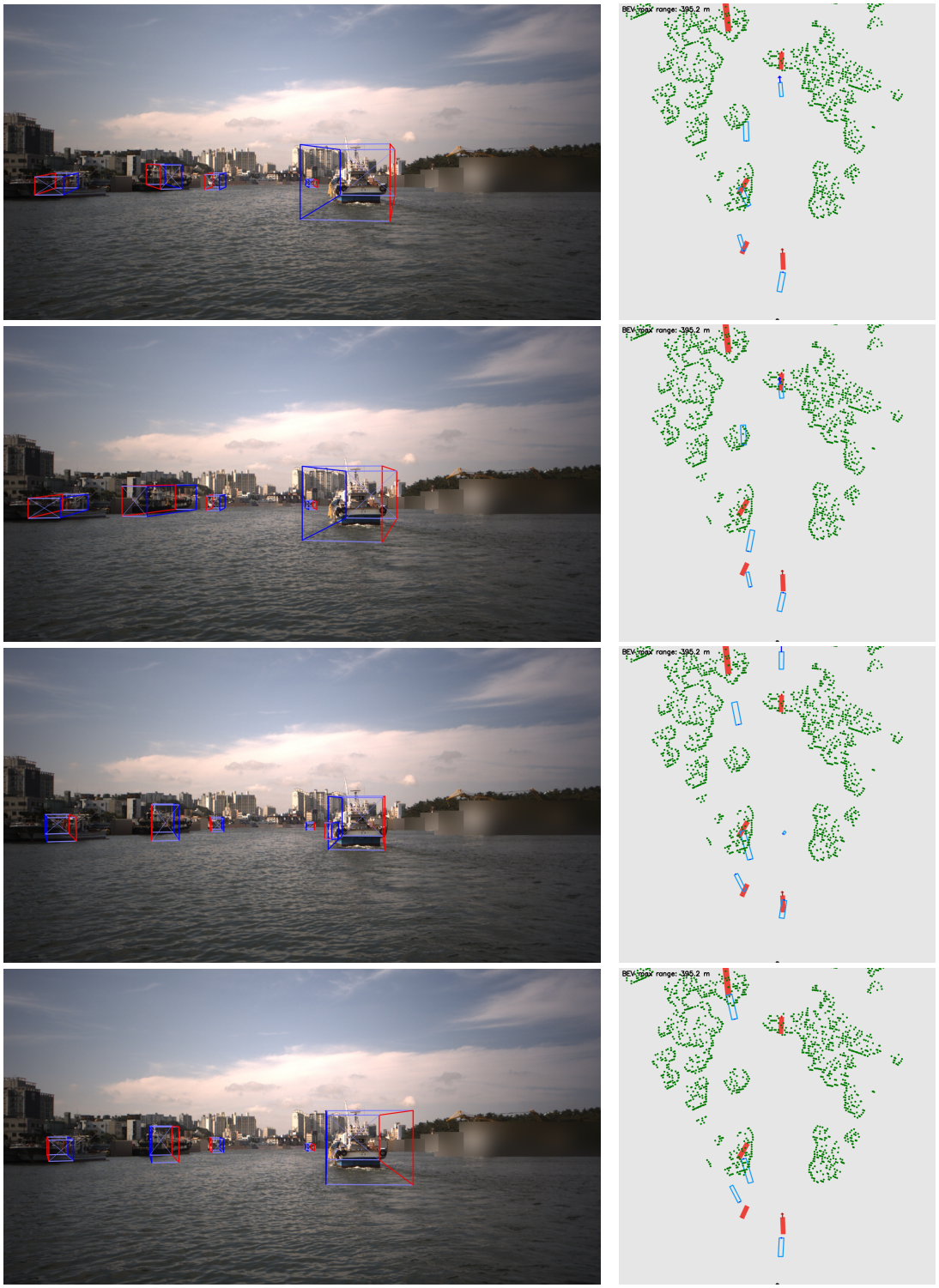


Figure A.2.13: Qualitative detection results on sample 2 for CenterFusion (first row), GaussianFusion (second row), CenterFusion++ (third row), and GaussianFusion++ (last row) under dynamic frustum enlargement using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

Click **here** to jump back to Table 5.5.

Histograms of aggregated results

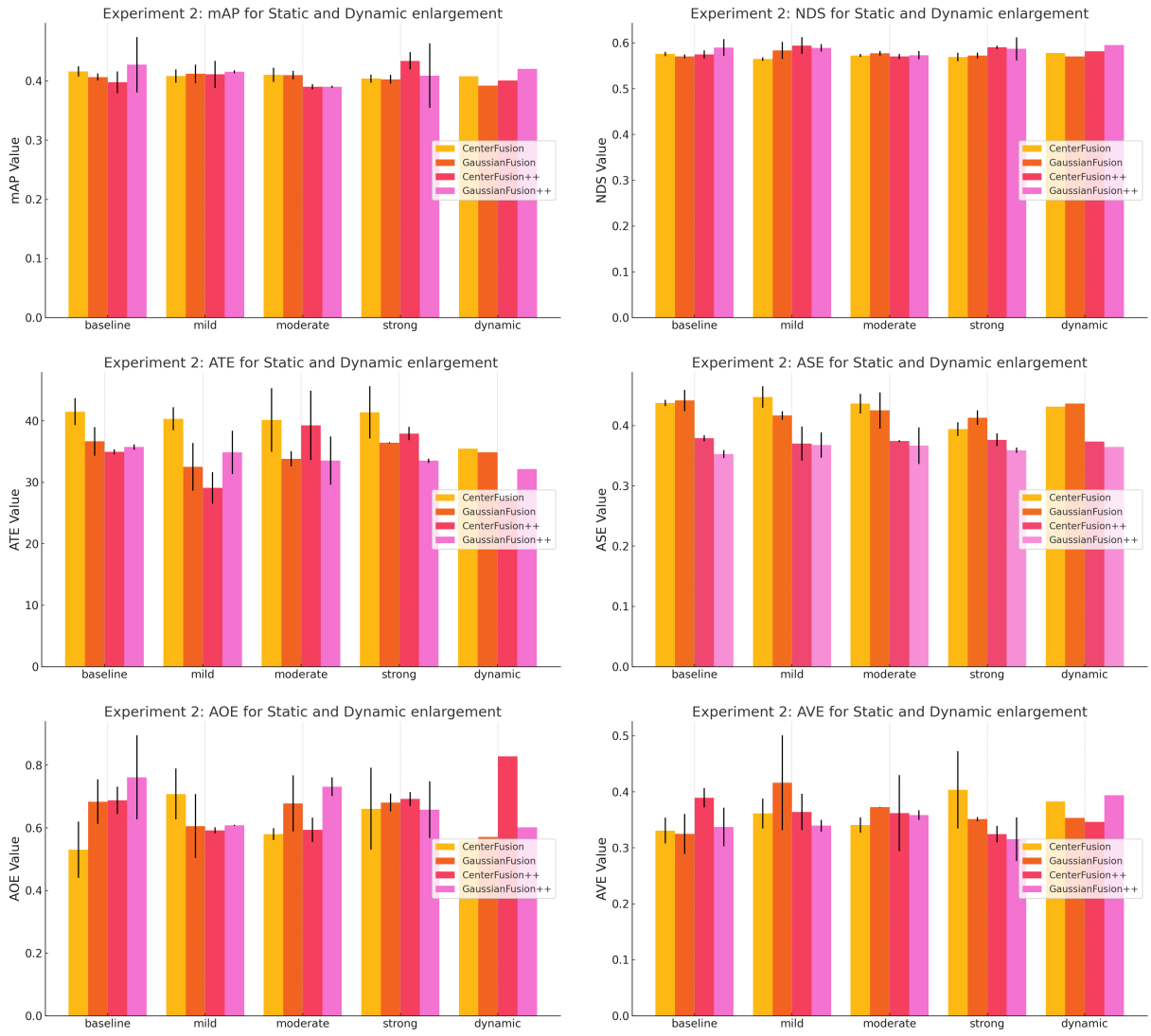


Figure A.2.14: Aggregated performance metrics (mAP, NDS, ATE, ASE, AOE, AVE) under dynamic frustum enlargement using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. Each subplot shows the trend for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++.

Click **here** to jump back to Table 5.5.

Histograms of stratified results

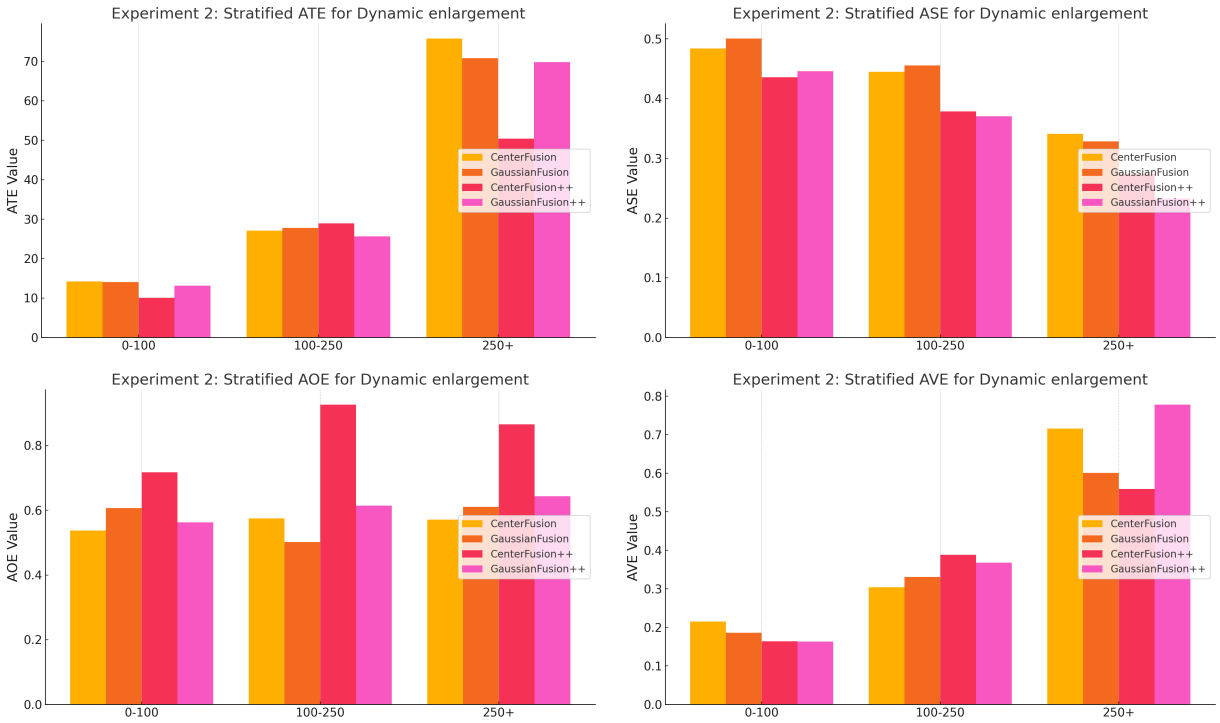


Figure A.2.15: Stratified metrics (ATE, ASE, AOE, AVE) for CenterFusion, GaussianFusion, CenterFusion++, and GaussianFusion++ under dynamic frustum enlargement using $\epsilon_{base} = 0.5$, $\delta_{base} = 1$, and $k = 5 \times 10^{-6}$. Each plot shows results grouped by depth ranges: 0–100 m, 100–250 m, and beyond 250 m.

Click **here** to jump back to Table 5.6.

A.2.3 Experiment 3

Predictions using different uncertainty scaling (α)

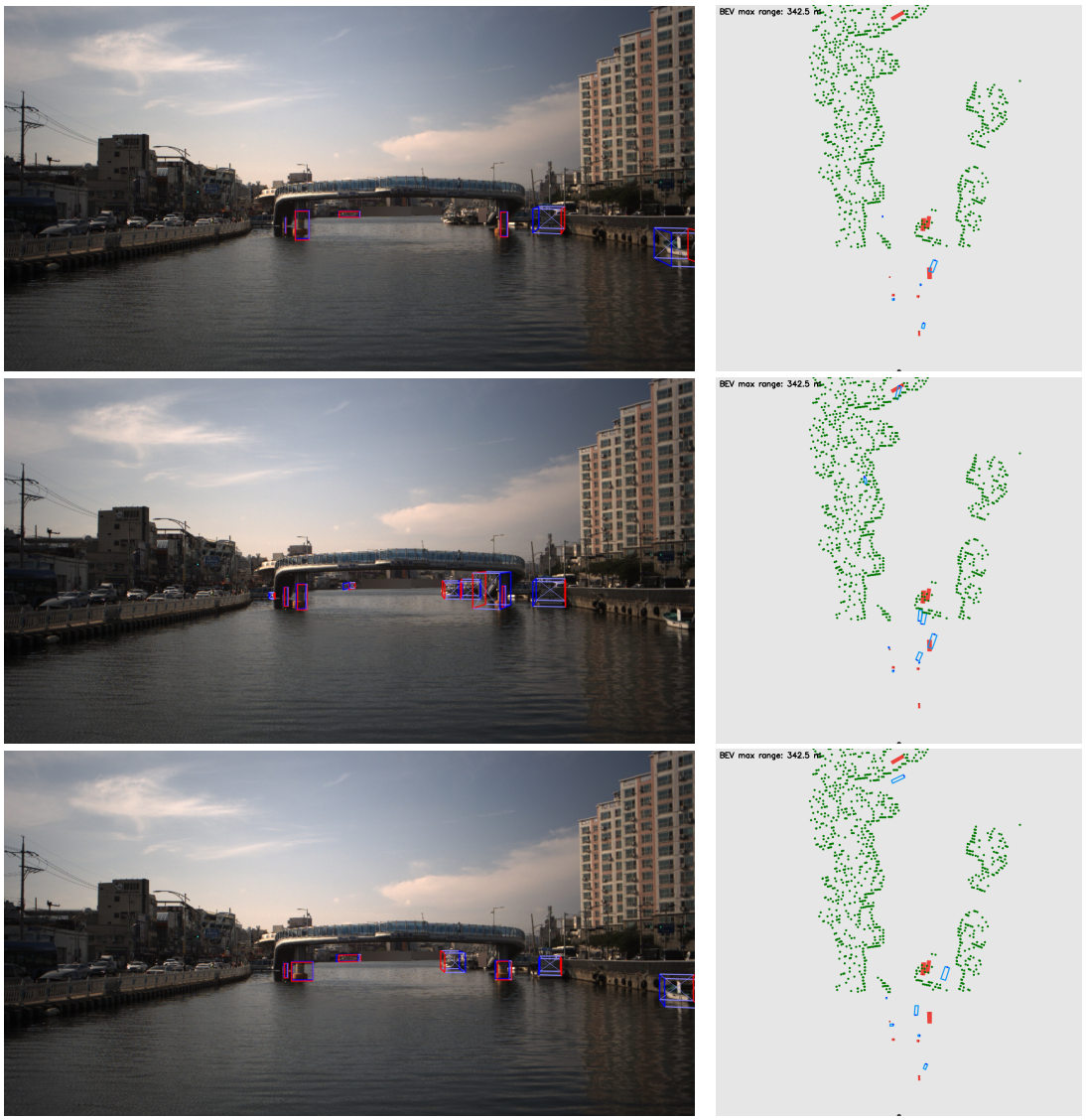


Figure A.2.16: Qualitative detection results on sample 1 for GaussianFusion under different uncertainty scale factors: $\alpha = 0.1$ (top row), $\alpha = 0.05$ (middle row), and $\alpha = 0.2$ (bottom row). Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

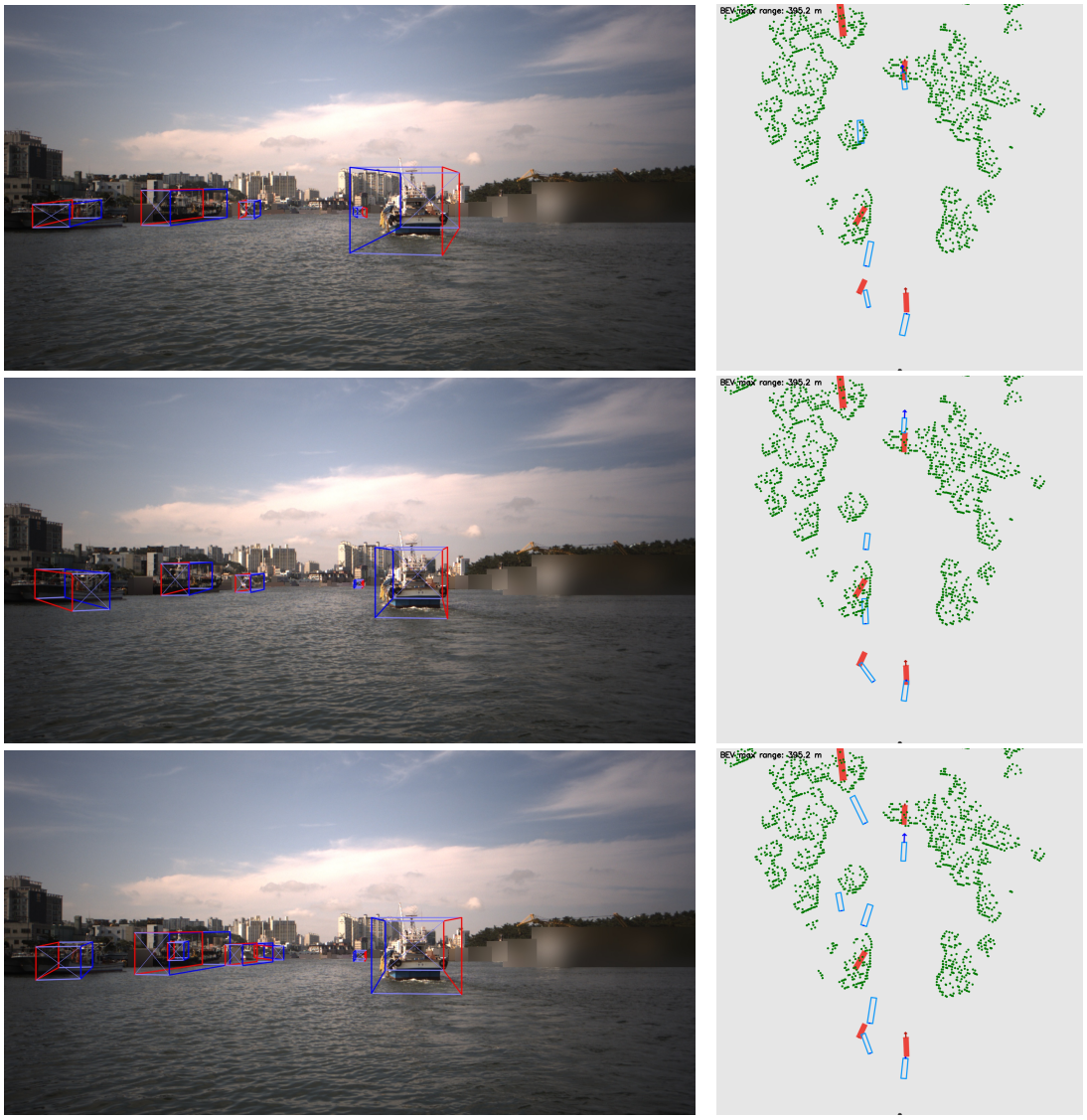


Figure A.2.17: Qualitative detection results on sample 2 for GaussianFusion under different uncertainty scale factors: $\alpha = 0.1$ (top row), $\alpha = 0.05$ (middle row), and $\alpha = 0.2$ (bottom row). Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

Click **here** to jump back to Table 5.11.

Histograms of aggregated results

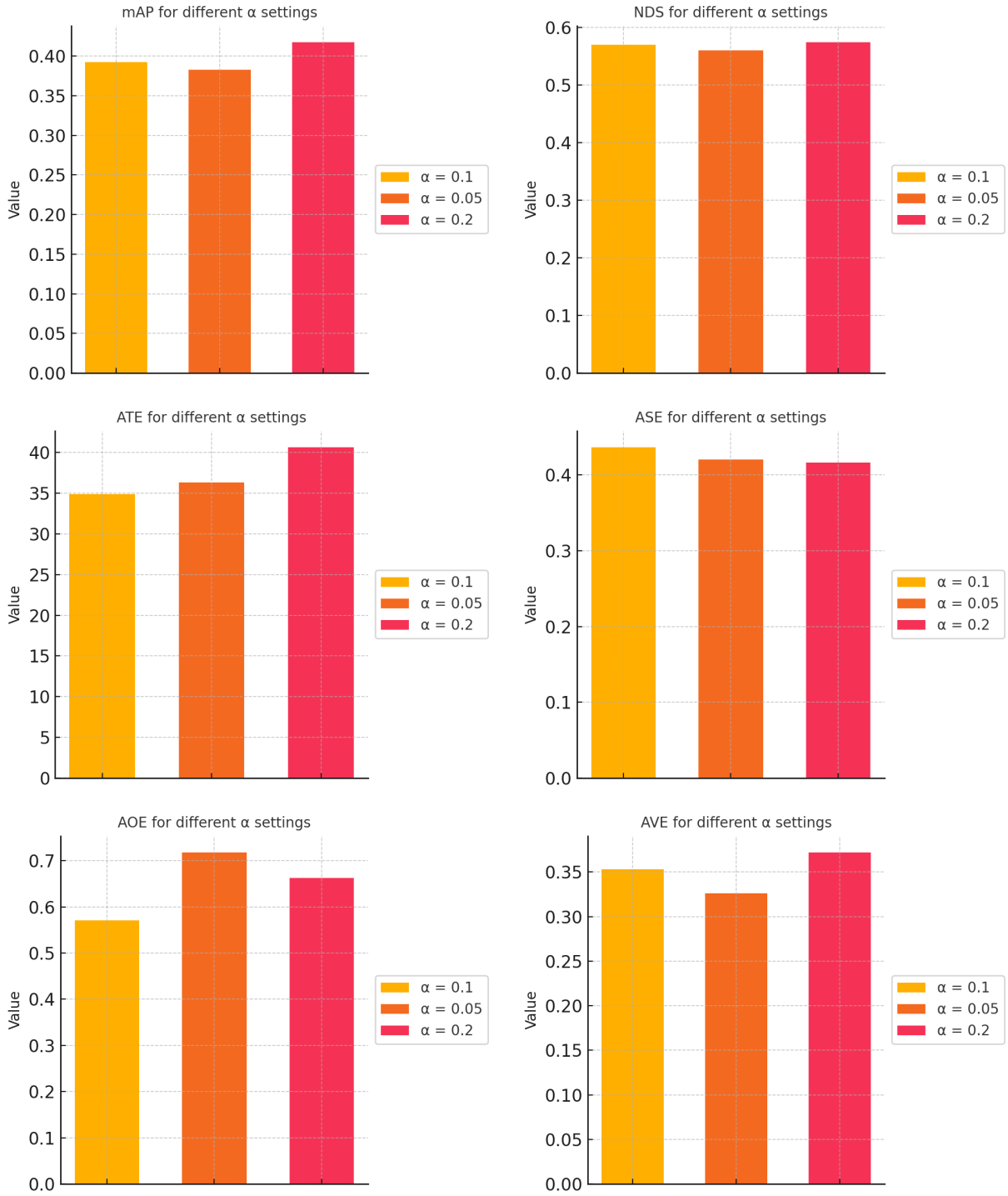


Figure A.2.18: Aggregated performance metrics (mAP, NDS, ATE, ASE, AOE, AVE) for GaussianFusion under different uncertainty scale factors (α) in Experiment 3. Each subplot visualises the trend for $\alpha = 0.05$, $\alpha = 0.1$, and $\alpha = 0.2$.

Click **here** to jump back to Table 5.11.

Histograms of stratified results

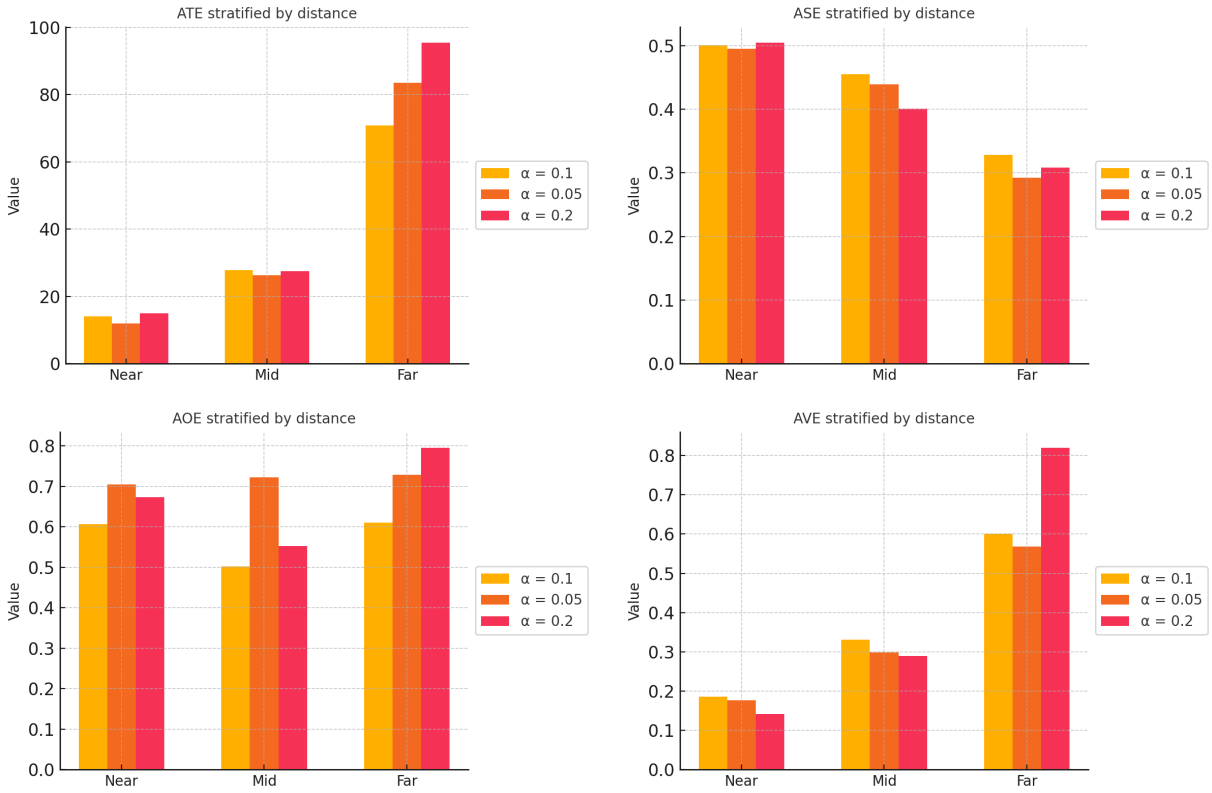


Figure A.2.19: Stratified metrics (ATE, ASE, AOE, AVE) for GaussianFusion under different uncertainty scale factors (α) in Experiment 3. Each plot shows results grouped by depth ranges: 0–100 m, 100–250 m, and beyond 250 m.

Click **here** to jump back to Table 5.12.

A.2.4 Experiment 4

Predictions for RGPNet variants

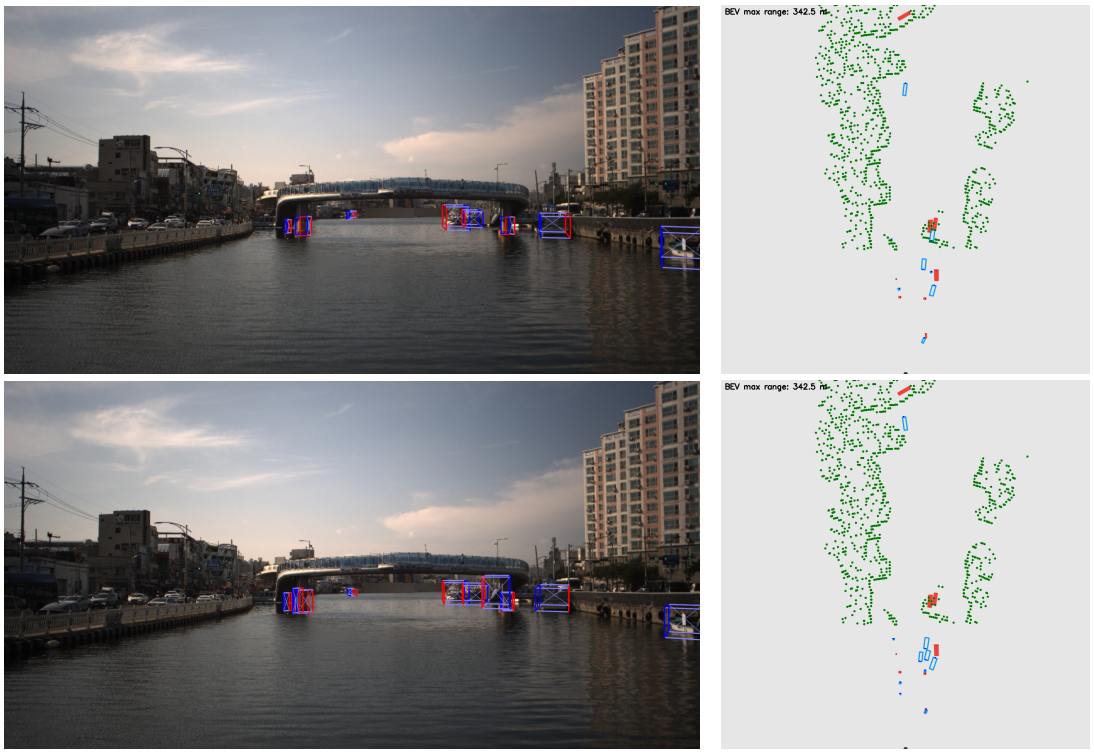


Figure A.2.20: Qualitative detection results on sample 1 for RGPNet-A (top row) and RGPNet-B (bottom row) under dynamic frustum enlargement. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

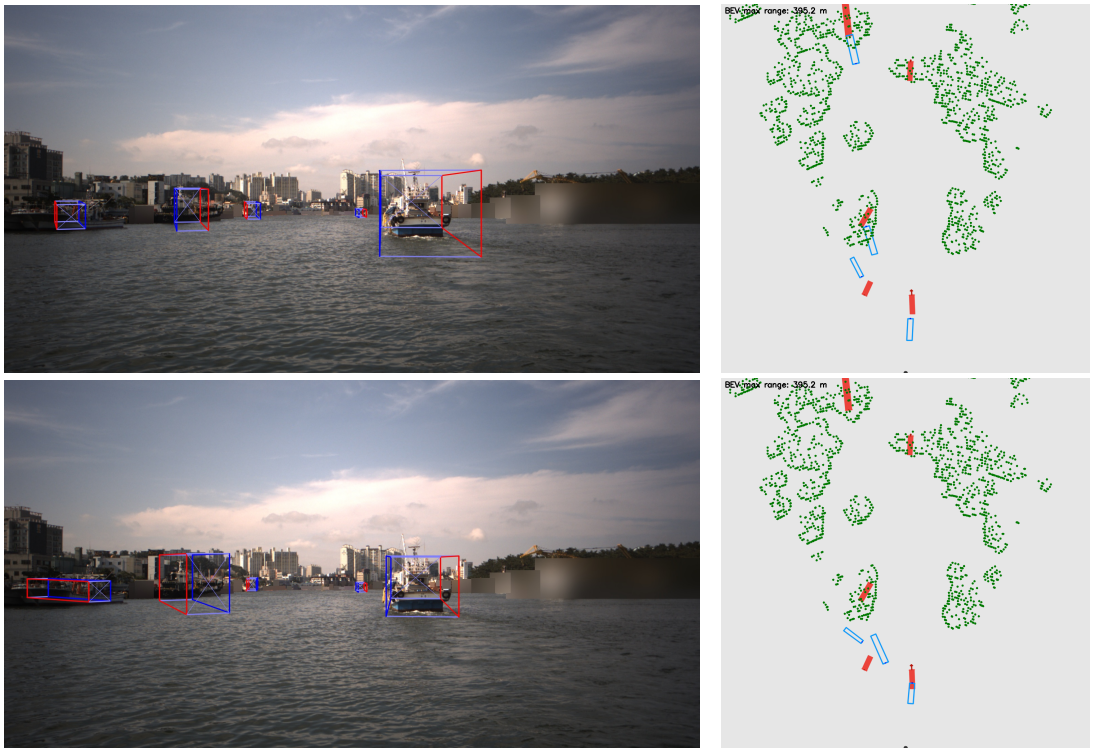


Figure A.2.21: Qualitative detection results on sample 2 for RGPNet-A (top row) and RGPNet-B (bottom row) under dynamic frustum enlargement. Each row shows the camera view (left) and the corresponding BEV (right). Refer to Figure A.2.1 for the ground truth annotations.

Click **here** to jump back to Table 5.13.

Histograms of aggregated results

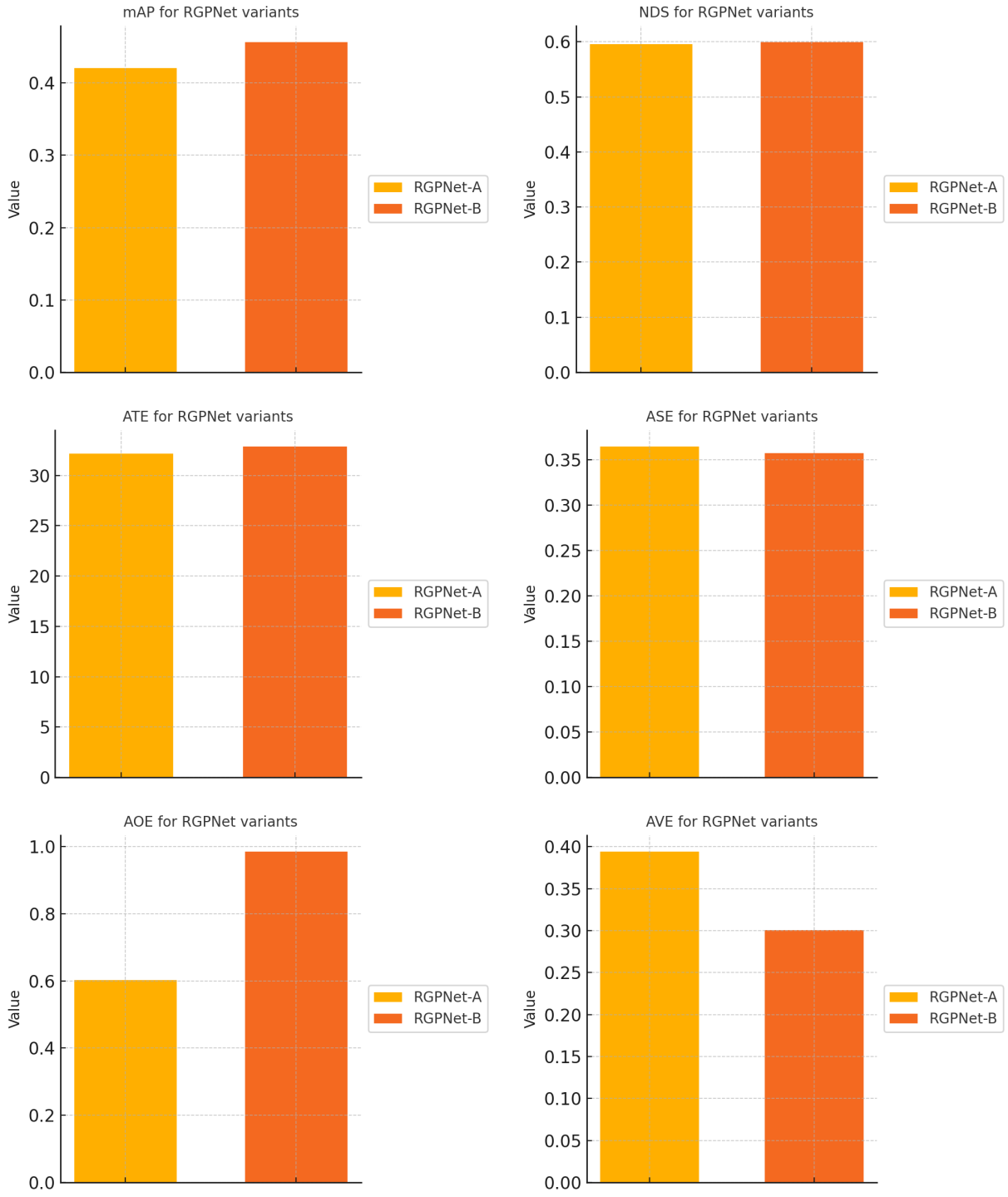


Figure A.2.22: Aggregated performance metrics (mAP, NDS, ATE, ASE, AOE, AVE) for RGPNet-A and RGPNet-B under dynamic frustum enlargement in Experiment 4.

Click **here** to jump back to Table 5.13.

Histograms of stratified results

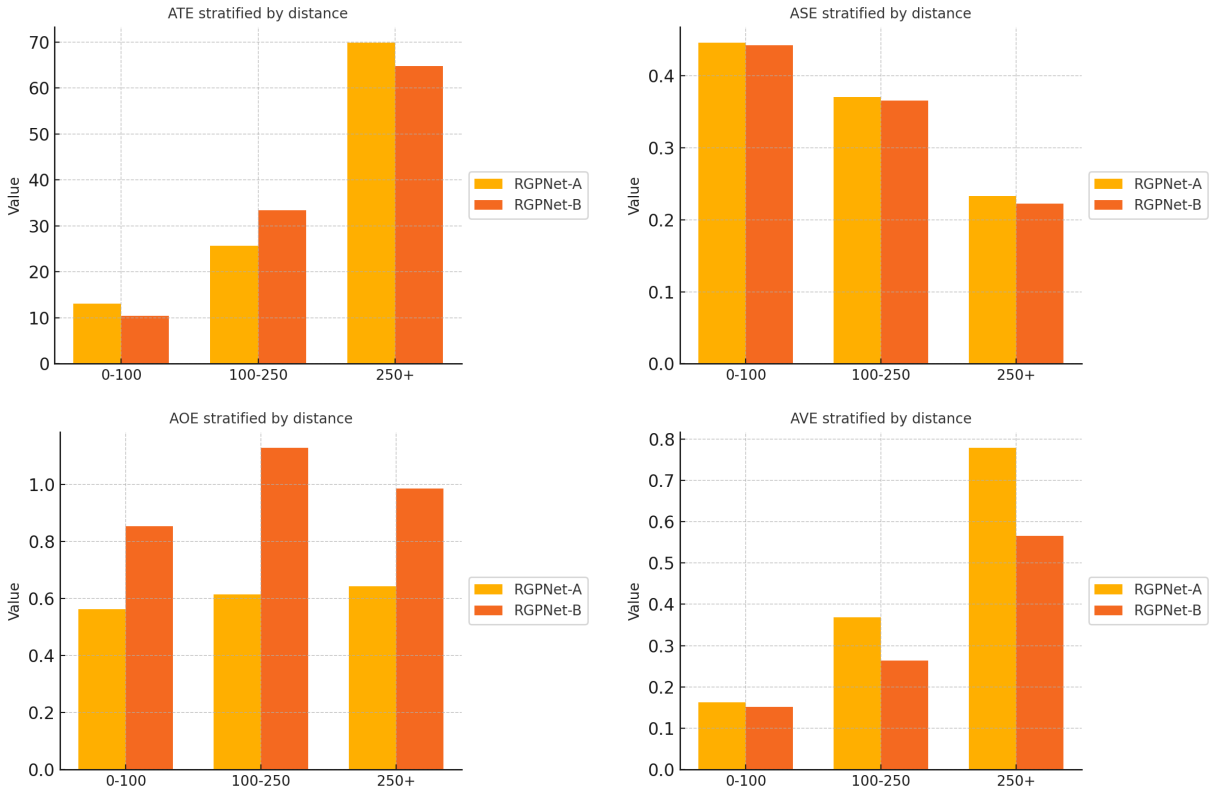


Figure A.2.23: Stratified metrics (ATE, ASE, AOE, AVE) for RGPNet-A and RGPNet-B under dynamic frustum enlargement in Experiment 4. Each plot shows results grouped by depth ranges: 0–100 m, 100–250 m, and beyond 250 m.

Click **here** to jump back to Table 5.14.

Bibliography

[1] Y. Ma, Z. Wang, H. Yang, and L. Yang, "Artificial intelligence applications in the development of autonomous vehicles: A survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 315–329, 2020.

[2] L. Kretschmann, H.-C. Burmeister, and C. Jahn, "Analyzing the economic benefit of unmanned autonomous ships: An exploratory cost-comparison between an autonomous and a conventional bulk carrier," *Research in transportation business & management*, vol. 25, pp. 76–86, 2017.

[3] D. Liu, "Autonomous vessel technology, safety, and ocean impacts," in *The Future of Ocean Governance and Capacity Development*. Brill Nijhoff, 2019, pp. 490–494.

[4] S. Y. Nof, "Automation: What it means to us around the world," in *Springer handbook of automation*. Springer, 2009, pp. 13–52.

[5] J. M. Anderson, K. Nidhi, K. D. Stanley, P. Sorensen, C. Samaras, and O. A. Oluwatola, *Autonomous vehicle technology: A guide for policymakers*. Rand Corporation, 2014.

[6] Statista, "Global automotive industry revenue 2017-2022," 2024, accessed: 25-09-2024. [Online]. Available: https://www.statista.com/statistics/574151/global-automotive-industry-revenue/

[7] Fortune Business Insights, "Unmanned surface vehicle (usv) market size, share, growth — 2024-2032," 2025, last updated: April 7, 2025. Accessed: April 28, 2025. [Online]. Available: https://www.fortunebusinessinsights.com/unmanned-surface-vehicle-usv-s-market-102526

[8] E. M. S. A. (EMSA), "Annual overview of marine casualties and incidents 2024," European Maritime Safety Agency, Tech. Rep., June 2024.

[9] T. Miao, E. El Amam, P. Slaets, and D. Pissoort, "An improved real-time collision-avoidance algorithm based on hybrid a* in a multi-object-encountering scenario for autonomous surface vessels," *Ocean Engineering*, vol. 255, p. 111406, 2022.

[10] T. Miao, "Efficient autonomous sailing system in real-life scenarios-advanced multi-object tracking and collision avoidance algorithms," 2023.

[11] L. Su, Y. Chen, H. Song, and W. Li, "A survey of maritime vision datasets," *Multimedia Tools and Applications*, vol. 82, no. 19, pp. 28 873–28 893, 2023.

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The international journal of robotics research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[13] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.

[14] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[15] FURUNO Electric Co., Ltd., "Radar basics — furuno technology," https://www.furuno.com/en/technology/radar/basic/, n.d., accessed: 2025-05-04.

[16] D. Chung, J. Kim, C. Lee, and J. Kim, "Pohang canal dataset: A multimodal maritime dataset for autonomous navigation in restricted waters," https://sites.google.com/view/pohang-canal-dataset/home, 2023, accessed: 2025-05-04.

[17] H. Ji, P. Liang, and E. Cheng, "Enhancing 3d object detection with 2d detection-guided query anchors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 178–21 187.

[18] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 2010, pp. 242–264.

[19] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.

[20] Y. Luo, Z. Huang, and Z. Bao, "Adapting depth distribution for 3d object detection with a two-stage training paradigm," in *International Conference on Intelligent Computing*. Springer, 2024, pp. 62–73.

[21] X. Liang, Y. Wu, J. Han, H. Xu, C. Xu, and X. Liang, "Effective adaptation in multi-task co-training for unified autonomous driving," *Advances in Neural Information Processing Systems*, vol. 35, pp. 19 645–19 658, 2022.

[22] J. Z. Sasiadek, "Sensor fusion," *Annual Reviews in Control*, vol. 26, no. 2, pp. 203–228, 2002.

[23] P. Kolar, P. Benavidez, and M. Jamshidi, "Survey of datafusion techniques for laser and vision based sensor integration for autonomous navigation," *Sensors*, vol. 20, no. 8, p. 2180, 2020.

[24] M. Schmitt and X. X. Zhu, "Data fusion and remote sensing: An ever-growing relationship," *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 4, pp. 6–23, 2016.

[25] K. Liu, *Dual-sensor approaches for real-time robust hand gesture recognition*. The University of Texas at Dallas, 2015.

[26] R. F. Brena, A. A. Aguileta, L. A. Trejo, E. Molino-Minero-Re, and O. Mayora, "Choosing the best sensor fusion method: A machine-learning approach," *Sensors*, vol. 20, no. 8, p. 2350, 2020.

[27] S. Yao, R. Guan, X. Huang, Z. Li, X. Sha, Y. Yue, E. G. Lim, H. Seo, K. L. Man, X. Zhu *et al.*, "Radar-camera fusion for object detection and semantic segmentation in autonomous driving: A comprehensive review," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 2094–2128, 2023.

[28] S. R. Stahlschmidt, B. Ulfenborg, and J. Synnergren, "Multimodal deep learning for biomedical data fusion: a review," *Briefings in bioinformatics*, vol. 23, no. 2, p. bbab569, 2022.

[29] K. Liu, Y. Li, N. Xu, and P. Natarajan, "Learn to combine modalities in multimodal deep learning," *arXiv preprint arXiv:1805.11730*, 2018.

[30] S. Chang, Y. Zhang, F. Zhang, X. Zhao, S. Huang, Z. Feng, and Z. Wei, "Spatial attention fusion for obstacle detection using mmwave radar and vision sensor," *Sensors*, vol. 20, no. 4, p. 956, 2020.

[31] R. Yadav, A. Vierling, and K. Berns, "Radar+ rgb attentive fusion for robust object detection in autonomous vehicles," *arXiv preprint arXiv:2008.13642*, 2020.

[32] Y. Kim, J. Shin, S. Kim, I.-J. Lee, J. W. Choi, and D. Kum, "Crn: Camera radar net for accurate, robust, efficient 3d perception," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 17 615–17 626.

[33] Z. Lin, Z. Liu, Z. Xia, X. Wang, Y. Wang, S. Qi, Y. Dong, N. Dong, L. Zhang, and C. Zhu, "Rcbevdet: Radar-camera fusion in bird's eye view for 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14 928–14 937.

[34] Y. Li, A. W. Yu, T. Meng, B. Caine, J. Ngiam, D. Peng, J. Shen, Y. Lu, D. Zhou, Q. V. Le *et al.*, "Deepfusion: Lidar-camera deep fusion for multi-modal 3d object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 17 182–17 191.

[35] X. Bai, Z. Hu, X. Zhu, Q. Huang, Y. Chen, H. Fu, and C.-L. Tai, "Transfusion: Robust lidar-camera fusion for 3d object detection with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 1090–1099.

[36] Z. Yang, J. Chen, Z. Miao, W. Li, X. Zhu, and L. Zhang, "Deepinteraction: 3d object detection via modality interaction," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1992–2005, 2022.

[37] R. Nabati and H. Qi, "Centerfusion: Center-based radar and camera fusion for 3d object detection," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 1527–1536.

[38] J. Brandes and J. Kübel, "Camera-radar sensor fusion using deep learning. a frustum proposal-based 3d object detection network for multi stage fusion in autonomous driving," 2022.

[39] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6569–6578.

[40] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*. Springer, 2014, pp. 740–755.

[41] J. Yang, Z. Bian, Y. Zhao, W. Lu, and X. Gao, "Staged-learning: Assessing the quality of screen content images from distortion information," *IEEE Signal Processing Letters*, vol. 28, pp. 1480–1484, 2021.

[42] D. Chung, J. Kim, C. Lee, and J. Kim, "Pohang canal dataset: A multimodal maritime dataset for autonomous navigation in restricted waters," *The International Journal of Robotics Research*, vol. 42, no. 12, pp. 1104–1114, 2023.

[43] D. L. Mills, "Network time protocol (ntp)," Tech. Rep., 1985.

[44] Chrony Project. (2025) Chrony. Accessed: 2025-03-21. [Online]. Available: https://chrony-project.org/

[45] UnixTimestamp.com, "Unix timestamp converter," n.d., accessed: 2025-05-07. [Online]. Available: https://www.unixtimestamp.com/

[46] M. Kaess, A. Ranganathan, and F. Dellaert, "isam: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.

[47] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[48] Pennsylvania State University, "4.2 - bivariate normal distribution," 2025, accessed: 2025-05-08. [Online]. Available: https://online.stat.psu.edu/stat505/lesson/4/4.2

[49] B. Ge, F. Najar, and N. Bouguila, "Data-weighted multivariate generalized gaussian mixture model: Application to point cloud robust registration," *Journal of Imaging*, vol. 9, no. 9, p. 179, 2023.

[50] M. Singh, M. K. Mandal, and A. Basu, "Gaussian and laplacian of gaussian weighting functions for robust feature based tracking," *Pattern Recognition Letters*, vol. 26, no. 13, pp. 1995–2005, 2005.

[51] M.-C. Popescu, V. E. Balas, L. Perescu-Popescu, and N. Mastorakis, "Multilayer perceptron and neural networks," *WSEAS Transactions on Circuits and Systems*, vol. 8, no. 7, pp. 579–588, 2009.

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

[53] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[55] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. pmlr, 2015, pp. 448–456.

[56] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[57] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, D. Keysers, J. Uszkoreit, M. Lucic, and A. Dosovitskiy, "Mlp-mixer: An all-mlp architecture for vision," in *Advances in Neural Information Processing Systems*, 2021. [Online]. Available: https://arxiv.org/abs/2105.01601

[58] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *NeurIPS*, 2017.

[59] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

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

[61] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for neural networks for image processing," *arXiv preprint arXiv:1511.08861*, 2015.

[62] G. J. McLachlan, "Mahalanobis distance," *Resonance*, vol. 4, no. 6, pp. 20–26, 1999.

[63] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Advances in neural information processing systems*, vol. 30, 2017.

[64] U. Rajapaksha, F. Sohel, H. Laga, D. Diepeveen, and M. Bennamoun, "Deep learning-based depth estimation methods from monocular image and videos: A comprehensive survey," *ACM Computing Surveys*, vol. 56, no. 12, pp. 1–51, 2024.

[65] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[66] L. Prechelt, "Early stopping-but when?" in *Neural Networks: Tricks of the trade*. Springer, 2002, pp. 55–69.

[67] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, pp. 303–338, 2010.

[68] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Available: https://www.deeplearningbook.org/