Advancing Maritime Perception: PV-RCNN for 3D LiDAR Object Detection

Herberto Werner[1,2,*], Cosmin Delea[1], Nico Zantopp[1], Ching Nok Au[1], Tim Tiedemann[2]

[1]Fraunhofer Center for Maritime Logistics and Services (CML), Hamburg, Germany
[2]Hamburg University of Applied Sciences (HAW), Hamburg, Germany
[*]Corresponding author. Email: herberto.werner@cml.fraunhofer.de

Abstract.
The perception module of an Uncrewed Surface Vehicle (USV) is the basis for performing missions in real-world environments like harbours, rivers, seas, and water canals. Compared to manned systems, the advanced technology of USVs offers a wide range of applications in maritime infrastructures, such as inspections, transportation, and surveillance tasks or multi-agent combinations with other robotic systems like Autonomous Underwater Vehicles (AUVs). Nonetheless, maritime environments present certain challenges, e.g., disturbances like winds and waves with dynamic obstacles like ship traffic, buoys, and other moving objects. An USV needs to perceive and avoid these obstacles to ensure robust navigation and safe operation. Light Detection and Ranging (LiDAR)-based perception algorithms are widely used in robotic obstacle detection and avoidance due to their reliable and robust depth data measurements. The paper explores a deep-learning-based approach using PV-RCNN. The model takes raw LiDAR points as input and outputs classified boat detections. A LiDAR dataset was created with manually labeled boats and extended through augmentation techniques, such as rotation, scaling, and GT-sampling. The model was trained and evaluated with different hyperparameter settings, with the goal of improving autonomous navigation of USVs in maritime environments. Experiments showed that applying a moderate rotation of 10° during augmentation achieved optimal results in Recall and detection performance at IoU=0.5/0.7, leading to improved generalization and robustness compared to both no augmentation and higher rotation degrees. In contrast, longer training durations and extensive augmentation with high rotation angles led to low performance values. Altogether, the experiments demonstrate that adding more diverse data and optimizing dataset configuration, model architecture, and hyperparameters (e.g., higher batch size, more training epochs, additional object classes) can improve detection performance. These improvements contribute to more robust and reliable boat detection.

1 Introduction
In recent years, there has been a surge in the development and deployment of waterborne uncrewed vehicles, with the USVs being used in a wide area of application. USVs present numerous advantages in some key applications, including maritime surveillance, environmental missions, offshore inspection, deep-sea resource exploration, autonomous tasks in economic harbours, and military purposes [1]. While there are some commercially available USVs, there are even more experimental platforms, these being driven by the need for reducing the overall size, increasing the payload, or enhancing the level of autonomy.
Due to the nature of most applications, USVs usually operate in dynamic maritime environments with certain challenges. They must navigate through confined spaces (e.g., harbours, rivers, and narrow channels) with moving objects, like other vessels or buoys, and endure harsh weather conditions and changing environmental disturbances (e.g., currents, waves, and winds). These factors affect the

performance of sensors and navigation systems, requiring adaptive capabilities to detect and avoid obstacles for safe navigation to a specific location.

The ability of these systems to detect obstacles, recognize and track targets, and map environments is, in turn, heavily dependent on navigation sensors (Global Navigation and Satellite System (GNSS), Inertia Measuring Unit (IMU), Doppler Velocity Logger (DVL)) and environmental perception sensors (Camera, Radar, Sonar, Laser, LiDAR)[2]. For the robustness of the navigation solution, there have been several solutions proposed, most of them using the Guidance Navigation and Control (GNC) framework, theorized by [3]. While there is a certain degree of robustness in the navigation system that allows coping with some of the environmental disturbances, the perception system is still critical for obtaining an accurate understanding of the environment.

Motivated by the SeaClear2.0 Project and its primary goal of employing marine robotics to clean ports and coastal areas by removing surface and seafloor debris, a medium-sized USV acts as a shuttle tender, having to autonomously navigate to a specific location, where another USVs is located and perform a stern-to-stern docking. The latter is assumed to be laden with a piece of litter weighing up to 250 kg, that can be discarded from a position 1.5 m behind the USV and at least 0.5 m above the water surface. Provided that the shuttle tender USV has safely navigated through the harbour and reached its docking position, the collected waste is lowered using a grapple onto the USV's litter bay. A safe manoeuvre is considered one that avoids colliding with static and dynamic objects that are to be found on, or in the close proximity of the path followed by the USV.

To achieve this level of navigational safety, the USV is dependent on the perception system's ability to sense its environment. Light Detection and Ranging (LiDAR) sensors are a robust key technology for obtaining accurate depth data and have proven their efficacy numerous times in previous research for perception in autonomous driving and maritime environments.

A review of current literature reveals that traditional unsupervised learning methods segment and cluster obstacles [4, 5]. A common approach involves unsupervised methods (partition-based, hierarchical, and density-based clustering) to segment and classify obstacles in the environment. However, clustering methods still have limitations and can be unstable. These studies also show that the usage of deep learning-based supervised methods has been increasing. However, most methods are image-based, and there is a lack of maritime point-based deep learning methods [6]. Another challenge is missing benchmarks, training and validation datasets for LiDAR-based 2D/3D object detection and tracking in maritime scenarios.

In recent years, maritime perception has increasingly depended on image-based object detections to fulfill maritime use cases. Maritime perception as a research topic has not been explored as thoroughly as autonomous driving on the road. Two-stage methods (e.g. Faster R-CNN, R-FCN, Cascade R-CNN, Mask R-CNN), as well as single-stage methods (e.g. the single-shot detector (SSD), RetinaNet, You Only Look Once (YOLO), and EfficientDet) are widely used in maritime environments [5]. Recent studies use multi-sensor fusion solutions and various YOLO versions to detect maritime objects (e.g. buoys, ships) or dynamic obstacles [7, 8, 9]. In contrast, the field of object detectors in the maritime sector has not been explored as extensively as in the automotive sector, especially with only point clouds as input data. Benchmarks like the KITTI dataset or other point cloud datasets aren't established and widespread in the maritime context. In turn, the KITTI dataset itself is also used as a benchmark for maritime tasks [10, 11]. Recording and labeling data from different maritime scenes with various weather conditions is time-consuming and a complex task. Therefore, studies often use generated synthetic–or a mix of synthetic and recorded data–for training and validation. The study in [12] used simulated data generated with Gazebo along with real-world data as training and validation sets. In this study, an adapted PointPillar architecture with a modified SSD detection head was used to detect 2D objects in harbour areas. Their perception method–which consists of three functional modules: object detection, multi-object tracking, and static environment mapping–enabled the mapping of a static environment and simultaneous tracking of floating objects. [6] introduces a LiDAR point cloud dataset of ships, which addresses the lack of LiDAR point cloud data with a focus on ships. The dataset combines authentic recorded data collected using a multibeam photon-counting LiDAR sensor applied for ocean exploration, with simulated data for object detection and object tracking tasks within marine scenarios. The simulated data was generated using Blensor, featuring rainy and foggy weather simulation along with data augmentation. The dataset is used to train SECOND, PointPillars, and PointRCNN and consists of 10,571 frames for training, 1,200 frames for validation, and 1,500 frames for testing. Among the four distinct classes–Cargo ship, tour boat, Engineering ship, and Speedboat–the detection object is the cargo ship. The evaluation shows that the recorded and simulated data in all model training sets achieve higher precision around 50-60% for all difficulty levels: easy, moderate, and hard.

Identifying the challenges of deep learning-based maritime perception leads to the central research question of this paper: To what extent can data augmentation techniques compensate for a limited-scale, real-world maritime LiDAR dataset when training a 3D object detection model for boat detection using PV-RCNN, known for high accuracy in the 3D detection benchmarks?

To answer this question and address the identified challenges, this work presents the following contributions:

- We present a robust LiDAR-based object detection pipeline for dynamic maritime conditions using the Point–Voxel Feature Set Abstraction for 3D Object Detection (PV-RCNN) with raw LiDAR points as input, optimized for detecting boats in novel scenes.
- We evaluate and analyze the impact of data augmentation and hyperparameter tuning on the model's performance, providing insights of the challenges and opportunities of implementing a 3D LiDAR-based object detection onboard a USVs for achieving autonomous navigation within a harbour environment.
- We create and validate a custom maritime point cloud dataset, adressing the lack of maritime point cloud data.

2 Background

2.1 Point cloud-based 3D object detection

Point cloud-based 3D object detection pipeline starts with data acquisition from the LiDAR sensor. The raw points are often transformed from polar to cartesian coordinates. The next pre-processing steps are ground removal and noise filtering to remove outlier points. The following step is data segmentation. The outputs are distinct grouped points to form candidate objects for classification and detection. Finally, objects are detected and recognised. Traditional object detection algorithms, like Support Vector Machines (SVM), rely on feature extraction, where the feature vector consists mostly of common quantities, like real cluster dimensions, the number of points, elements of the covariance matrix, elements of the inertia tensor, central moments, or reflectance intensity histogram [13]. In contrast to modern deep learning techniques, the largely manual feature extraction has been replaced by integrating neural networks. This approach improves accuracy and robustness. However, they also have a high computational demand and are memory complex. Deep learning approaches have been applied to point cloud processing inspired by image-based detections [4]. The following Figure 1 shows an overview of the pipeline for 3D object detection using a custom dataset.

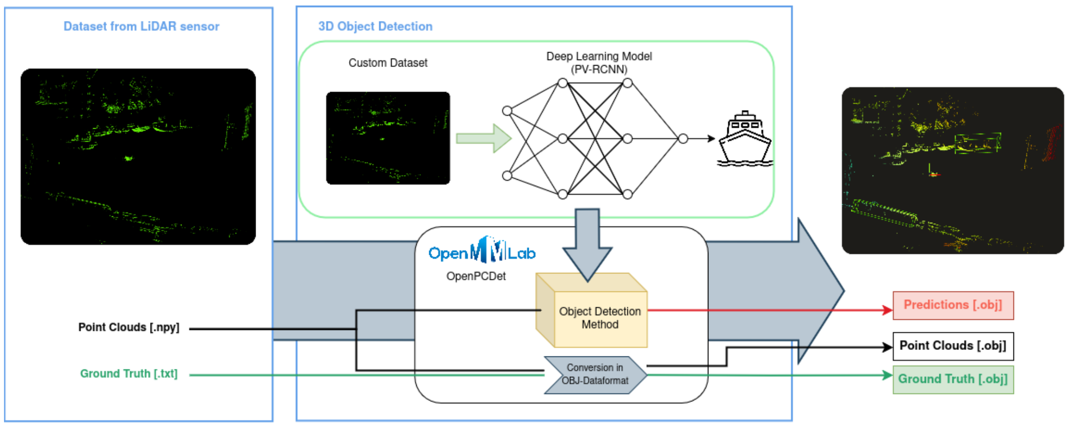

Figure 1: Workflow overview of point cloud-based object detection using custom dataset

A point cloud dataset from a LiDAR sensor, recorded in different maritime environments, is utilised as custom input and consists of 3D raw points. The 3D point cloud data are stored as files encoded in NumPy array format. Corresponding ground truth data describe the object location and size within the point cloud. A deep learning model (PV-RCNN), implemented within the 3D object detection method box, processes the custom dataset. The method transforms the input data into output predictions, represented by 3D bounding boxes around detected objects within the point clouds. A popular tool in this area is the open-source toolbox OpenPCDet [14]. It supports multiple LiDAR-based perception models and a variety of state-of-the-art 3D object detection architectures. Its advantages include adaptability and effectiveness, due to its modular design, YAML-based model and dataset configurations, as well as various interchangeable backbones, dense-heads, etc. Deep learning-based 3D object detection methods share an architecture composed of three main components: *Data Representation, Feature Extraction, and Detection Network.*

2.2 Structure of pointcloud-based 3D object detection methods
The following figure 2 shows a general structure and the three main components.

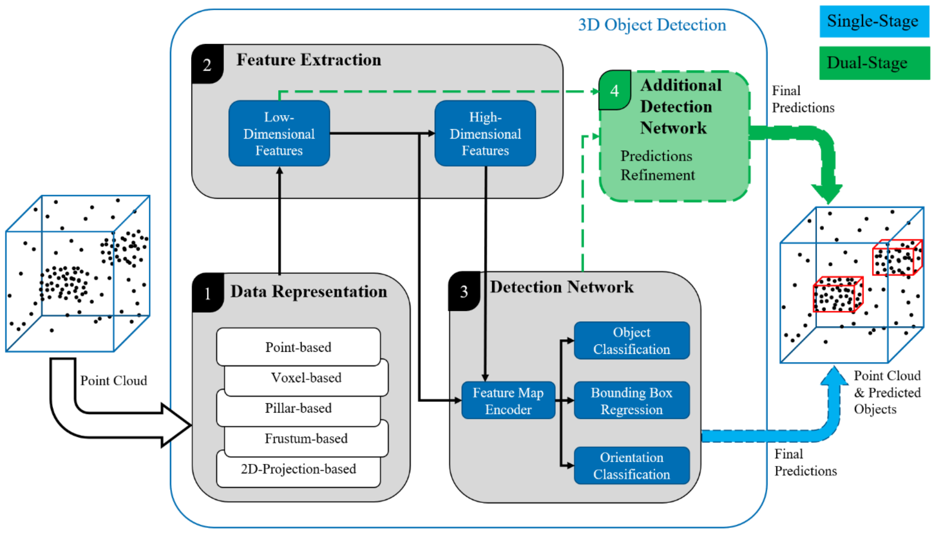

Figure 2: Main components of a 3D Object Detection model according to [15]

2.2.1 Data representation
Deep learning models efficiently process the input data by transforming the raw data into a structured format. Several types of data representations and variations for processing LiDAR data are shown in figure 2. *Point-based* methods apply point-wise feature learning on raw, unordered point clouds (e.g., PointNet, PointNet++). In *Voxel-based* methods, point clouds are divided into small 3D voxels and aggregate point features within each voxel. 2D or 3D Convolutional Neural Networks (CNNs) are applied in these methods (e.g., VoxelNet). In *Pillar-based* methods (e.g., PointPillars), point clouds are divided into vertical column pillars on a uniform 2D grid (in the x-y plane). In *2D-Projection-based* methods, point clouds are converted into 2D feature maps using view projection (e.g., bird's-eye view or front view), which are processed by CNNs (e.g., MV3D [16]). *Frustums-based* methods convert point clouds into frustums using an image-to-point cloud projection approach. [17, 18, 19]

2.2.2 Feature Extraction
The feature extraction stage generates low- and high-dimensional features from the previous data representation to create a *feature map*. These feature maps are multi-dimensional tensors capturing local and global geometric structures and semantic information. They enable the model to understand spatial relationships, object shapes, and patterns. Depending on the type of data representation, different feature extraction methods can be applied, including *point-wise, segment-wise, object-by-object, CNNs.* [17, 18, 19]

2.2.3 Detection Network
The detection network determines the object classes, bounding box regression, and orientation based on the encoded feature map. Two detector architectures: *single stage and dual stage* are used to generate a 3D box with a class score for the object of interest. The single stage performs detection in one pass and is faster. A dual stage detector firstly proposes candidate regions and then refines predictions. It is a slower and more accurate network. Additionally, the *region proposal-based detector* uses a Region of Interest (RoI) and *anchorless detector* suitable for hidden or truncated objects. [17, 18, 19]

2.3 PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection
PV-RCNN [20] is a two-stage 3D object detection framework that integrates the advantages from point-based and voxel-based feature learning methods to achieve high accuracy. Most detectors have disadvantages when using either a grid-based method or a point-based method. Grid-based methods transform irregular points to regular representations (3D voxels or 2D bird-view maps). Then, these

representations are processed by 3D or 2D CNNs to learn point features. It is computationally efficient, but the method loses information with fine-grained localization. Point-based methods, like PointNet, are accurate but computationally heavy. PV-RCNN's two-step strategy (voxel-to-keypoint encoding and keypoint-to-grid pooling) combines grid-based processing with the fine-grained localization of point-based methods for efficiency and balance. With the benchmark of KITTI [21] and Waymo Open datasets, it's achieving top ranks in 3D detection benchmarks. Due to its benchmark scores and memory efficiency, the following experiments show the maritime application by detecting boats.

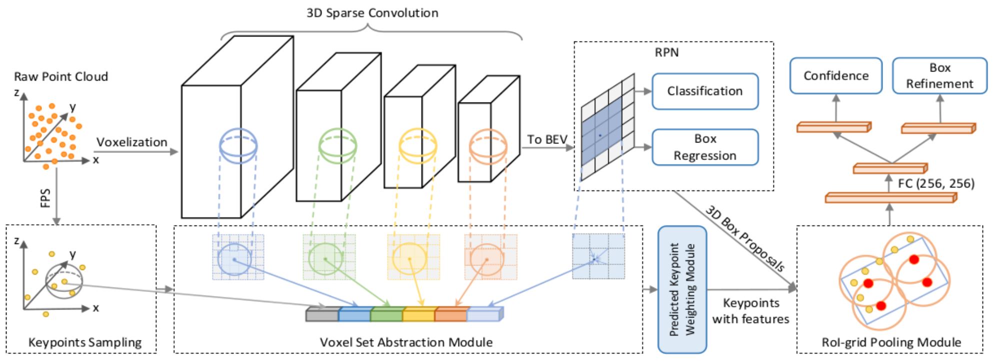

Figure 3: Architecture Overview of PV-RCNN [20]

The following section describes the main components of the architecture:

1. **Input: Raw point cloud** - The input of the model are raw 3D points. The points are preprocessed by converting them into a sparse 3D grid as voxels.
2. **3D Sparse Convolution** - Then, the voxels are processed by 3D sparse convolutions. They extract multi-scale features while the spatial structure are preserved.
3. **Bird's Eye View (BEV)** - The 3D voxel features are projected into a 2D top-down view. It simplifies detections with only X-Y spatial relationships for proposal generation. Furthermore, it reduces computational complexity.
4. **Proposal Generation** - The input of the Region Proposal Network (RPN) is the BEV map to generate 3D anchor box proposals for objects of interest.
5. **Keypoint Feature Aggregation** - With *Keypoints Sampling*, a sparse set of keypoints is sampled with the Farthest Point Sampling algorithm, which preserves geometric details of the raw point cloud. The *Voxel Set Abstraction (VSA) Module* aggregates the multi-scale voxel features from the 3D CNN backbone into the sampled keypoints. Local voxel features, raw point features, and BEV features are fused by the VSA. The *Predicted Keypoint Weighting Module* processes the keypoints with feature. These features are rich and multi-scale contextual information about the scene.
6. **RoI-Grid Pooling and Refinement** - The *RoI-Grid Pooling Module* samples uniformly for each proposal (RoI) within the 3D bounding box. Therefore, the set abstraction puts the features from keypoints into the grid points, which captures local and global context. Then, the *FC (256,256)* fully connected layers processes the RoI-grid features for refining object predictions.
7. **Output** - Finally, a predicted object class with an estimated detection confidence score is emitted. The *Box Refinement* adjust coordinates (center, size and orientation) for a precise 3D bounding box regression.

2.4 Performance evaluation
Selecting a performance evaluation metric is significant for understanding and improving 3D point-cloud-based object detection for USVs. In 3D object detection, each predicted bounding box is represented by the vector $(x, y, z, l, w, h, \theta, class)$ with the center coordinates (x, y, z), dimensions (length l, width w, height h), 1D orientation (θ), and object label [22].
Evaluation of 3D detectors needs to evaluate both classification and localization accuracy. The OpenPCDet toolkit supports mapping custom datasets into the standard KITTI evaluation protocol,

which is widely recognized as a benchmark for 3D object detection [23]. In this work, KITTI-format data is used to evaluate boat detection performance using two common metrics—Average Precision (AP) at specific Intersection over Union (IoU) threshold and Recall—under the established KITTI evaluation rules. Table 1 presents a confusion matrix, which reflects the model's detection accuracy against ground truth. The matrix has four elements: True Positive (TP) - correct detected objects, False Positive (FP) - incorrect (non-existent) objects, True Negative (TN) - correct undetected (non-existent) objects, and False Negative (FN) - missed detections, which are present [22]. Precision and Recall can be computed through the confusion matrix. Precision is the proportion of true positive detections among all detections. Recall is the proportion of true positive detections among all existing ground truth objects for a given class. Mean Average Precision (mAP) summarizes the performance across multiple classes and measures the model's Precision-Recall performance with various IoU thresholds. Precision and recall inherently have a conflict of objectives. A large number of candidate bounding boxes increases recall since more true objects are detected. On the other hand, it decreases the precision, as more false positives are introduced.

Table 1: Confusion Matrix for 3D Detection according to [15]

	Object detected	**No Object detected**
Object Present	True Positive (TP)	False Negative (FN)
No Object Present	False Positive (FP)	True Negative (TN)

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

The 3D IoU metric calculates the overlap of the prediction and the ground truth. It measures the object position accuracy and is used to classify objects, as TP and FP for overall performance evaluation [22]. To classify predictions, IoU thresholds are used. The detection is considered a TP if the 3D IoU exceeds a given IoU threshold; otherwise, it is a FP.

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}} \tag{3}$$

OpenPCDet evaluates two-stage detectors using Recall at IoU thresholds of 0.30, 0.50, 0.70, which measure the fraction of ground-truth objects correctly proposed (ROI stage) and refined (Region-based Convolutional Neural Network (RCNN) stage). Table 2 summarizes these metrics used in our experiments, as presented in section 5. Furthermore, Average Precision (AP) can be further evaluated by interpolating the precision-recall curve at evenly spaced Recall levels. *AP_R40* samples the curve at 40 evenly distributed recall values. For every detection confidence threshold, a precision is computed and the precision-recall curve is built. For each Recall value in the range $0.025, 0.05, \ldots, 0.975, 1.0$, the maximum precision at or above that recall is considered. The mean of these interpolated precision values is the *AP_R40* score. Compared to the 11-point AP metric, *AP_R40*'s samples finer and captures the precision-recall curve's shape and granularity.

Table 2: Description of Evaluation Metrics

Metric	**Description**
ROI Recall	Recall rate after the Region of Interest proposal stage.
RCNN Recall	Recall rate after the refinement (RCNN) stage.
bbox AP	Average Precision of the predicted 3D bounding boxes at a given IoU threshold.
BEV AP	Average Precision measured in the bird's-eye view projection at a given IoU.
3D AP	Average Precision in full 3D space, evaluating both localization and size.
AOS (Orientation)	Average Orientation Similarity; measures the accuracy of estimated object orientations alongside detection.

3 Methodology

3.1 High-level Architecture

As depicted in Figure 4, the GNC architecture for USVs (called NJORDIQ Stack), which is a modular and flexible framework designed to support various USV applications. The stack includes components for perception, navigation, control, and communication, allowing for efficient integration of different sensors and algorithms. The perception module processes sensor data (e.g. LiDAR) to detect and track objects in the environment. The navigation module uses the processed data to plan safe paths and avoid obstacles, while the control module executes the planned manoeuvers. The communication module ensures that the USV can exchange information with other systems and operators.

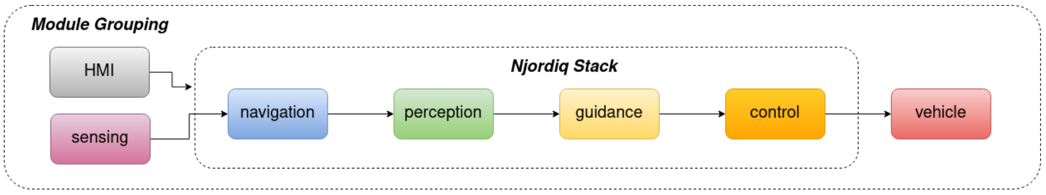

Figure 4: Block view of the main modules of the Njordiq stack. The perception pipeline is integrated within the general GNC architecture, but has a dedicate module that augments the global planner (inside the guidance module)

3.2 Perception Deployment on the USV

The perception module's main responsibility within the NJORDIQ-stack is detecting and localising static and dynamic obstacles in close proximity to the USV, as well as predicting their future positions. The output of the perception module is a set of detected objects with their positions, velocities, and predicted trajectories, which are used by the guidance module's local path planner for replanning the USV's itinerary in real-time, keeping it collision-free. The perception module is implemented on the USV using the Robot Operating System 2 (ROS2) framework and consists of several components, including data acquisition, data processing, and data visualization. The data acquisition component collects sensor data from the environment, while the data processing component applies algorithms to detect and track objects. The integration of the perception module into the high-level control system, particularly the guidance module, poses several technical challenges. A primary concern is ensuring real-time performance, as the perception module must process LiDAR data, detect objects, and predict both static (e.g.,length or beam) and dynamic parameters (e.g., course or velocity) within a time frame that allows the guidance module to replan the USV's path or trajectory effectively. Given that most pleasure crafts are not mandated to transmit their position via Automatic Identification System (AIS), there is often no ground truth observation available to validate the perception module's performance. Consequently, the accuracy of detections directly influences the set of feasible manoeuvers. Late or imprecise detections may result in suboptimal path planning, potentially leading to conservative actions (e.g., station-keeping until the vessel is no longer in proximity) or even collisions with environmental objects. To mitigate such inaccuracies, a predefined close-proximity safety zone around the USV is established, within which any detected objects are treated as higher collision risks, prompting conservative manoeuvers. Additionally, the applicability of COLREG[1] rules to uncrewed vehicles remains an open question. A more conservative approach, such as always yielding to other vessels, may be preferable and would significantly influence the manoeuvring decisions made by the guidance module. While the integration of the perception module into the overall system architecture is beyond the scope of this study, it will be addressed in future research.

3.3 Datasets and Data Annotation pipeline

The OpenPCDet custom dataset requires a customized data structure in order to train PV-RCNN using raw points. The following figure 5 shows the data preparation and annotation pipeline:

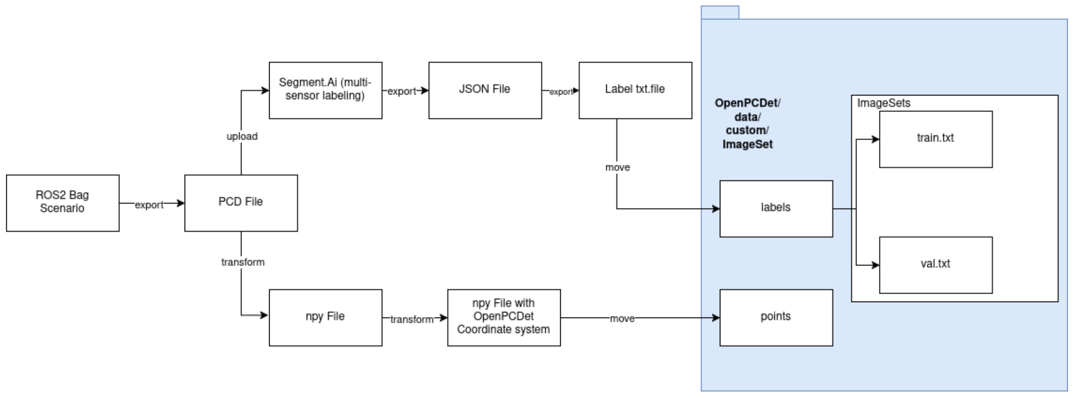

Figure 5: Data annotation and preparation pipeline.

The Point Cloud Data (PCD) files, which are frame shots of the recorded data (see 4.1), were exported separately from the ROS2 rosbags and then merged. The raw data (PCD files) for this study was collected from two distinct maritime environments, Kiel Bay and the Lotsekanal in Hamburg (see Section 4.1). Visible boat objects with the size of dimensions: *x: 37-38m*, *y: 8-10m*, *z: 12-20m*, as shown in figure 6 in the frames were extracted from these recordings to form a base dataset. To ensure the model learns from realistic, imperfect data, the selection criteria focused on capturing a variety of perspectives and distances to the target objects (boats) without excluding frames due to sensor noise or partial closures. The data is weighted towards the Lotsekanal recordings, comprising 121 frames from Hamburg and 260 frames from Kiel, as the Bay environment in Kiel offered more variable perspectives and a larger acquisition area. Furthermore, the selection of boat types and sizes for annotation was data-driven. Medium-sized boats were prioritized and were the most frequently and clearly captured objects in our rosbag recordings. This approach allowed us to build a baseline dataset centering on the most common objects in the operational domain and offering an opportunity to extend more boat sizes and classes.
The PCD files were uploaded and annotated using Segment.ai, a multi-sensor data annotation platform. Each PCD file with boat objects was manually annotated (see figure 6).The base dataset of 361 manually annotated PCD files was used for data preparation. This sample size was chosen to establish a foundational dataset, acknowledging the time-intensive nature of 3D point cloud annotation. Segment.ai offered to export a JSON file with the coordinates of the bounding boxes. The exported JSON data was used to create annotation text files containing the boat object and the corresponding bounding box coordinates. The PCD data was then converted into NumPy files. Since OpenPCDet specifies a certain coordinate orientation, the NumPy files were adjusted with a coordinate transformation and uploaded to a folder. For training and validation, the LiDAR data was split in a ratio of 80 % for training and 20 % for testing.
The base data of annotated 361 PCD samples is further used for data augmentation. The following figure 6 shows various labeled boats.

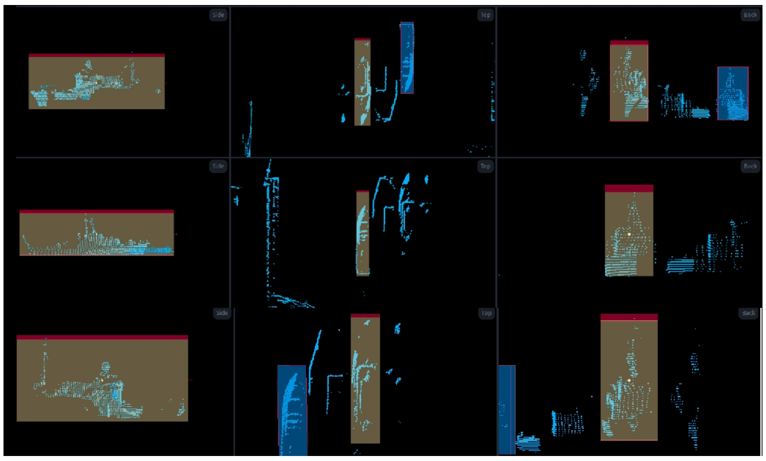

Figure 6: The figure show the various perspectives in the segment.ai labeling process. It's visible that some points are missing to form a boat hull. The noise is created depending, where the USV's Ouster LiDAR is pointed and the position of the boat itself. Other conditions, like weather and environment also affect the sensor measurements.

3.4 Data Augmentation

Annotating objects in 3D is time-consuming and labor-intensive, and manual labeling is subjectively biased. In order to generalize the training and to enhance the diversity and quantity of data, further data augmentations are applied to the base dataset. Data augmentation is commonly used to generate more data for 3D object detection, including basic transformations, such as rotations, scaling, cropping, mirror flipping, etc. [6]. A rotation of x, y, z with the degrees *10, 30* was applied on each sample's axis. The following figure 7 shows an example transformation.

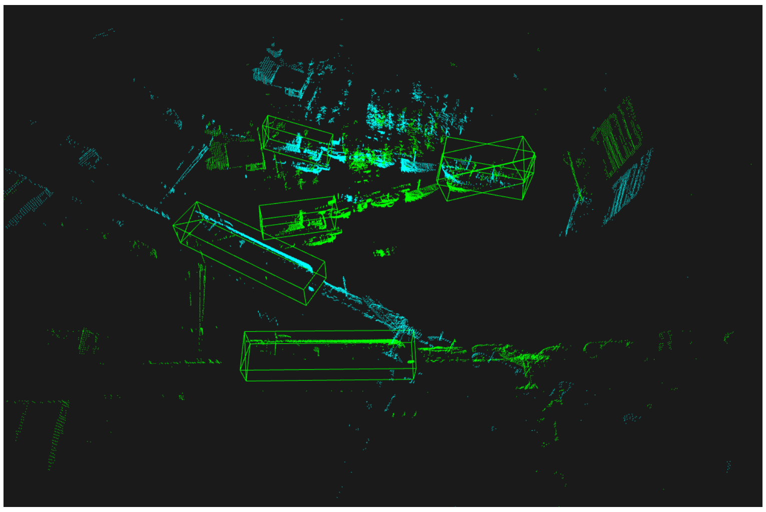

Figure 7: Rotation around z-axis with 10° with corresponding annotated 3D bounding boxes.

Additionally, OpenPCDet also offers data augmentation methods: random world scaling and GT-sampling. GT-sampling is an advanced augmentation technique, which randomly adds some new ground-truth objects from other scenes to the current training scenes. It's used to simulate objects in various environments. Both methods were applied on the base and augmented datasets. With the merged data for each degree and accumulating each total sample set to the next one, the total sum adds up as shown in the Table 3.

Table 3: Overview of Dataset Splits and Sample Counts

Dataset Split	**Number of Samples**
Base	361
Dataset_10_degree	1.448
Dataset_30_degree	2.534

4 Experiment Setup

The following section describes the experiment setup, including the dataset acquisition site, test vessel, sensor setup, and the PV-RCNN model configurations.

4.1 Dataset Acquisition Site

One data source was located in the Kiel Bay near WTD 71 and was recorded between June and July 2023. The terrain was around a bay environment, sheltered with an open shallow-water expanse, and limited wave/tidal action. The primary disturbances were around $8\,\mathrm{m\,s^{-1}}$ to $10\,\mathrm{m\,s^{-1}}$ (15 kn to 19 kn) with mostly westerly wind directions. Another dataset site in this study was collected between March and May 2025 at Lotsekanal (Pilot's Channel) in Hamburg, Germany, as shown in Figure 8. Located in the southern part of the port city, the channel is shielded by a lock that minimizes the influence of the Elbe river's currents. Consequently, tidal effects and currents are significantly reduced. During the sea trials, the primary disturbances were caused by wind, which ranged from $6\,\mathrm{m\,s^{-1}}$ to $8\,\mathrm{m\,s^{-1}}$ (11 kn to 16 kn) and varied in direction from N, NE, to E, depending on the day.

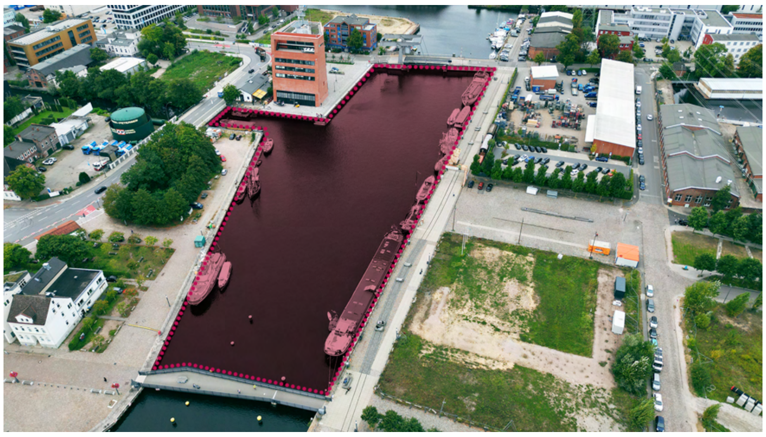

Figure 8: The Lotsekanal (Pilot's Channel), situated in the Port of Hamburg, has been chosen for conducting the sea trials. The designed area, marked with dark-red colour, is approximately 200 m long and 65 m wide in its broadest point

4.2 Test Vessel

The LiDAR Ouster1 (OS1) is utilized for the dataset creation and experiments. The sensor was mounted on the SeaML:SeaDragon USV (see Figure 9), which is a 5.5 m x 2.1 m catamaran, having 350 kg payload and capable of developing up to 6 kn speed. The USV actuation system consists of two steerable pod drives with 6 kW power each. The USV is designed for autonomous navigation tasks, including visual perception and LiDAR-based environment mapping.
This LiDAR sensor is a mid-range and high-resolution imaging LiDAR. It's designed for all-weather environments and use in industrial automation. The sensor offers a maximum range of 170 m (80 % Lambertian reflectivity, horizontal resolution 1024 pixels @ 10 Hz mode) according to the manufacturer's data sheet [24]. Table 4 shows the settings for the data recordings and dataset creation.

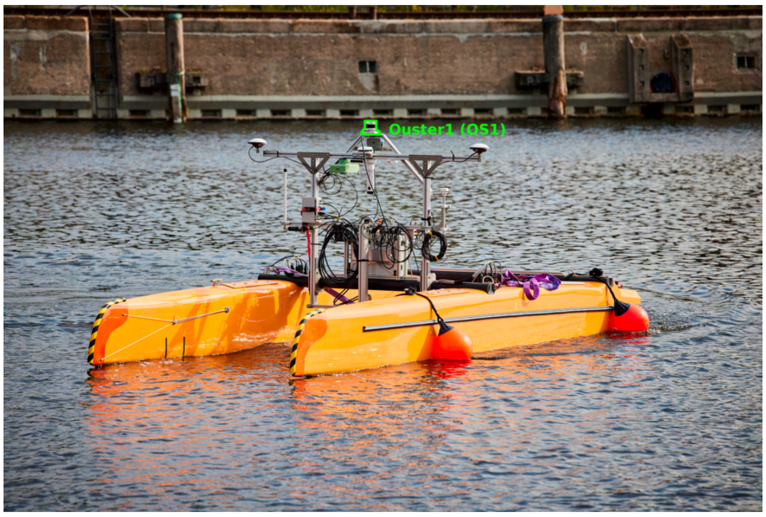

Figure 9: The LiDAR Ouster1 (OS1) is mounted on top of the mast of SeaML:SeaDragon USV. The recorded data of the LiDAR sensor is used for data collection, training, and evaluation for 3D object detection of boats.

Table 4: Settings used for the Ouster OS1 rev.7v3.

Ouster OS1 LiDAR	
Max. Range	up to 170 m
Min. Range	0.3 m
Range Accuracy	± 0.03 m
Vertical Resolution	128
Horizontal Resolution	1024
Rotation Rate	10 Hz

The LiDAR coordinate system is oriented as follows: the X-coordinate points to the back, the Y-coordinate points to the right, and the Z-coordinate is upwards. These coordinates are transformed into the OpenPCDet coordinate system: the X-coordinate points forward, the Y-coordinate points left, and the Z-coordinate is still upwards.

4.3 Training setup

OpenPCDet offers two *.yaml* files to configure the dataset generation and model configurations, for example, hyperparameters, layers for feature extraction, epoch, voxel size, etc. These files offer modular adjustments suitable for training to reach higher detection accuracies. The following tables show dataset (Table 5) and model settings (Table 6) for the experiments:

Table 5: OpenPCDet Dataset Configuration

Parameter	**Value**
`cfg.DATA_CONFIG.POINT_CLOUD_RANGE`	[−243.34063, −218.742655, −75.071892, 365.93937, 216.457345, 75.589012]
`cfg.DATA_CONFIG.VOXEL_SIZE`	[0.16, 0.16, 3.7665226]
`cfg.DATA_CONFIG.FILTER_BY_MIN_POINTS`	{`boat` : 50}
`cfg.DATA_CONFIG.SAMPLE_GROUPS`	{`boat` : 20}

Table 6: OpenPCDet Model Configuration

Parameter	**Value**
`cfg.OPTIMIZATION.BATCH_SIZE_PER_GPU`	`1`
`cfg.OPTIMIZATION.NUM_EPOCHS`	`40, 80`
`cfg.OPTIMIZATION.OPTIMIZER`	`adam_onecycle`
`cfg.MODEL.DENSE_HEAD.ANCHOR_GENERATOR_CONFIG`	[{'class_name': 'boat', 'anchor_sizes': [[36.28, 8.95, 14.82]], 'anchor_rotations': [0, 1.57], 'anchor_bottom_heights': [0], 'align_center': False, 'feature_map_stride': 8, 'matched_threshold': 0.55, 'unmatched_threshold': 0.4}]

The custom datasets were trained with a NVIDIA RTX A6000, and Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz on an Ubuntu 22.04.5 LTS machine.

5 Results

The following section presents quantitative results of 3D boat detection using PV-RCNN. The PV-RCNN models with different datasets and hyperparameters were evaluated on the validation set using KITTI metrics, as explained in section 2.4. Key metrics include Recall at IoU thresholds (0.3, 0.5, 0.7) and 3D detection performance (IoU=0.50/0.70) for bird's-eye view (BEV), 3D object detection (3D), orientation similarity (AOS), and bounding box (bbox) tasks. The distinct performance overview between Region-of-Interest (ROI) and (RCNN) stages allows for detailed insight into both the feature backbone and the refinement stage. The following Table 7 shows the Recall with different IoU thresholds. Then, Table 8 and Table 9 present the detection performance at IoU=0.5/0.7.

Table 7: Recall at thresholds 0.3, 0.5, and 0.7 for ROI and RCNN Stage

Dataset (Epoch)	ROI @ 0.3	ROI @ 0.5	ROI @ 0.7	RCNN @ 0.3	RCNN @ 0.5	RCNN @ 0.7
Base (40)	0.9291	0.9134	0.7165	0.9291	0.8976	0.6929
Base + 10° (40)	0.9863	0.9726	0.8571	0.9863	0.9746	0.9217
Base + 30° (40)	0.9911	0.9788	0.7929	0.9911	0.9833	0.8842
Base (80)	0.9764	0.8976	0.0079	0.9764	0.9685	0.1417
Base + 10° (80)	0.9980	0.9765	0.9022	0.9961	0.9804	0.9374
Base + 30° (80)	1.0000	0.9396	0.0039	1.0000	0.9981	0.1657

The Recall values in Table 7 with 0.3 and 0.5 thresholds remain high above 0.89 in all datasets and epochs, but significantly drop at the 0.7. Furthermore, the results demonstrate that data augmentation improves Recall performance: the *Base + 10* dataset achieves >90% RCNN recall at IoU=0.7 for both epochs. The results show that the effect of the rotation degrees affects the performance, as *Base + 10* outperforms *Base + 30.* Extended training (80 epochs) decreases the robustness for *Base + 30* and non-augmented *Base* datasets.

Table 8: 3D Detection Performance at IoU = 0.50

Dataset (Epoch)	bbox	BEV	3D	AOS	bbox_R40	BEV_R40	3D_R40	AOS_R40
Base (40)	97.09	71.79	62.96	77.61	98.14	71.85	64.82	77.16
Base + 10° (40)	96.57	90.28	90.03	94.26	98.80	95.91	95.36	96.33
Base + 30° (40)	95.55	88.70	86.74	43.02	98.14	91.33	87.92	42.79
Base (80)	98.78	76.22	58.59	64.40	99.36	80.34	56.97	64.79
Base + 10° (80)	97.18	90.66	90.66	72.62	99.06	96.36	96.34	74.02
Base + 30° (80)	91.02	0.27	0.16	48.81	91.82	0.13	0.04	49.36

Base + 10 (40) demonstrates strong performance, achieving high scores in BEV (90.28), 3D (90.03), and AOS (94.26). It shows robust 3D object detection and orientation estimation at a moderate IoU. *Base + 10 (40)* performs better than *Base (40)* and *Base + 30 (40)* in all 3D metrics, both with and without R40 evaluation. Overall, the performance results at IoU=0.5 show high accuracies. With *Base + 30 (80)*, the detection and orientation values severely drop, suggesting that aggressive data augmentation can lead to overfitting and degrading generalization. Additionally, the AOS values are generally lower, which highlights the model's ability on low object orientation estimation.
Comparing the moderate IoU=0.5 to the stricter IoU=0.7, all 3D detection metrics–BEV, 3D, and AOS–show a performance drop in all configurations. This emphasizes that the increased difficulty of

Table 9: 3D Detection Performance at IoU = 0.70

Dataset (Epoch)	bbox	BEV	3D	AOS	bbox_R40	BEV_R40	3D_R40	AOS_R40
Base (40)	97.09	33.72	20.10	77.61	98.14	28.61	14.76	77.16
Base + 10° (40)	96.57	85.09	66.20	94.26	98.80	85.88	69.36	96.33
Base + 30° (40)	95.55	56.51	41.05	43.02	98.14	54.18	39.55	42.79
Base (80)	98.78	1.01	0.51	64.40	99.36	0.51	0.06	64.79
Base + 10° (80)	97.18	89.06	76.53	72.62	99.06	89.83	78.23	74.02
Base + 30° (80)	91.02	0.04	0.04	48.81	91.82	0.01	0.01	49.36

precise object localization, especially regarding depth and orientation, is more difficult. Despite the performance drop, *Base + 10 (40)* shows superior generalization and robustness due to its relatively high performance values. In contrast, *Base (80)* and *Base + 30(80)* have very low 3D metrics scores that are close to 0. Compared to IoU=0.5, *Base + 30(80)* represents similar values while *Base (80)* values show a significant drop. Both configurations fail to generalize and perform complex 3D localization and orientation tasks, due to overfitting or ineffective generalization.

6 Discussions

The results of this study provide critical insights into the performance of PV-RCNN for 3D object detection in maritime environments, particularly under conditions of limited data availability. The experiments demonstrate that data augmentation techniques, specifically moderate rotations, significantly enhance the model's generalisation capability and accuracy in detecting boats across varying orientations and perspectives. By integrating the perception module into the decision-making process of the GNC system, the model can effectively contribute to the high level of autonomy envisioned for USVs operating under the NJORDIQ-stack. Specifically on the docking task envisioned in the SeaClear2.0 Project, the perception module's output can be utilised to provide precise pose information required for successful docking onto another USV, even in confined spaces, such as harbours or marinas. Previous works have relied on the assumption of exact GNSS position and orientation for docking, an assumption that may not hold in environments where GNSS signals are obstructed or unreliable. The integration of the perception module offers a more robust and adaptive approach, enabling the USV to navigate dynamic environments and accurately detect and localise boats, even under challenging conditions.

6.1 Performance

The training and evaluation of PV-RCNN for LiDAR-based 3D object detection presents several challenges, including critical insights regarding data augmentation, training duration, configuration settings, and localization precision.
Firstly, the results indicate that data augmentation is crucial and enhances the model performance. Compared to the *Base (40/80)* dataset, the performance of the *Base + 10(40)* dataset shows superb generalization and robustness. A moderate rotation (10°) leads to precise localization and helps the model generalize to minor variations in object orientation and perspective, which are common in real-world scenarios. However, a higher rotation (30°) increases data diversity and complexity. Combined with extended training, the model struggled to predict the orientation and localization, potentially due to exposure to unrealistic, very wide, and rotational variations.
Secondly, the duration of training has a complex and conditional impact on model performance. The results show that extended training (80 epochs) leads to overfitting for both non-augmented and augmented datasets, resulting in reduced detection performance. This suggests that extended training does not necessarily refine the model's understanding or improve localization precision. Further work could include validation monitoring and early stopping protocols.
Finally, achieving high localization precision at a strict IoU=0.7 threshold remains challenging. As seen in the results, the drop in recall and 3D detection metrics indicates that while objects are detected, the model struggles with accurately localizing fine-grained features.

6.2 Maritime dataset and data augmentation

A small base set of manually annotated real-world samples can lead to promising results using advanced augmentation techniques. Moreover, different dataset configurations (point cloud range, voxel size, etc.) should be explored to observe their impact on 3D detection performance. Nevertheless, the custom-collected maritime data presents limitations, such as variable scenes, missing points in boat shapes, ship type scarcity, and complex scenes in larger harbours. Furthermore, applying data augmentation does not introduce novel scenes, and it can lead to overfitting. This could also result in a lack of ability to generalize when the model encounters novel scenes or ship types. In future works, more diverse data should be collected to test different configurations and improve the robustness of the model in novel maritime scenes.

7 Conclusion

The current work addressed a critical challenge in maritime perception: the lack of large-scale LiDAR datasets required for training robust 3D object detection models. The research goal was to analyse to what extent data augmentation techniques can enable PV-RCNN, a model known for its high accuracy on 3D detection benchmarks, to achieve reliable boat detection when trained on a small, custom point cloud dataset. The experiments were conducted using a dataset acquired via the onboard sensory system of a small-sized USV. The envisioned scenario is the autonomous navigation and docking of the aforementioned USV onto a second USV, in the context of marine litter collection. Experiments showed that a moderate rotation of 10° achieved optimal results, leading to better generalization and robustness compared to using no augmentation or applying larger rotation angles during augmentation. In contrast, increased training duration combined with higher rotations led to reduced performance and potential overfitting. The primary conclusion of this work is that the parameters of the augmentation strategy itself are a critical factor. While the technique is an effective method for overcoming data scarcity in the maritime domain, this study demonstrates that achieving robust detection is highly dependent on moderate transformations, as shown by 10° performance results. Future work will explore different configurations (e.g., batch size, number of epochs, additional object classes) and focus on collecting more diverse data to improve model robustness in novel maritime environments.

8 Acknowledgement

The work presented has been conducted within the SeaClear2.0 Project and has been co-funded by the European Union under the Horizon Europe Programme (Grant Agreement no. 101093822).

References

[1] Emanuele Coccolo, Cosmin Delea, Fabian Steinmetz, Roberto Francescon, Alberto Signori, Ching Nok Au, Filippo Campagnaro, Vincent Schneider, Federico Favaro, Johannes Oeffner, Bernd-Christian Renner, and Michele Zorzi. System architecture and communication infrastructure for the robovaas project. *IEEE Journal of Oceanic Engineering*, 48(3):716–739, July 2023.

[2] Ping Yang, Changhui Song, Linke Chen, and Weicheng Cui. Image based river navigation system of catamaran usv with image semantic segmentation. In *2022 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*, pages 147–151, 2022.

[3] Thor I Fossen. *Handbook of marine craft hydrodynamics and motion control.* John Wiley & Sons, 2011.

[4] Gongxing Wu, Debiao Li, Hao Ding, Danda Shi, and Bing Han. An overview of developments and challenges for unmanned surface vehicle autonomous berthing. *Complex & Intelligent Systems*, 10(1):981–1003, August 2023.

[5] Yuanyuan Qiao, Jiaxin Yin, Wei Wang, Fábio Duarte, Jie Yang, and Carlo Ratti. Survey of deep learning for autonomous surface vehicles in marine environments. *IEEE Transactions on Intelligent Transportation Systems*, 24(4):3678–3701, April 2023.

[6] Qiuyu Zhang, Lipeng Wang, Hao Meng, Wen Zhang, and Genghua Huang. A lidar point clouds dataset of ships in a maritime environment. *IEEE/CAA Journal of Automatica Sinica*, 11(7):1681–1694, July 2024.

[7] Zilong Lu, Baoan Li, and Jiping Yan. Research on unmanned surface vessel perception algorithm based on multi-sensor fusion. In *2022 4th International Conference on Frontiers Technology of Information and Computer (ICFTIC)*, page 286–289. IEEE, December 2022.

[8] Yousef Abd Alhattab, Zulkifli Bin Zainal Abidin, Ahmed Rimaz Faizabadi, Hasan F. M. Zaki, and Ahmad Imran Ibrahim. Integration of stereo vision and moos-ivp for enhanced obstacle detection and navigation in unmanned surface vehicles. *IEEE Access*, 11:128932–128956, 2023.

[9] Guotong Chen, Jiangtao Qi, and Zeyu Dai. *Real-Time Maritime Obstacle Detection Based on YOLOv5 for Autonomous Berthing*, page 412–427. Springer Singapore, 2022.

[10] Juri Zach, Christian Busse, Steffen Funk, Christian Möllmann, Bernd-Christian Renner, and Tim Tiedemann. Towards non-invasive fish monitoring in hard-to-access habitats using autonomous underwater vehicles and machine learning. In *Oceans 2021 : San Diego – Porto*, OCEANS 2021: San Diego-Porto, 2022.

[11] Juri Zach and Tim Tiedemann. First results of a low-cost visual odometry approach for autonomous underwater localization and fish habitat mapping, 2023.

[12] Jiaying Lin, Phillip Diekmann, Christian-Eike Framing, René Zweigel, and Dirk Abel. Maritime environment perception based on deep learning. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):15487–15497, September 2022.

[13] Joanna Stanisz, Konrad Lis, Tomasz Kryjak, and Marek Gorgon. Optimisation of the pointpillars network for 3d object detection in point clouds. 2020.

[14] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. `https://github.com/open-mmlab/OpenPCDet`, 2020.

[15] Milin Patel and Rolf Jung. Simulation-based performance evaluation of 3d object detection methods with deep learning for a lidar point cloud dataset in a sotif-related use case. In *Proceedings of the 10th International Conference on Vehicle Technology and Intelligent Transport Systems*, page 415–426. SCITEPRESS - Science and Technology Publications, 2024.

[16] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, July 2017.

[17] Duarte Fernandes, António Silva, Rafael Névoa, Cláudia Simões, Dibet Gonzalez, Miguel Guevara, Paulo Novais, João Monteiro, and Pedro Melo-Pinto. Point-cloud based 3d object detection and classification methods for self-driving applications: A survey and taxonomy. *Information Fusion*, 68:161–191, April 2021.

[18] Yutian Wu, Yueyu Wang, Shuwei Zhang, and Harutoshi Ogai. Deep 3d object detection networks using lidar data: A review. *IEEE Sensors Journal*, 21(2):1152–1171, January 2021.

[19] Ying Li, Lingfei Ma, Zilong Zhong, Fei Liu, Dongpu Cao, Jonathan Li, and Michael A. Chapman. Deep learning for lidar point clouds in autonomous driving: A review. *CoRR*, abs/2005.09830, 2020.

[20] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: point-voxel feature set abstraction for 3d object detection. *CoRR*, abs/1912.13192, 2019.

[21] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.

[22] Yutian Wu, Yueyu Wang, Shuwei Zhang, and Harutoshi Ogai. Deep 3d object detection networks using LiDAR data: A review. *IEEE Sensors Journal*, 21(2):1152–1171, 2021.

[23] Saifullahi Aminu Bello, Shangshu Yu, and Cheng Wang. Review: deep learning on 3d point clouds, 2020.

[24] Ouster Inc. Ouster os1 rev7v3 lidar sensor datasheet. `https://data.ouster.io/downloads/datasheets/datasheet-rev7-v3p0-os1.pdf`, 2024. [Accessed 16-06-2025].