Received March 15, 2021, accepted March 29, 2021, date of publication April 2, 2021, date of current version April 12, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3070694

Advancing Autonomous Surface Vehicles: A 3D Perception System for the Recognition and Assessment of Docking-Based Structures

MARIA INÊS PEREIRA, RAFAEL MARQUES CLARO, PEDRO NUNO LEITE, AND ANDRY MAYKOL PINTO
Centre for Robotics and Autonomous Systems, INESC TEC, 4200-465 Porto, Portugal
Faculty of Engineering, University of Porto, 4200-465 Porto, Portugal

Corresponding author: Maria Inês Pereira (maria.i.pereira@inesctec.pt)

This work was supported in part by the European Regional Development Fund (ERDF) through the Operational Programme for Competitiveness and Internationalization—COMPETE 2020 Programme, in part by the National Funds through the Portuguese Funding Agency, Fundação para a Ciência e a Tecnologia (FCT), under Project POCI-01-0145-FEDER-030010, and in part by the European Union's Horizon 2020—The EU Framework Programme for Research and Innovation 2014–2020, under Grant 871571.

ABSTRACT The automation of typically intelligent and decision-making processes in the maritime industry leads to fewer accidents and more cost-effective operations. However, there are still lots of challenges to solve until fully autonomous systems can be employed. Artificial Intelligence (AI) has played a major role in this paradigm shift and shows great potential for solving some of these challenges, such as the docking process of an autonomous vessel. This work proposes a lightweight volumetric Convolutional Neural Network (vCNN) capable of recognizing different docking-based structures using 3D data in real-time. A synthetic-to-real domain adaptation approach is also proposed to accelerate the training process of the vCNN. This approach makes it possible to greatly decrease the cost of data acquisition and the need for advanced computational resources. Extensive experiments demonstrate an accuracy of over 90% in the recognition of different docking structures, using low resolution sensors. The inference time of the system was about 120ms on average. Results obtained using a real Autonomous Surface Vehicle (ASV) demonstrated that the vCNN trained with the synthetic-to-real domain adaptation approach is suitable for maritime mobile robots. This novel AI recognition method, combined with the utilization of 3D data, contributes to an increased robustness of the docking process regarding environmental constraints, such as rain and fog, as well as insufficient lighting in nighttime operations.

INDEX TERMS Autonomous surface vehicle, docking, object recognition, point cloud.

I. INTRODUCTION

Over the past few years, Artificial Intelligence (AI) has been embraced by nearly every industry, enabling machines to carry out tasks which typically require human intellect. The maritime industry is no exception, but it has only taken its first steps towards automation. The development of Autonomous Surface Vehicles (ASVs) falls under this scope, making it possible to greatly reduce the role of human intervention in this industry and the inherent consequences of human error. When navigating through real unstructured maritime environments, this type of vehicle faces many challenges, such as harsh environmental conditions and high cluster density. However, there is in general a "*lack of research in autonomous decision-making tools*" [1] in the field of robotics to enable autonomous vessels to tackle such challenges. There is a need for the development of more appropriate sensors to provide vessels with improved spatial awareness [2], but also of new navigation methods that ensure collision-free trajectories [3], [4]. This shortage of robust systems is particularly noticeable with respect to one of the key functions of an autonomous vessel: docking. The docking process of an autonomous vessel is illustrated in figure 1. A robust docking maneuver is fundamental for autonomous vessels to be safely deployed to real operations and requires the real-time detection and recognition of the dock. A number of reasons make this a critical maneuver, while the complexity of seaports and harbours make them a location prone to

The associate editor coordinating the review of this manuscript and approving it for publication was Emre Koyuncu.

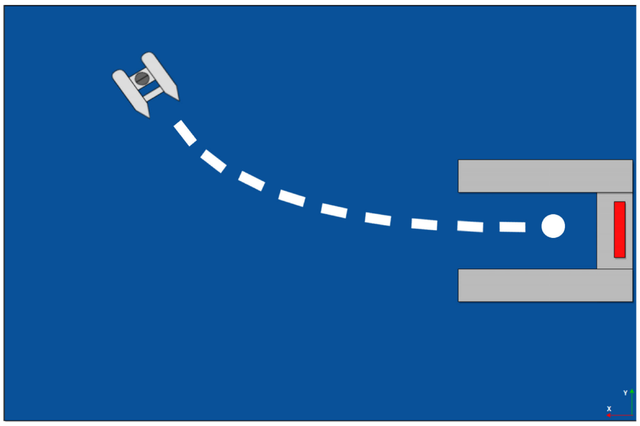


FIGURE 1. Illustration of the docking maneuver of an Autonomous Surface Vehicle towards a docking structure.

accidents [5]. Yet, current works do not present the mechanisms necessary to cope with these constraints. Oftentimes, they are based on inadequate representation models of the surrounding environment, which do not provide the required spatial information, and/or establish their performance on simulated environments, which cannot represent the challenges of a real setting.

This research aims to advance the autonomous docking process of a vessel, by providing a perception tool necessary for this maneuver. The developed work proposes a Deep Learning (DL) classifier able to identify the type of a docking structure based on 3D information retrieved by sensors onboard of the vehicle moving in any direction. The classification of the type of dock is considered an essential step for docking because the maneuver should take into account the shape of the structure. Different layouts may require a frontwards docking maneuver or a side-ways one, for example. In this manner, the identification of the type of dock is vital for a collision-free docking process. For the development of this recognition system, a synthetic-to-real domain adaptation methodology was adopted to ensure robustness to the challenges posed by authentic maritime environments while reducing computation and data acquisition costs. The contributions of this article include:

- A Deep Learning approach for the recognition of docking structures based on 3D point clouds for an Autonomous Surface Vehicle. Inference is performed in real time due to a lightweight volumetric Convolutional Neural Network (vCNN) based approach;
- A synthetic-to-real domain adaptation methodology to accelerate the training process. Synthetic source data was generated with different levels of noise to approximate it as much as possible to the target domain (real environment), thus decreasing domain bias, i.e., divergence between domains;
- The integration of the classification module with a detection component, forming a real-time system that operates at more than 8Hz with low computational resources;
- Extensive experiments in a real commercial harbour, using an Autonomous Surface Vehicle that was able to classify docking structures with an accuracy higher than 90%.

II. RELATED WORK

Most current object recognition methods were developed within the 2D domain. Fast R-CNN [6] and YOLO [7] are a couple of examples of these already consolidated strategies which were widely employed in 2D applications. However, the nature of the task at hand requires a robustness level that a 2D-based system cannot provide. A 3D representation of the environment is able to provide a different level of spatial awareness and, therefore, it is fundamental for this work. Nonetheless, as explained below, 3D object recognition is a very recent topic. There are not many works which explore this subject and those which employ Deep Learning are even fewer. The first part of this section focuses on 3D generic object recognition, describing some representation models and architectures. The second part summarizes the few existing dock recognition strategies found in the literature, which make use of some of the mentioned 3D data representations.

The performance of a 3D object recognition method is highly dependent on the representation model it uses for the input data. A point cloud is an unstructured and unordered set of 3D points of rather inexpensive acquisition. Its processing, on the other hand, is a demanding task. PointNet [8] is an example of a method which deals with raw point clouds. This network presents good performance, but has a high number of parameters and, therefore, requires a high computational effort. Volumetric representations, such as voxel grids and octrees, are a common strategy to deal with this issue. A voxel grid characterizes data as a 3D grid in which each voxel is classified as free or occupied, making it rather easy to access and manipulate data. Among the works analysed in this scope, a few of the developed architectures stood out, such as VoxNet [9] and LightNet [10]. These 3D CNNs have a rather low number of parameters, while being amongst the most accurate methods. An octree representation may allow for an even faster processing of the input data. This modeling approach is similar to voxel grids, but it is based on varying-sized voxels. This means that a large empty space may be represented as a single voxel and, therefore, a lot of computation steps may be skipped. This feature makes octrees more memory efficient than other representations. However, due to its hierarchical structure, it does not allow for efficient access to the underlying data [11].

Considering classification methods based on Deep Learning, an important factor that influences performance is the selection of the optimizer. A study by Bera *et al.* in 2020 [12] analysed the performance of several optimizers, such as SGD, Adagrad, Adadelta, RMSprop, Adam and Nadam, and found that regarding classification accuracy Adam presented the best results.

When shifting the application of object recognition from a generic matter to a more specific topic, such as docking and harbour platforms, the work developed by the scientific community is found to be quite scant and does not present suitable results. For example, in 2014, J. M. Esposito and M. Graves proposed an algorithm to detect docking platforms from 3D LiDAR scans [13]. This work achieved quite accurate results, but was developed on top of data collected from a LiDAR mounted onshore and not on a moving vehicle. Furthermore, the filtering and detection of a single structure took over 10s and, thus, the system did not provide real-time operation, which is of major importance in the field of robotics. The task of recognition of docking structures is also explored in the Maritime RobotX Challenge,[1] a well-known robotics competition. A few of the solutions proposed by participants [14], [15] made use of LiDAR and/or image data to solve these challenges. Despite having obtained good results, these methods relied upon the geometric symbols imprinted in the docks. Another approach to the detection of docking structures for ASVs was that proposed by P. Leite *et al.* in 2019 [16]. The work consisted in an hierarchical approach to template matching which, although having achieved a good performance for minor and more severe environmental circumstances, required the structure to have a certain shape. A different strategy was developed by M. I. Pereira *et al.* in 2020 [17], in which the detection is carried out by a neural network that achieved an accuracy of approximately 96%, being robust to several degrees of Gaussian noise. Despite the accurate results, this work only produces a prediction about whether there is a dock present in the data or not and does not extract any other information regarding the structure.

In conclusion, there are many gaps to fill in this subject. Some of the presented approaches attempted to build methods suitable for real maritime environments, but required high computational resources. Others achieved a fast operation, but were only tested in simulated scenarios and/or dealt with a very low diversity of structures. The robotics field requires precision, speed and robustness to the challenges posed by a real environment. This paper presents a strategy to tackle all of these issues.

III. 3D REAL-TIME RECOGNITION OF DOCKING STRUCTURES

Most of the related work focuses solely on the detection of the docking structure. This article goes a step further in the topic of perception of docking platforms, providing information such as the type of structure. This assessment is necessary for the docking approach given that the maneuver can vary according to the layout of the dock. Furthermore, this work proposes a method to deal with the challenges posed by 3D recognition, a very unexplored topic, by developing a system focused on the balance between speed and precision. A 3D LiDAR approach is robust to different lighting conditions and harsh environmental constraints such as rain and fog. Moreover, a 3D scene representation provides a more complete portrait of the environment, including depth information, allowing the extraction of the 3D position of objects.

To ensure real-time inference, the recognition system is based on a lightweight volumetric Convolutional Neural Network (vCNN) with a low number of parameters. The input point clouds are also subject to a pre-processing stage to further reduce inference time. To obtain a robust inference system, which is able to perform in different conditions and in a multitude of scenarios, it is necessary to have access to a large quantity of data for the training process of the network. However, there are no datasets available targeted for real maritime environments and different types of docking structures. The acquisition process of this kind of data is also very difficult and expensive to carry out. Synthetic-to-real-domain adaptation methodologies have been adopted and obtained success in other tasks employing Deep Learning [18]–[20], taking advantage of the simplicity and inexpensiveness of generating synthetic data, and then building a strategy to adapt the system to a real scenario. The objective is to accelerate the training process using a large amount of synthetic data, which will be generated in a way to minimize domain bias, that is, the divergence between the synthetic and real domain. To accomplish this feature approximation between domains, different techniques will be employed, such as the utilization of a simulated maritime environment and different docking structures, as well as the addition of noise, scaling and downsampling. Due to this approach, the development of the network will be complemented by a training process on a small amount of data collected from a real seaport, which greatly decreases the cost of data acquisition, to fine-tune the model and ensure its applicability to real maritime environments.

In the context of this research work, two distinct problems arose: detecting whether there is a docking structure in the vehicle's surrounding environment, thus providing its location relative to the vehicle, and identifying the type of dock from a set of pre-defined layouts. The recognition of the shape is critical for the maneuver. Given these circumstances, developing a set of two sequential classifiers was the adopted strategy. Figure 2 illustrates the architecture of this cascade classifier. The first classifier has a binary output: the input point cloud either contains a dock or it does not. These two cases represent the two output classes for the first model [17]. If a dock is detected, the input data will proceed to the second model, where the dock will be classified according to its type. The development of this module is described throughout the rest of the paper. Considering the most common dock layouts found in commercial harbours and the primary shapes which constitute them, the models in figure 3 were selected and labeled as *Y-Shape*, *U-Shape*, *T-Shape*, *L-Shape*, and *Straight*. The classification module entails seven output classes ('Y', 'U', 'T', 'L', 'S', 'N', 'O'): one for each of the five models aforementioned and two additional ones. The first of the latter two classes ('N') represents the case in which the extracted information is not enough to identify the type of structure. In this case, the vehicle should move closer to the

[1] https://robotx.org/

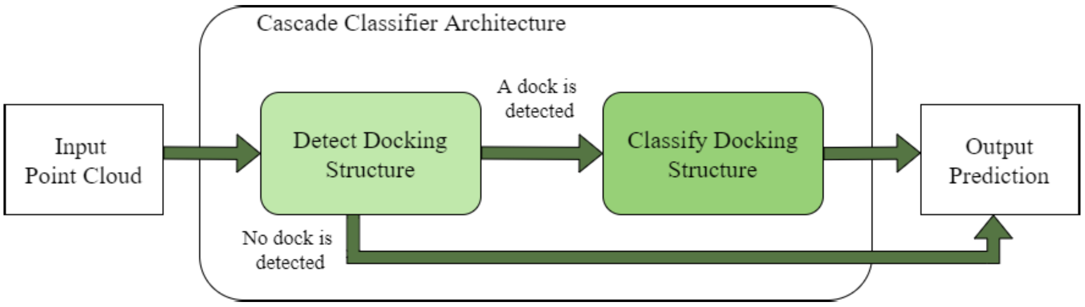


FIGURE 2. **Integration of the detection and classification of docking structures in a single sequential system that takes as input a LiDAR point cloud.**

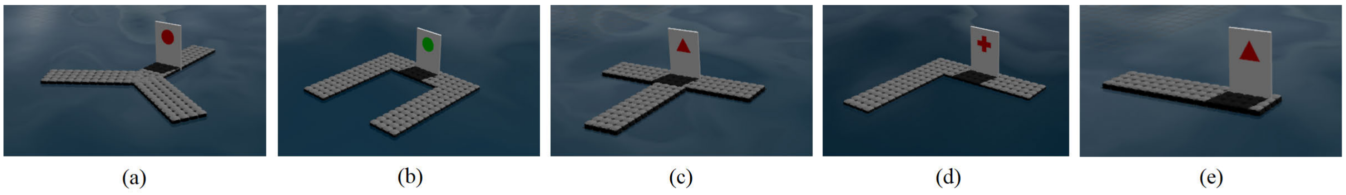


FIGURE 3. **The five different dock models considered during the development of the system: *Y-Shape* (a), *U-shape* (b), *T-shape* (c), *L-shape* (d), and *Straight* (e). The resulting seven classes of the inference model are 'Y', 'U', 'T', 'L', 'S', 'N', 'O', and include these five layouts and the 2 additional cases previously described.**

dock and the classifier should compute a new prediction. The classifier is also allowed to reject the detector's prediction. If the input data features differ from the generic features of a docking structure, the classifier should be able to conclude that it does not represent a dock after all and assign such an instance to class 'O'.

The remainder of this paper is organized as follows: Section III-A presents the first stage of the synthetic-to-real domain adaptation method, describing the techniques for approximating the two domains (III-A1), the architecture of the network (III-A2 and III-A3), and several experiments to evaluate the system's robustness and its behaviour in a dynamic setup (III-A4 and III-A5). The second stage of the adaptation process to a real domain is presented in section III-B, depicting the fine-tuning of the vCNN and the experiments conducted in a real commercial harbour.

A. TRAINING ACCELERATION WITH SYNTHETIC DATA GENERATION

1) SYNTHETIC DATA GENERATION

As explained above, the network will be trained over a large corpus of synthetically generated data. The DORA@CRAS dataset [17] was utilized for this purpose. This process of generation of synthetic data was carried out within the 3D simulator Gazebo, which is able to simulate the dynamics of a maritime environment, such as tidal waves and wind, and the kinematics of the vessel, thus providing a framework similar to a real maritime setting. It also allows for the 3D representation of a great variety of objects, such as docks, buoys, and boats. An ASV model equipped with several sensors was placed in the simulated scenario, as depicted in figure 4,

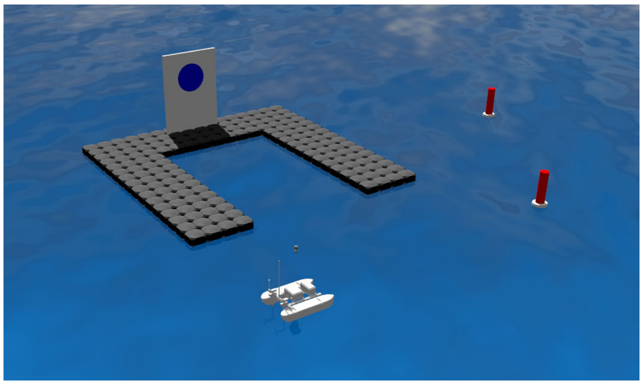


FIGURE 4. **Simulated maritime environment in Gazebo, in which data was acquired.**

to gather information regarding the docking structures. The specifications of these sensors are given below:

- **LiDAR** - *HDL-64E* - Frame Rate: 10Hz, Range: 120m, Range Accuracy: 0.02m, Field of View: 360° horizontal and 41.34° vertical;
- **Stereo Camera** - *Mako G-125* - Frame Rate: 30Hz, Resolution: 1292 x 964, Field of View: 80° horizontal;
- **IMU** - *MTi-30 Xsens* - Frame Rate: 200 Hz, Angular Accuracy: 0.2°/0.5°;
- **GPS** - *Swift Navigation* - Frame Rate: 20 Hz, L1/L2 RTK, Accuracy: < 0.04m.

The dataset is composed of circa 38 000 data instances of LiDAR points clouds, stereo images, IMU, and GPS information, which can be accessed in an online repository.[2]

[2]https://doi.org/10.25747/xpj9-2j17

From the generated data, only the output of the LiDAR was needed for the proposed recognition method. To obtain an accurate assessment of the network, it is necessary to utilize different data for the training and evaluation processes. Therefore, the point clouds of the dataset were split into three fractions: training (72%), validation (18%), and testing (10%) data.

a: DATA PRE-PROCESSING

Processing the input data plays a major role not only in improving the accuracy of the system but also in ensuring a real-time operation. A point cloud is a very unstructured 3D representation and, thus, a pre-processing stage greatly contributes to facilitate the inference of the model and speed up the classification process. Initially, the points clouds have to be filtered to remove outliers. An infra-red based LiDAR sensor of 905 nm wavelength such as the one considered in this work does not obtain returns from water surfaces, given that the light is only able to penetrate through 1 cm below the water [21]. However, a drawback of 3D simulator Gazebo is that it is not able to replicate this characteristic and, thus, the majority of the points in the LiDAR scan correspond to the surface of the water, as depicted in figure 5a. Due to this feature, the point cloud can be approximated to a plane and then, one can define a certain threshold with respect to the plane's surface, below which all points should be removed. The result is illustrated in figure 5b.

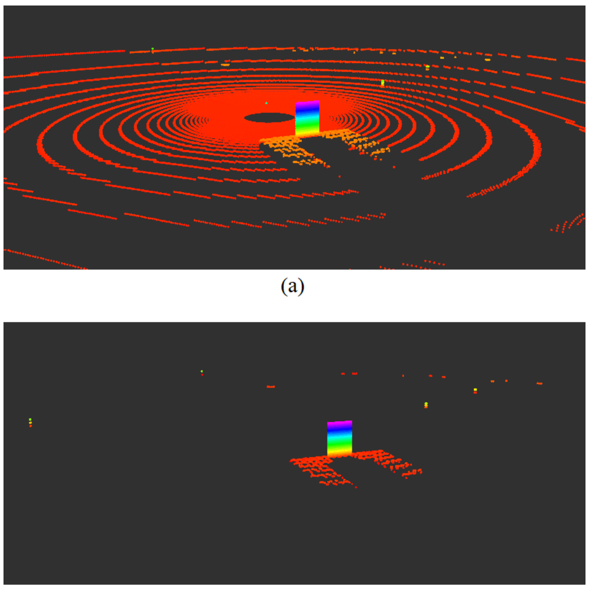


FIGURE 5. Example of a point cloud before (a) and after (b) the filtering stage which removed the points that corresponded to the surface of the water.

Secondly, each object in the point cloud is isolated through an Euclidean clustering approach [22]. This stage will enable the system to easily extract the docking structure's location relative to the ASV and will facilitate the classification process by providing input data representing solely the dock and dismissing the rest of the point cloud's content, thus reducing inference time. Finally, to obtain a more lightweight and structured representation of the data, the point clouds are converted to voxel grids, as those represented in figures 6a and 6b. The grid is formed based on a binary occupancy approach, which means that each voxel is defined as being either free or occupied. The definition of the grid resolution is done through a trade-off between ensuring enough object detail and minimizing the processing time. In this case, each grid has 32 × 32 × 32 voxels, which is in fact a 24 × 24 × 24 grid with padding of 8 voxels. Padding the voxel grid prevents data from becoming too sparse.

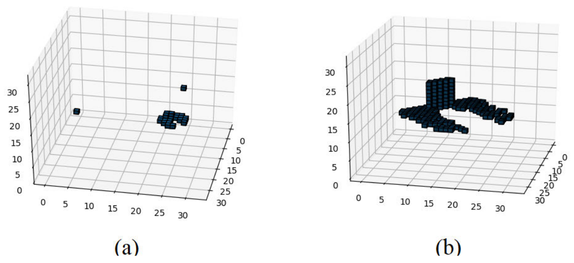


FIGURE 6. Result of the voxelization of a point cloud (grid resolution of 32 × 32 × 32) before (a) and after clustering (b).

b: DATA AUGMENTATION

Data augmentation is a crucial step in increasing the robustness of the system, given that it contributes to decreasing domain bias, i.e., the divergence between the synthetic and real domains. Networks trained within a synthetic environment usually do not function well in real settings [18]. However, if one renders synthetic data as realistic as possible, through the implementation of data augmentation techniques, for example, then the system will need no or minor fine-tuning to be able to perform in authentic environments. Three techniques were defined for the augmentation of data: scaling, downsampling, and the addition of Gaussian noise. Table 1 presents an overview of the chosen methods and specifications.

TABLE 1. Data Augmentation Parameters.

Augmentation	Specifications	
Scaling	Scaling factor between 0.6 and 1.5	
Downsampling	Downsampling factor between 0.5 and 1.0	
Gaussian Noise	Network	Standard Deviation
	Low-Level Robustness (LRN)	{0.0}
	Mid-level Robustness (MRN)	{0.05;0.1;0.3;0.5}
	High-level Robustness (HRN)	{0.1;0.5;1.0}

The first technique has the objective of preventing the network from learning the size of a dock as an invariable characteristic of these structures. Thus, each point cloud was subject

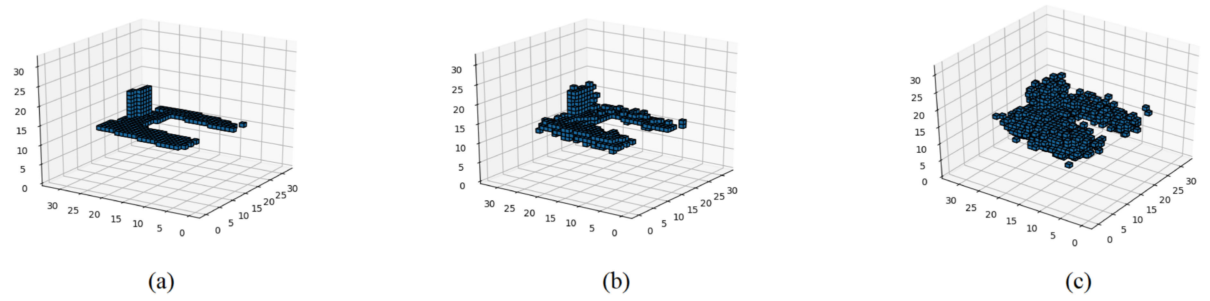


FIGURE 7. Effect of different degrees of Gaussian noise on voxel grids representing a docking structure. The noise is quantified by the standard deviation (σ): $\sigma = 0.0$m (a), $\sigma = 0.1$m (b), and $\sigma = 0.5$m (c).

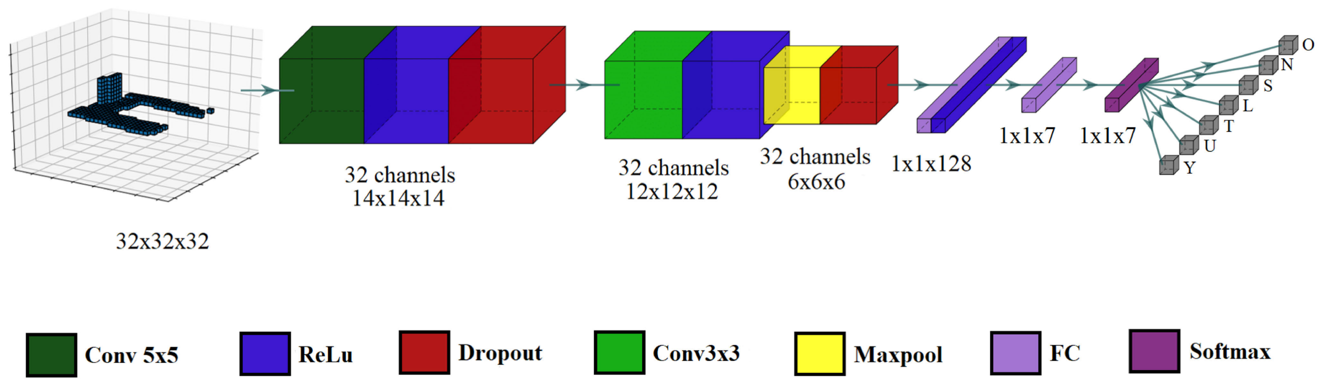


FIGURE 8. Network Architecture: two convolution stages, and two fully connected layers followed by a softmax layer, which outputs a probability for each of the 7 classes. The dimensions of the output channels of the layers are presented as Height x Width x Depth.

to an operation of up-scaling or down-scaling by a factor that randomly varied between 0.6 and 1.5. This approach also enables the system to recognize structures in a wide range of distances. Secondly, considering that physical sensors, rather than simulated ones, may not be able to acquire data with such density, a downsampling operation was applied to the point clouds, retaining between 50% and 100% of the points in each instance. Finally, to ensure the network is robust, a noise component was added to the point clouds. This operation simulates the intrinsic error of the sensors and the perturbation caused by adverse environmental conditions. According to E. T. Jaynes [23], when there is not sufficient data to estimate the noise distribution, then a Gaussian form is a good fit for such distribution. Figure 7 illustrates a few examples of the effect of the addition of noise to data. To apply this method of data augmentation, three Gaussian noise setups were considered. The model was trained individually for each of the three setups, resulting in three distinct networks, as depicted in table 1. In the first scenario, no noise was applied to the data (*Low-level Robustness Network - LRN*). The second setup, denominated *Mid-level Robustness Network - MRN*, corresponds to significant but not extreme values of standard deviation of the Gaussian noise, whereas the third and last one, called *High-level Robustness Network - HRN*, involves severe noise degrees.

2) LIGHTWEIGHT VOLUMETRIC CNN ARCHITECTURE

The dock classification module is composed of a volumetric Convolutional Neural Network. Taking into consideration the research conducted on related works, a model based on the VoxNet [9], which was pre-trained on the ModelNet dataset [24], achieving an accuracy of 81%, was utilized as the basis of the training process. Figure 8 illustrates the architecture of the network that constitutes the classifier. This network takes as input a voxel grid of size $32 \times 32 \times 32$ and includes two convolution stages, each composed of a Convolutional, a ReLU, and a Dropout layer. The second stage also contains a Maxpool layer which downsamples the data by a factor of 2. Two Fully Connected (FC) layers follow these stages and flatten the data, resulting in a number of output neurons equal to the number of classes, which is 7 in this case. The output of the final layer is utilized to calculate the softmax probabilities [25] for each of the seven classes ('Y', 'U', 'T', 'L', 'S', 'N', 'O'). Each of these probabilities represents the certainty of the model concerning the corresponding prediction.

The Cross-Entropy Loss, also referred to as Logarithmic Loss, is the most suitable for both binary and multi-class classification problems [26] and, thus, it was selected for the optimization process. For the training of the network of the classifier, several optimizers that are used in state-of-the-art methods [12] were experimented with, and those which achieved the best results, depicted in the following section, were the following: Adadelta, Adam, RMSprop, and SGD (Stochastic Gradient Descent).

To prevent the weights of a neural network from overfitting the training data, a strategy called layer freezing was adopted, thus avoiding a significant decline of performance between the training and testing phases. With this approach, one freezes the parameters of any chosen layer of the network, thus preventing them from being updated during the optimization stage. These layers are then gradually unfrozen during the training process. Another advantage of this strategy is the preservation of the ability of the pre-trained model to capture low-level features. Layers which are closer to the input of the model extract these lower-level features, while those which are farther extract higher-level ones, that is, features which are specific for the context of the task [27]. Therefore, a layer freezing strategy intends to apply a more intensive retraining to the latter layers. In this case, the two convolutional layers and the first fully connected one, as illustrated in figure 8, were the selected layers for implementing this approach.

3) MODEL SELECTION

A few initial tests were performed to select the optimal values for the hyperparameters. To minimize the duration of this initial process, these tests were conducted on a fraction of the training data, selected in a way to thoroughly represent the entire dataset. All training instances considered the Cross-Entropy as loss function, and a batch size of 1. The best learning rates obtained for each optimizer are presented in table 2.

TABLE 2. Best learning rates for each optimizer.

Optimizer	Adadelta	Adam	RMSprop	SGD
Learning Rate	1.000	0.001	0.001	0.001

Figure 9 presents the results of the training instances conducted with the layer freezing strategy (9b) and without it (9a). The layer freezing approach is indeed able to reduce overfitting. Overall, the optimizer Adam presents the highest accuracy, while SGD obtains a comparable performance with more stable results. Both optimizers where considered in further tests, in which the batch sized was varied within the set {1;2;4;8;16;32;64;128}. When trained with Adam, the network's accuracy obtained a maximum increase of 5% for a batch size of 16, whereas SGD presented a downward behaviour, showing no improvement when compared to the previous tests. Considering all the tests conducted up to this point, the training of the network of the classifier was carried out according to the following parameters:

- Optimizer: Adam (learning rate = 0.001);
- Loss function: Cross-Entropy;
- Batch size: 16;
- Layer freezing: Yes.

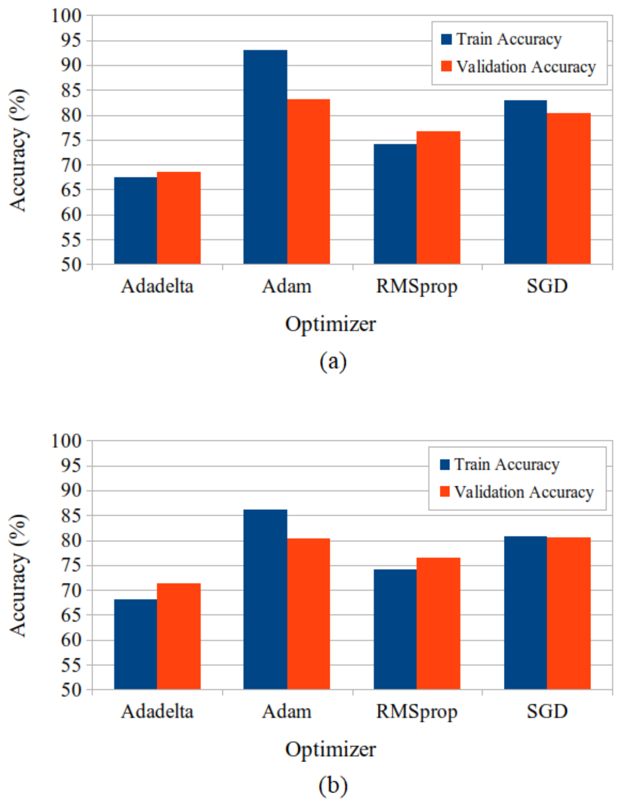


FIGURE 9. Train and validation accuracy obtained with (b) and without (a) a layer freezing strategy, using different optimizers. The training process was conducted for 10 epochs, with a batch size of 1 and the Cross-Entropy loss.

4) ROBUSTNESS EVALUATION

A classification model, be it a binary or a multi-class one such as the case studied in this work, can be evaluated by several metrics [28]. Usually these measures are computed from the confusion matrix, which is a table that presents the number of data instances for each combination of predicted and actual class. The matrix has a dimension of K rows per K columns, in which K is the number of classes, which is 7 in this case. The accuracy of the model corresponds to the percentage of correctly predicted instances and can be calculated as a per-class average, according to (1). The precision corresponds to the percentage of instances classified as belonging to a certain class which were effectively of that class, as demonstrated in (2). Equation (3) shows that the recall represents the fraction of the instances of a certain class which were correctly classified. This measure can be described as the "*effectiveness of a classifier to identify positive labels*" [28]. In the expressions below, *TP* refers to *True Positive* instances, *TN* to *True Negative*, *FP* to *False Positive*, and *FN* to *False Negative*.

$$Accuracy = \frac{\sum_{i=1}^{K} \frac{TP_i+TN_i}{TP_i+FP_i+TN_i+FN_i}}{K} \tag{1}$$

$$Precision = \frac{\sum_{i=1}^{K} \frac{TP_i}{TP_i+FP_i}}{K} \quad (2)$$

$$Recall = \frac{\sum_{i=1}^{K} \frac{TP_i}{TP_i+FN_i}}{K} \quad (3)$$

The classification model was trained on the full dataset according to the parameters defined in the previous section. As explained before, three noise setups were considered during this training process. Therefore, the training of the model resulted in three distinct networks: *Low-level Robustness Network (LRN)*, *Mid-level Robustness Network (MRN)*, and *High-level Robustness Network (HRN)*, which were initially evaluated on the test fraction of the dataset, without the addition of noise. The validation and test accuracy obtained by the models are represented in table 3. The *LRN* achieved an accuracy of nearly 87%, whereas the other two models obtained slightly poorer results: 86.16% for the *MRN* and 85.86% for the *HRN*.

TABLE 3. Validation and Test accuracy for the three networks trained on different noise setups.

Network[a]	Validation Accuracy	Test Accuracy
LRN	86.90%	86.70%
MRN	86.15%	86.16%
HRN	85.24%	85.86%

[a]Train setup details presented in table 1.

To ascertain the effective level of robustness of each network to different conditions which may affect the quality of the input point clouds, a number of tests of increasing noise intensity were conducted for the three models. The test dataset was subject to the addition of Gaussian Noise of standard deviation that varied within the set: $\{0.05;0.1;0.2;0.3;0.5;0.7;1.0\}$. Figure 10 presents the test accuracy achieved by the *LRN*, the *MRN*, and the *HRN*. As a term of comparison, the red horizontal line in each chart corresponds to the test accuracy the network obtained when tested on noiseless data. As depicted in figure 10a, which corresponds to the *LRN* model, the test accuracy remains above 80% only for tests with a noise degree up to 0.1m. The performance declines severely for higher values of the standard deviation, reaching a minimum of 18.58%. On the other hand, the *MRN*, represented in figure 10b, shows an improved robustness, given that the average accuracy across all tests increases from 48.25% to 76.28%, relative to the previously described model. Considering that this network was trained on data with noise up to 0.5m of standard deviation, the average accuracy becomes 84.31% if computed only on the tests up to this noise degree. Finally, the chart of figure 10c depicts the results of the *HRN*, the one trained with the most severe noise. An even greater improvement can be perceived, with an average accuracy of nearly 81% and a minimum one of 70.68% for the worst case. Naturally, exposing this network to data with such extreme perturbation levels during the training process has been proved efficient in avoiding a substantial decline for tests with high noise degrees. However, it can also be verified that there is some performance loss on the lower end of noise intensity, which represents the most probable perturbation range that may affect input data. In fact, the LiDAR sensor has a typical accuracy of ± 2cm.[3] Considering the circumstances, the selected model was the *MRN*, given that it is robust to noise to a significant extent without compromising performance when tested on data without noise.

The accuracy gives insight on the overall performance of the model, but analysing the confusion matrix enables one to study the behaviour of the network regarding each class. Figure 11 presents the confusion matrices obtained when testing the model on three different noise degrees. The values are presented as a percentage of the total amount of instances of the corresponding predicted class (all values were rounded), which means that the diagonal cells represent the precision value for the respective class. Figure 11a depicts the results of the test in which no noise was added to the input data. Class precision is above 90% for all dock types, except for the 'S' class. The geometry of this layout is the least characteristic one of all the dock models, which might explain the lower precision. Class 'N' also presents a poorer result (67%), which indicates that the network is often assigning this label to instances that belong to other classes. Considering that this class' instances are characterized by partially represented platforms, it is expected that the boundary between labeling a dock as a specific layout and labeling it as undefined may not be very clear. Figures 11b and 11c depict the confusion matrices of tests that involved the addition of noise to the input data. For a standard deviation of 0.5m, represented in figure 11b, the precision of the model is not highly affected. As previously stated, classes 'S' and 'N' remain as those with a poorer performance, with a precision of 73% and 52%, respectively, versus a precision of above 90% for the other dock layouts. Class 'O' sees a decline in performance, achieving a precision of 75%. As noise increases, the features of a docking structure become more difficult to recognize and, thus, the structure may be mistaken for a different object. The confusion matrix corresponding to the test with the most extreme noise degree is shown in figure 11c. A significant decrease of the precision is verified on a general basis. Specifically, one of the greatest sources of error is the attribution of label 'T' to instances belonging to class 'Y'. The cause of this misclassification is likely the fact that the two dock types are the only layouts made up of three branches. Classes 'S', 'N', and 'O' follow the already discussed downward trend, reaching a precision as low as 30%. Nonetheless, three of the dock types ('Y', 'U', and 'L') remain above 80%, meaning that, despite the noise, when the network assigns one of these three labels it is correct most of the time.

[3]https://autonomoustuff.com/wp-content/uploads/2019/05/HDL-64E_S3_whitelabel.pdf

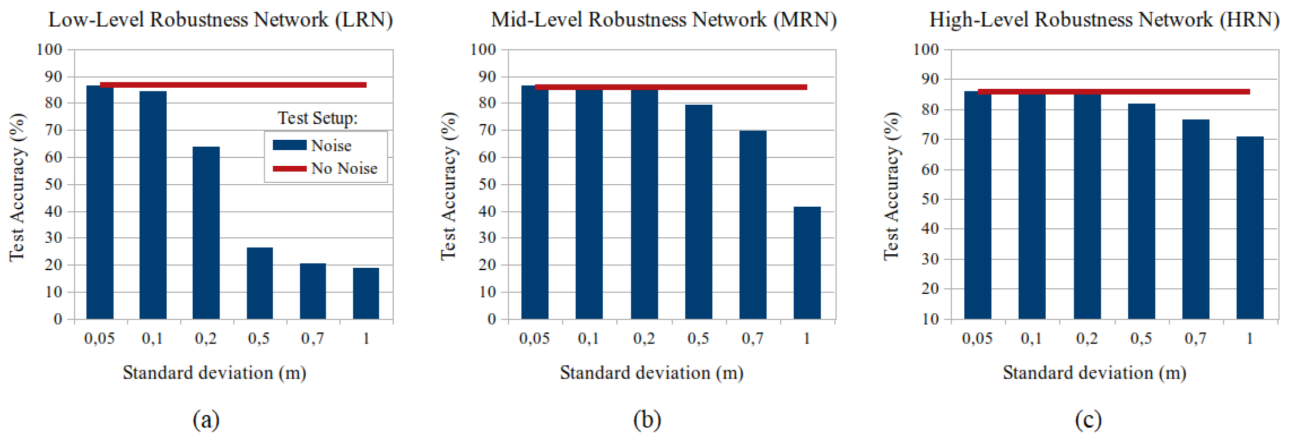
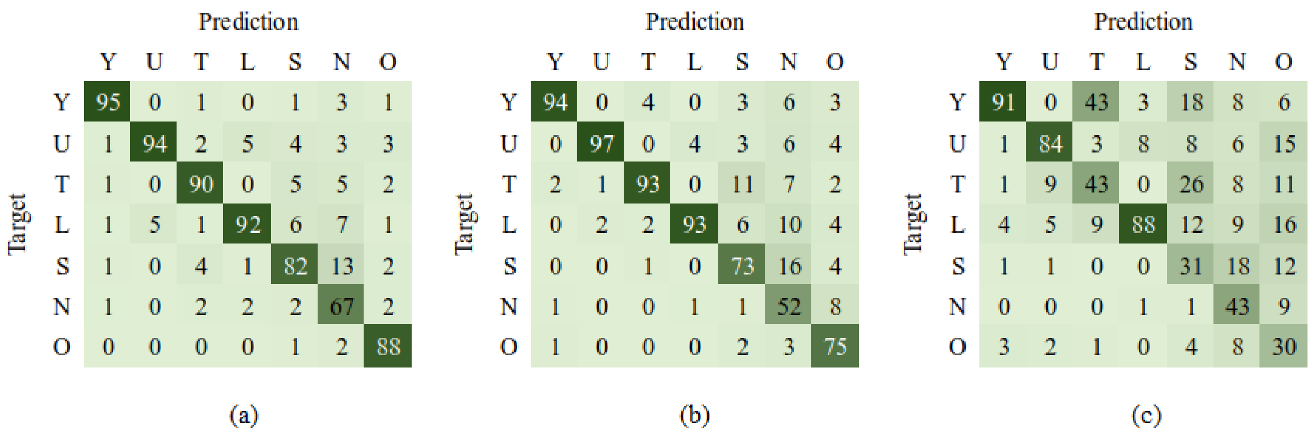


FIGURE 10. Robustness Evaluation of the networks: ***LRN*** (a), ***MRN*** (b), and ***HRN*** (c).



(a) Prediction

Target	Y	U	T	L	S	N	O
Y	95	0	1	0	1	3	1
U	1	94	2	5	4	3	3
T	1	0	90	0	5	5	2
L	1	5	1	92	6	7	1
S	1	0	4	1	82	13	2
N	1	0	2	2	2	67	2
O	0	0	0	0	1	2	88

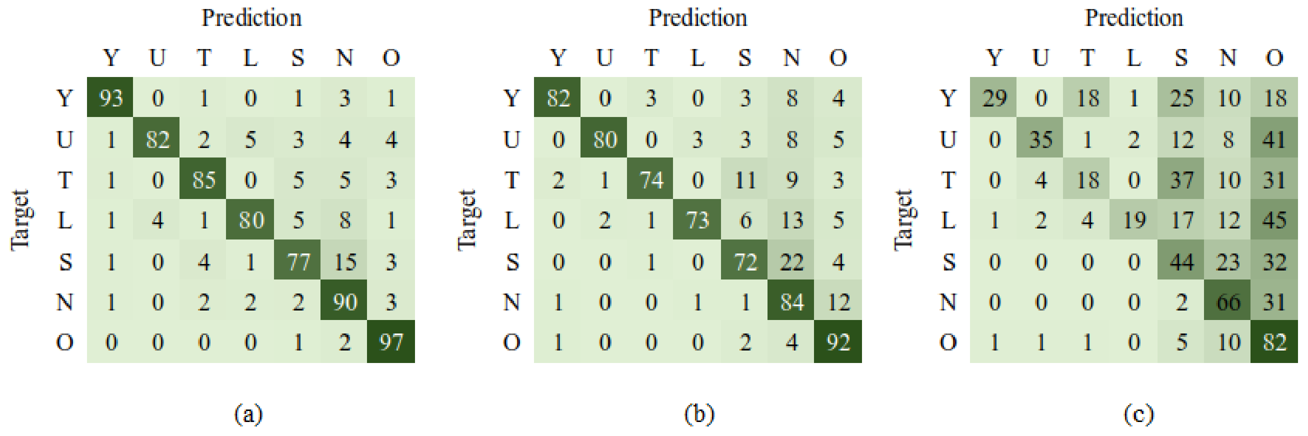
(b) Prediction

Target	Y	U	T	L	S	N	O
Y	94	0	4	0	3	6	3
U	0	97	0	4	3	6	4
T	2	1	93	0	11	7	2
L	0	2	2	93	6	10	4
S	0	0	1	0	73	16	4
N	1	0	0	1	1	52	8
O	1	0	0	0	2	3	75

(c) Prediction

Target	Y	U	T	L	S	N	O
Y	91	0	43	3	18	8	6
U	1	84	3	8	8	6	15
T	1	9	43	0	26	8	11
L	4	5	9	88	12	9	16
S	1	1	0	0	31	18	12
N	0	0	0	1	1	43	9
O	3	2	1	0	4	8	30

FIGURE 11. Confusion Matrices for tests of three different noise degrees: $\sigma = 0.0m$ (a), $\sigma = 0.5m$ (b) and $\sigma = 1.0m$ (c). The values are presented as a percentage of the total amount of instances which were assigned the corresponding column class (all values were rounded). The diagonal cells represent the **precision** of the respective class. The colouring of the cells of each matrix was defined in such a way that a darker colour corresponds to a higher value.



(a) Prediction

Target	Y	U	T	L	S	N	O
Y	93	0	1	0	1	3	1
U	1	82	2	5	3	4	4
T	1	0	85	0	5	5	3
L	1	4	1	80	5	8	1
S	1	0	4	1	77	15	3
N	1	0	2	2	2	90	3
O	0	0	0	0	1	2	97

(b) Prediction

Target	Y	U	T	L	S	N	O
Y	82	0	3	0	3	8	4
U	0	80	0	3	3	8	5
T	2	1	74	0	11	9	3
L	0	2	1	73	6	13	5
S	0	0	1	0	72	22	4
N	1	0	0	1	1	84	12
O	1	0	0	0	2	4	92

(c) Prediction

Target	Y	U	T	L	S	N	O
Y	29	0	18	1	25	10	18
U	0	35	1	2	12	8	41
T	0	4	18	0	37	10	31
L	1	2	4	19	17	12	45
S	0	0	0	0	44	23	32
N	0	0	0	0	2	66	31
O	1	1	1	0	5	10	82

FIGURE 12. Confusion Matrices for tests of three different noise degrees: $\sigma = 0.0m$ (a), $\sigma = 0.5m$ (b) and $\sigma = 1.0m$ (c). The values are presented as a percentage of the total amount of instances of the corresponding actual class (all values were rounded). The diagonal cells represent the **recall** of the respective class. The colouring of the cells of each matrix was defined in such a way that a darker colour corresponds to a higher value.

Precision is a measure that does not take into account the False Negatives of each class and, thus, it may provide an overly optimistic view of the network's behaviour. To assess how reliable this analysis is, one should take into account other measures, such as the recall. Figure 12 illustrates the confusion matrices of the same tests presented in figure 11,

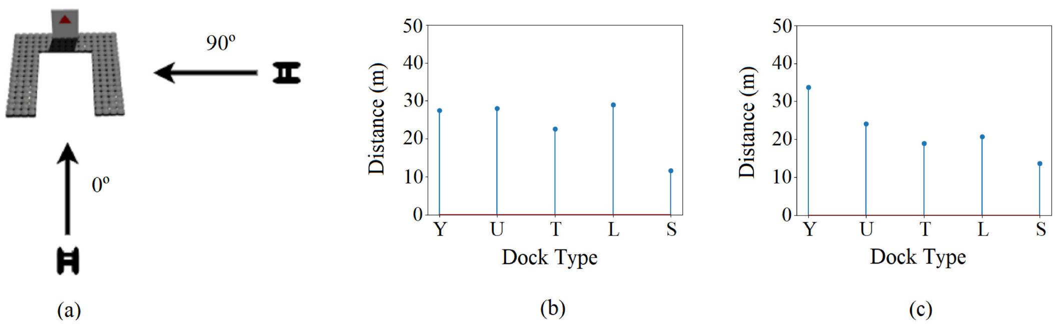


FIGURE 13. Orientation Scheme (a). Maximum distance at which the ASV can detect the type of dock with a minimum of 50% certainty from a front-facing approach (0°) (b) and sideways approach (90°) (c).

but the normalization of the values took into account the total of instances of each actual class. Therefore, the diagonal cells represent the recall of the corresponding class. Figure 12a presents the confusion matrix of the noiseless test. The class with the highest recall corresponds to the dock type 'Y', while the lowest is that of type 'S'. These results can be explained by the fact that the geometry of the platforms of these two classes are the most and the least characteristic ones, respectively. As a matter of fact, the greatest cause of error corresponds to the case in which instances belonging to type 'S' are instead classified as 'N'. When the system cannot perceive the entirety of the structure's base and, therefore, cannot determine its type, such instance should be classified as 'N'. Nonetheless, a few visible points of the platform may induce a prediction of type 'S'. Regarding the identification of the False Positive instances which may be produced by the detection module, the classifier shows a great performance, with a recall of about 97% for the 'O' class. Figures 12b and 12c illustrate the confusion matrices of two noise tests (0.5m and 1.0m). Overall, there is an increase of incorrect predictions, such as instances belonging to type 'Y' being classified as type 'T', as explained during the precision analysis, and the class 'S' being assigned to instances of other types, which is also likely due to partial observations of the horizontal platform. Another significant variation is the shift of incorrect predictions towards the False Positive class ('O'), especially for the test with higher noise. This analysis showed that, regarding the relationship between the instance distribution among the layouts and the correct label assignments, the precision assessment was indeed providing an optimistic perspective of the network's performance. On the other hand, the percentage of instances belonging to the classes 'S', 'N', and 'O' which are correctly classified is shown to be higher than presumed in the previous analysis. These conclusions show that it is important to consider several measures when evaluating a model.

5) INTEGRATION AND TESTING IN A SIMULATED ENVIRONMENT

Thus far, the classifier was evaluated on the test fraction of the dataset. To test the system in a simulated maritime environment, with all the integrated components, that is, detection and classification, is to provide a more realistic perspective of its behaviour in a dynamic setup. The present section will disclose the results of the tests conducted in the simulator Gazebo, which were performed on a device with the following specifications:

- **CPU** - Intel® Core^{TM} i7-5500U Quad-core 2.40GHz;
- **GPU** - NVIDIA GeForce 920M 2GB DDR3;
- **RAM** - 8GB.

For an intermediate assessment of the potential of the synthetic-to-real domain adaptation methodology, it is necessary to replicate the conditions of an authentic maritime environment. Table 4 presents the simulated environmental conditions (waves) under which the following tests were conducted. Figure 4 illustrates an example of the simulated scenario in Gazebo.

TABLE 4. Environmental Constraints considered for the testing of the system.

Wave Component	Amplitude (m)	Period (s)	Direction ($\vec{x}$,$\vec{y}$)
1	0.06	12.6	(-1.0,0.0)
2	0.04	3.7	(-0.7,0.7)
3	0.03	6.3	(0.7,0.7)

The objective of the first test was to measure the range in which the system was capable of correctly identifying the type of the detected docking platform, as well as analysing the effect of distinct perspectives by conducting trajectories from different approach angles. Each of the five dock models was placed in the simulated maritime scenario, in Gazebo, and the ASV moved towards the structure from a distance of 100m. Two different directions were considered for the trajectory: a front-facing approach (angle of 0°) and a sideways one (angle of 90°), as depicted in figure 13a. Figures 13b and 13c depict for each dock type the distance at which the system can correctly classify the structure with a minimum certainty of 50%. Overall, this distance is nearly 24m for a front-facing approach, and slightly above 22m for a sideways trajectory. On average, the results presented here are consistent with those previously discussed, in particular, the ones suggesting the type 'Y' to be the easiest layout to identify and the type 'S' the most difficult one. When analysing the results of each of the two subfigures (13b and 13c), one finds differences within the outcomes of each class. Classes 'U', 'T', and 'L' present a decrease of the detection distance between the 0° and 90° tests. This tendency might be explained by the fact that the three layouts look quite identical at great distance from a side-view. On the other hand, types 'Y' and 'S' can be correctly classified farther for an angle of approach of 90° than for one of 0°, with a range increase of 7m and 2m, respectively. This variation is also easily justifiable. If observed from a side perspective, all of the branches of the base of a 'Y' dock can be visible, which does not apply to a front-facing view. When a dock of type 'S' is perceived from the side, one may confirm that the platform has only one branch. Class 'S' presents a considerably lower detecting distance when compared to the other classes. As previously mentioned, this layout is the less characteristic one, which might explain this result. Taking into account these simulation outcomes, other conclusions can be made. The average distance at which the classifier can correctly identify the type of dock is approximately 24m. However, the collected dataset contains instances featuring structures that distance up to 40m from the vehicle. In the farther range, any horizontal platform might not be entirely visible and may resemble the 'S' type, even if belonging to other classes. Through such learning process, the network may mistake docks of type 'S', even at a short distance, for structures of other layouts.

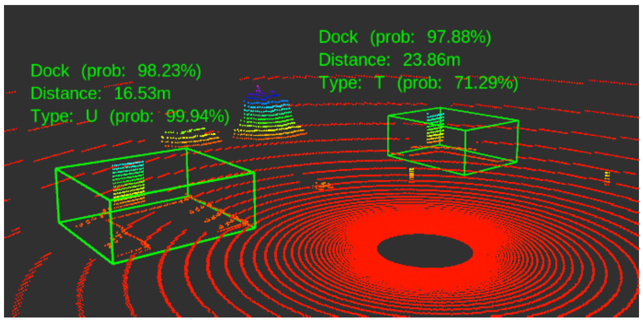


FIGURE 14. **Display of the output of the system: the certainty of the detection of the dock, the distance between the structure and the ASV, the identified class, and corresponding probability.**

The classification module was integrated in the recognition system along with the detection component. A dynamic test was carried out in a simulated scenario. The ASV moved along a straightforward path, passing by different types of docking platforms from those considered in the development of the classifier. Other types of objects were also scattered through the scenario, such as floating buoys and other boats. The system continuously searched through the surrounding environment and labeled any detected dock, as illustrated in figure 14. The results were recorded in video.[4]

[4]https://bit.ly/2B13G14

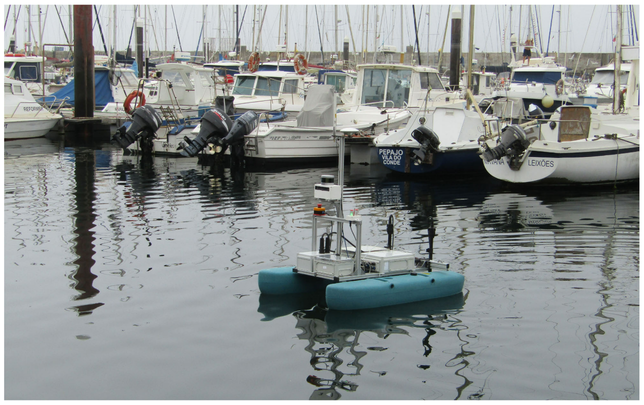


FIGURE 15. **Zarco ASV developed by INESC TEC [29], operating in the Port of Leixões (Porto, Portugal).**

B. REAL DOMAIN ADAPTATION

A perception system developed on a basis of synthetically generated data is often not prepared to function in a real environment with all of its added constraints and challenges. As described before, decreasing domain bias during the training process was a top priority in this research work. Several techniques were adopted to implement this strategy, such as transforming the input data through scaling, down-sampling, and adding Gaussian noise. Therefore, the cost of the adaptation process of the system for an authentic maritime environment was greatly reduced, both in data acquisition expenses and in additional computational effort required to fine-tune the network. The present section depicts the process of domain adaptation and evaluation.

1) DATA ACQUISITION IN SEAPORT

To collect data for the fine-tuning and testing of the network, the team went to the Port of Leixões (Porto, Portugal), which is depicted in figure 15. This commercial harbour provided an appropriate setup, given that it covered the main challenges commonly posed by real maritime environments: high density and diversity of objects, and lack of control over weather conditions. At the time of the operation, the seaport was under a clear sky, but there was moderate wind which caused some wavelets. The conducted tests made use of the Zarco ASV developed by INESC TEC [29], also illustrated in figure 15. This vehicle is equipped with a *VLP-16* LiDAR and a stereo camera, as well as other sensors. As already mentioned, the wavelength of the selected LiDAR sensor is 905 nm. A study by Wojtanowski *et al.* [30] found that this type of sensor suffers a performance degradation significantly lower than a 1550 nm sensor in case of harsh environmental conditions such as heavy rain and fog. The utilization of hardware with these characteristics contributes to an increased robustness of the system.

Due to logistic reasons, only two of the five dock models were built for this process. The selected layouts were types 'U' and 'T'. The intermediate layout complexity of these two

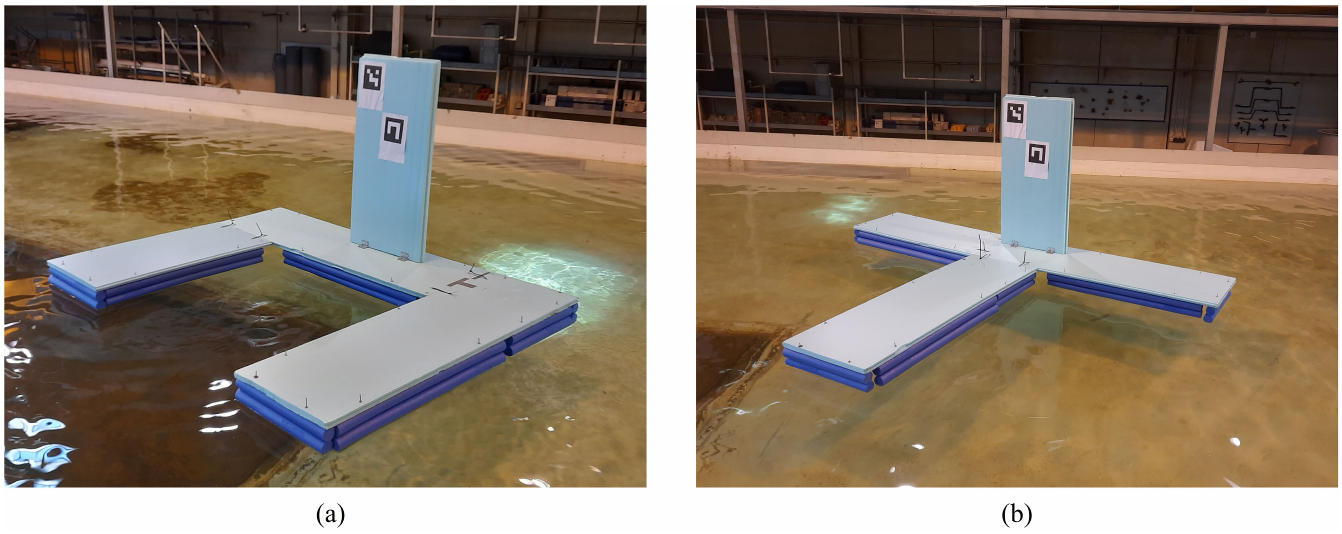


FIGURE 16. Docking structures of type 'U' (a) and 'T' (b), built from extruded polystyrene foam (XPS). Images were taken in the wave tank of the Civil Engineering Department of the Faculty of Engineering of the University of Porto.

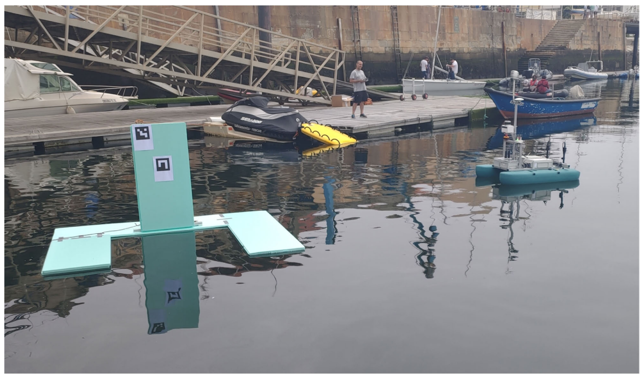


FIGURE 17. Real commercial harbour (Port of Leixões in Porto, Portugal), featuring a floating docking platform and the Zarco ASV.

models ensures they generally represent the most common dock layouts, unlike the type 'S', for example, which is too simple, but it also introduces an increase in the difficulty level of the recognition process, given that they're not as characteristic as layout 'Y', for example, and are, therefore, harder to recognize. The two docking structures were built from extruded polystyrene foam (XPS) with maximum dimensions 2.6mx3.3mx1.5m and a 15cm high platform, as illustrated in figures 16a and 16b. The images were captured in the wave tank of the Civil Engineering Department of the Faculty of Engineering of the University of Porto. Each of the docking structures was placed in the port, as depicted in figure 17, and the ASV moved through the surroundings, gathering data. Unlike the tests conducted in simulation, in which the vehicle moved through a deterministic trajectory, the path performed during this task was arbitrary.

2) LOW-COST NETWORK FINE-TUNING

The fine-tuning of a neural network should be conducted on a different dataset from the one utilized during the main training process, but usually the new data belongs to the same domain. In this research work, however, the objective was to apply this technique between two different domains: synthetic and real. As observed in figure 18, an authentic maritime environment does differ quite a lot from a simulated one. Dissimilarities observed in the real scene include a very high cluster density, no laser reflection from the water surface, and a few sparser areas due to the utilization of a lower resolution LiDAR (*VLP-16* vs. *HDL-64E*). Some of these differences pose great challenges for perception systems. In this work, the main training process took into account strategies to approximate features between domains and thus overcome these challenges. Due to the high cluster density, only the classification module of the system was subject to this adaptation process. The fine-tuning of the network followed a very identical process to the main training instance, but on a reduced scale. From the data acquisition described above, a small set of training data was selected for this process, of which 36 point clouds were utilized for the actual training of the network and 12 for validation purposes. Similarly to the core training method, the layer freezing strategy was employed during this domain adaptation to preserve the already acquired predicting abilities of the network. The fine-tuning is intended to ensure the network is able to process features from a different domain, not to rebuild its recognition abilities from scratch, hence, the importance of an appropriate strategy to freeze layers and adjust the parameters' update. The network was then retrained for 5 epochs, a procedure of 12 seconds on a device with the previously described characteristics, which represents a very low computational

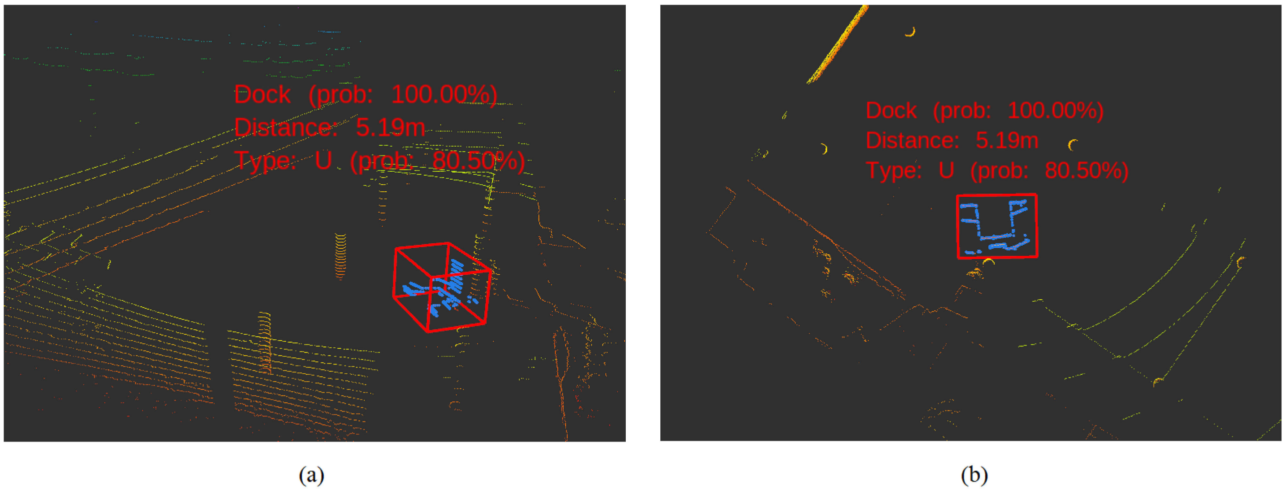


FIGURE 18. **Recognition of dock of type 'U'. The structure is represented by blue points and surrounded by a red bounding box. The result of the classification process is presented in the coloured text and includes the certainty of the prediction. A third-person view is presented in (a) and a top-down view in (b).**

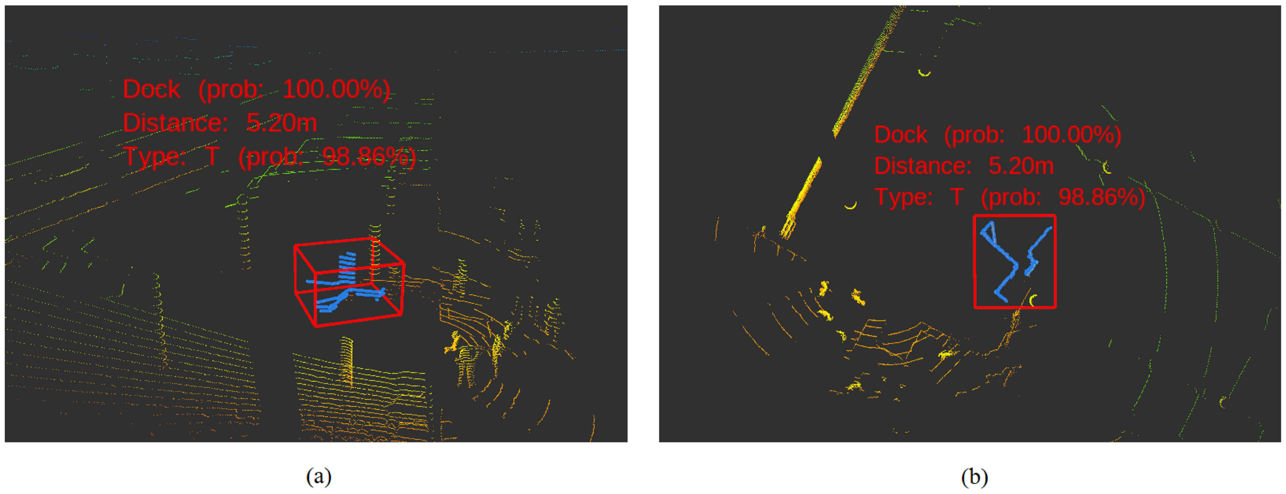


FIGURE 19. **Recognition of dock of type 'T'. The structure is represented by blue points and surrounded by a red bounding box. The result of the classification process is presented in the coloured text and includes the certainty of the prediction. A third-person view is presented in (a) and a top-down view in (b).**

expense. The train and validation accuracies achieved by the model were both equal to 91.67%, which is higher than the average accuracy of 86.16% obtained during the main training process. The utilization of only two docks out of the five models may be an influencing factor, but the increase of the accuracy may also indicate that, despite the dissimilarities between the domains from which data was gathered in either situation, the model was able to improve its prediction ability on top of that previously acquired. Given the small scale of the validation set, however, the tests depicted in the following section constitute a more appropriate assessment of the refined model.

3) PERCEPTION TESTS IN A REAL COMMERCIAL SEAPORT

To validate the success of the adaptation process, a dynamic test was conducted for each of the two docks, in which a subtrajectory of approximately 45 seconds of the vehicle's path was supplied to the model frame-by-frame and the output predictions were computed in real time. Figures 18 and 19 show examples of the results of the conducted evaluation on the gathered LiDAR point clouds. The docking platform is represented in each image by blue points and surrounded by a red bounding box. Figures 18a and 19a represent a third-person view, whereas figures 18b and 19b depict a top-down view of the scene.

TABLE 5. **Percentage of instances predicted as each column's corresponding class, per actual type of structure.**

		Prediction						
		Y	U	T	L	S	N	O
Target	U	3.88%	**84.47%**	0.00%	0.00%	10.68%	0.00%	0.97%
	T	2.67%	0.00%	**96.29%**	0.00%	0.43%	0.00%	0.61%

Videos of each test were recorded ('U' [5] and 'T' [6]). Table 5 presents for each of the two dock types the percentage of iterations predicted as the column's corresponding class. For the test conducted on the 'U' structure, the system correctly predicted the type of dock 84.47% of the time. This value represents both the accuracy and recall for this class. Figure 12a presented a recall of approximately 82% for the 'U' type in the simulation tests without noise, which is a bit lower than the result obtained in the test at hand. Among the incorrect predictions, class 'S' appears to be the most frequently assigned to the dock. This trend is due to the fact that if the vehicle is at a short distance from the dock it does not fully perceive the base of the structure. Therefore, the extracted cloud resembles the 'S' type. Regarding the test carried out on the 'T' docking platform, the obtained recall was 96.29%, which is a great deal higher than that obtained for a 'U' dock and than the 85% recall depicted in figure 12a for the simulation tests. The only considerable source of error corresponds to the assignment of class 'Y', given that the other types present residual error. The platforms of both layouts 'T' and 'Y' are composed of three branches, which may explain this trending misclassification.

There is a considerable difference between the results achieved for the 'U' and the 'T' structures. An arbitrary trajectory implies an unequal acquisition of testing data for the classes. Thus, the quality with which data represents the docks is greatly influenced by the path of the vehicle, namely regarding its orientation and distance to the structure. For the 'T' dock, the data collected during the vehicle's trajectory represented the structure without occlusions and at a favourable distance, as illustrated in the video, which explains the higher accuracy. For either case, the obtained results were better than those achieved in simulation, with an average accuracy of approximately 90%, although a straight comparison cannot be made, given the different testing conditions. The real setting test presented disadvantages, such as the high cluster density surrounding the vehicle and the structure, as well as the decreased density of points in the cloud caused by the utilization of a LiDAR sensor with a lower resolution. Moreover, the wind and waves caused the platform to move, thus bringing an instability factor to the perception system. A high accuracy under these conditions (lower quality sensors and environmental constraints) demonstrates that the domain adaptation methodology was successful in developing a system able to perform in harsher environments and with lower cost hardware. The utilization of a LiDAR sensor also enabled the system to be unaffected by the ambient light variations.

A fundamental characteristic of a perception system developed for robotics applications is the guarantee of real-time operation. To verify if this objective was fulfilled, the average duration of each iteration was calculated throughout the tests represented in the videos. The recognition system as a whole accounted for a total of 120ms per iteration, which corresponds to an operation rate of approximately 8Hz. Considering the sort of vehicle for which this work was developed (low speed - maximum of 2m/s[7]) and the nature of the task, the system has accomplished the objective of real-time operation. This speed was achieved without the need for great computational power due to the many strategies took into consideration throughout the development of the system, such as the pre-processing steps and the adoption of a lightweight network.

To the best of our knowledge, this is the first work that addresses the classification of docking structures. Even though a direct comparison cannot be made with methods that only handle the detection of such platforms, a brief discussion of related aspects can be presented. In [13], Esposito and Graves presented a detection method based on point clouds that took over 10 seconds per iteration, whereas our system works at a rate of 8Hz as presented above. Moreover, our method was evaluated on data collected from a moving vehicle and not from fixed sensors. Regarding participations in the Maritime Robot X Challenge, Lee *et al.* [14] used a 2D LiDAR based approach, which meant they had to resort to previous knowledge of the structures' geometry to be able to locate it. This issue demonstrates the relevance of using a 3D LiDAR in our research, as it enabled us to easily extract the 3D position of the structure. On the other hand, Huang *et al.* [15] adopted a 3D LiDAR approach, but did not address the problem of sparser data in real environments and, thus, suffered a considerable decline in performance when deploying the system from simulated to real scenarios. On the contrary, our data augmentation techniques were successful in avoiding this outcome. Finally, Leite *et al.* [16] proposed a detection method that required the structure to have a specific shape and was only evaluated in simulated environments. Furthermore, it was based on a template matching approach, which is not suitable for cluttered scenarios.

IV. CONCLUSION

This article proposed a perception system based on a Convolutional Neural Network for the recognition of a docking

[5]https://bit.ly/3lRRZMd

[6]https://bit.ly/2J2DzdW

[7]http://pisces.inesctec.pt/2017/03/08/zarco-gama-asvs/

structure from 3D data, thus advancing the docking process of an ASV. Unlike current methods, this work did not focus strictly on the detection of a dock, but aimed to provide further knowledge about the structure, such as the identification of its layout. A 3D real-time perception system was developed based on a volumetric Convolutional Neural Network with a low number of parameters. Different techniques were used to guarantee system robustness to harsh and fluctuating environmental conditions. A synthetic-to-real domain adaptation methodology was adopted to accelerate the training process and overcome the unavailability of datasets within this scope. The network was trained over a large corpus of synthetically generated data and fine-tuned on a small set of data collected from a real maritime environment. A few strategies, such as downsampling and adding Gaussian noise, allowed to replicate constraints found in authentic maritime environments and thus decrease domain bias.

Several experiments were conducted in a 3D simulated environment and in a real commercial seaport (Port of Leixões in Porto, Portugal). Results showed robustness to environmental conditions of different degrees, with an average accuracy of 84.31% obtained in simulation tests. The classifier was also evaluated in an authentic harbour, achieving an average classification accuracy above 90% even with lower resolution sensors. The integration of the detection and classification modules formed a system that worked at a rate of 8Hz, thus guaranteeing real-time operation for the targeted type of vehicle. In conclusion, the proposed synthetic-to-real domain adaptation methodology resulted in a fast, precise, and robust perception system without requiring the acquisition of large amounts of data nor the access to great computational resources.

REFERENCES

[1] A. Leite, A. Pinto, and A. Matos, "A safety monitoring model for a faulty mobile robot," *Robotics*, vol. 7, no. 3, p. 32, Jun. 2018.

[2] A. M. Pinto and A. C. Matos, "MARESye: A hybrid imaging system for underwater robotic applications," *Inf. Fusion*, vol. 55, pp. 16–29, Mar. 2020.

[3] D. F. Campos, A. Matos, and A. M. Pinto, "An adaptive velocity obstacle avoidance algorithm for autonomous surface vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Macau, China, Nov. 2019, pp. 8089–8096.

[4] R. Silva, P. Leite, D. Campos, and A. M. Pinto, "Hybrid approach to estimate a collision-free velocity for autonomous surface vehicles," in *Proc. IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Porto, Portugal, Apr. 2019, pp. 1–6.

[5] R.-M. Darbra and J. Casal, "Historical analysis of accidents in seaports," *Saf. Sci.*, vol. 42, no. 2, pp. 85–98, Feb. 2004.

[6] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 1440–1448.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 779–788.

[8] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 77–85.

[9] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Hamburg, Germany, Sep. 2015, pp. 922–928.

[10] S. Zhi, Y. Liu, X. Li, and Y. Guo, "Toward real-time 3D object recognition: A lightweight volumetric CNN framework using multitask learning," *Comput. Graph.*, vol. 71, pp. 199–207, Apr. 2018.

[11] G. Riegler, A. O. Ulusoy, and A. Geiger, "OctNet: Learning deep 3D representations at high resolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 6620–6629.

[12] S. Bera and V. K. Shrivastava, "Analysis of various optimizers on deep convolutional neural network model in the application of hyperspectral remote sensing image classification," *Int. J. Remote Sens.*, vol. 41, no. 7, pp. 2664–2683, Apr. 2020.

[13] J. M. Esposito and M. Graves, "An algorithm to identify docking locations for autonomous surface vessels from 3-D LiDAR scans," in *Proc. IEEE Int. Conf. Technol. Practical Robot Appl. (TePRA)*, Woburn, MA, USA, Apr. 2014, pp. 1–6.

[14] J. Lee, J. Woo, and N. Kim, "Vision and 2D LiDAR based autonomous surface vehicle docking for identify symbols and dock task in 2016 maritime RobotX challenge," in *Proc. IEEE OES Int. Symp. Underwater Technol. (UT)*, Feb. 2017, pp. 1–5.

[15] Y.-W. Huang *et al.*, "Team NCTU : Toward AI-driving for autonomous surface vehicles—From Duckietown to RobotX," Oct. 2019 *arXiv:1910.14540*. [Online]. Available: https://arxiv.org/abs/1910.14540

[16] P. Leite, R. Silva, A. Matos, and A. M. Pinto, "An hierarchical architecture for docking autonomous surface vehicles," in *Proc. 19th IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Porto, Portugal, Apr. 2019, pp. 1–6.

[17] M. I. Pereira, P. N. Leite, and A. M. Pinto, "Detecting docking-based structures for persistent ASVs using a volumetric neural network," presented at the MTS/IEEE Global OCEANS, 2020.

[18] A. Atapour-Abarghouei and T. P. Breckon, "Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, Jun. 2018, pp. 2800–2810.

[19] Y. Zhang, P. David, H. Foroosh, and B. Gong, "A curriculum domain adaptation approach to the semantic segmentation of urban scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 1823–1841, Aug. 2020.

[20] X. Gu, Y. Guo, F. Deligianni, and G.-Z. Yang, "Coupled real-synthetic domain adaptation for real-world deep depth enhancement," *IEEE Trans. Image Process.*, vol. 29, pp. 6343–6356, 2020.

[21] D. F. Campos, A. Matos, and A. M. Pinto, "Multi-domain inspection of offshore wind farms using an autonomous surface vehicle," *Social Netw. Appl. Sci.*, vol. 3, no. 4, Apr. 2021, Art. no. 455.

[22] M. G. H. Omran, A. P. Engelbrecht, and A. Salman, "An overview of clustering methods," *Intell. Data Anal.*, vol. 11, no. 6, pp. 583–605, Nov. 2007.

[23] E. T. Jaynes, *Probability Theory: The Logic of Science*, G. L. Bretthorst, Ed. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[24] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1912–1920.

[25] B. Pang, E. Nijkamp, and Y. N. Wu, "Deep learning with tensorflow: A review," *J. Educ. Behav. Stat.*, vol. 45, no. 2, pp. 227–248, 2020.

[26] V. Subramanian, *Deep Learning With PyTorch : A Practical Approach to Building Neural Network Models Using PyTorch*. Birmingham, U.K.: Packt Publishing, 2018.

[27] J. Brownlee, *Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python*. Vermont, VIC, Australia: Machine Learning Mastery, 2019.

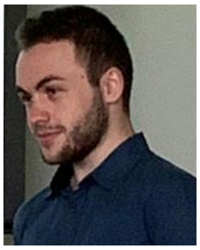
[28] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manage.*, vol. 45, no. 4, pp. 427–437, Jul. 2009.

[29] D. F. Campos, A. Matos, and A. M. Pinto, "Multi-domain mapping for offshore asset inspection using an autonomous surface vehicle," in *Proc. IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Apr. 2020, pp. 221–226.

[30] J. Wojtanowski, M. Zygmunt, M. Kaszczuk, Z. Mierczyk, and M. Muzal, "Comparison of 905 nm and 1550 nm semiconductor laser rangefinders' performance deterioration due to adverse environmental conditions," *Opto-Electron. Rev.*, vol. 22, no. 3, pp. 183–190, Jan. 2014.

MARIA INÊS PEREIRA received the M.Sc. degree in electrical and computer engineering from the Faculty of Engineering, University of Porto (FEUP), Portugal, in 2020. She is currently pursuing the Ph.D. degree in electrical and computer engineering, while conducting research with the Centre for Robotics and Autonomous Systems, INESC TEC. Her main research interests include maritime robotics, deep learning, and perception systems.

RAFAEL MARQUES CLARO received the M.Sc. degree in electrical and computer engineering from the Faculty of Engineering, University of Porto, Portugal, in 2020. He is currently pursuing the Ph.D. degree in electrical and computer engineering. He is currently a Researcher with the Centre for Robotics and Autonomous Systems, INESC TEC.



PEDRO NUNO LEITE was born in Porto, Portugal, in 1996. He received the M.Sc. degree in electrical and computer engineering from the Faculty of Engineering, University of Porto (FEUP), in 2019, where he is currently pursuing the Ph.D. degree. He is currently working with the CRAS, INESC TEC, as an Assistant Researcher. His current research interests include underwater robotics and perception techniques, computer vision, and deep learning.

ANDRY MAYKOL PINTO received the Ph.D. degree in electrical and computer engineering from the Faculty of Engineering, University of Porto, Portugal, in 2014. He is currently an Assistant Professor with the Faculty of Engineering, University of Porto, and a Senior Researcher with the Centre for Robotics and Autonomous Systems, INESC TEC. He is the Principal Investigator of national and international research and development projects related to robotic-based operation and maintenance (O&M) activities for offshore infrastructures. His main research interests include multi-domain perception, underwater imaging, artificial intelligence, and mobile robotics.