Mälardalen University
School of Innovation Design and Engineering
Västerås, Sweden

Thesis for the Degree of Master of Science in Engineering 120 credits

AUTOMATED ANNOTATION SCHEME FOR EXTENDING BOUNDING BOXES REPRESENTATION TO DETECT SHIP LOCATIONS

Oksana Havryliuk
ohk22002@student.mdh.se

Examiner: Mobyen Uddin Ahmed
Mälardalen University, Västerås, Sweden

Supervisors: Shahina Begum
Mälardalen University, Västerås, Sweden

Company supervisor: Pontus Boström
Groke Technologies, Turku, Finland

September 25, 2023

Abstract

Bounding boxes often provide limited information about the shape and location of an object on an image. Their limitations lie in their reduced ability to correctly represent objects that have complex shapes or are located at an angle. Related works introduce new object representations that include segmentation masks, keypoints, polylines, and regions and are effective in capturing complex shapes and attributes of an object, but lack computational efficiency for real-time applications and require annotated datasets. The aim of the thesis is to propose an approach to extend bounding box representation to include attributes of interest at a low computational cost. Moreover, the approach aims to automatically transform existing bounding boxes into a new object representation. As a result, the thesis is potentially beneficial for real-time applications that need a complex object representation at a small cost, as well as to create new datasets from existing bounding boxes data or to detect faulty bounding boxes. The approach consists of using a segmentation model to compose a new object representation from a bounding box. The task of object detection is essential in computer vision applications, such as autonomous driving, surveillance, and robotics. The traditional method of representing objects using bounding boxes has limitations in capturing complex shapes and attributes of an object. Therefore, the motivation for this thesis is to propose a low computational cost approach to extend bounding box representation to include attributes of interest. To address this problem, the proposed approach involves using a segmentation model to compose a new object representation from a bounding box. The segmentation model generates a mask for the object, which can be used to extract more detailed features, such as object contours, keypoints, and regions. The approach aims to automatically transform existing bounding boxes into a new object representation, which is potentially beneficial for real-time applications that require a complex object representation at a small cost, as well as to create new datasets from existing bounding box data or to detect faulty bounding boxes. In summary, the approach proposed in this thesis provides an efficient and automated way to improve object representation in computer vision tasks. The experimental results show that the proposed approach achieves better object detection accuracy compared to the traditional bounding box representation, especially for objects with complex shapes and attributes. The approach also has the potential to improve the efficiency of real-time applications that require a complex object representation. Overall, this thesis contributes to the development of more accurate and efficient computer vision systems for various applications.

Table of Contents

1. Introduction

This thesis aims to focus on a Computer Vision concept of bounding boxes to locate ship on the image in the maritime environment using Machine Learning (ML) methods. One of the main problems with using bounding boxes for object detection is that they can be sensitive to the scale and aspect ratio of the objects in an image. Objects that are smaller or have a different aspect ratio than the bounding box can be difficult to detect accurately. Additionally, bounding boxes can be sensitive to the location and pose of objects in an image, which can make it difficult to detect objects that are partially occluded or rotated. These problems occur in a maritime environment as well. When detecting bounding boxes of ship, one will face issues when the ship is close and at an angle or when the ship itself is rolling in the waves. Then, the bounding box will not approximate the waterline of the object and objects will seem closer than they are. Moreover, the bounding box can contain more sea than the actual boat location. Hence, a more accurate description of the ship's position is needed in certain situations.

The project will be done in cooperation with Groke technologies, a Finnish maritime company that develops an awareness system for ships that is intended to help crews make better decisions [1]. The suggested approach will be beneficial for maritime companies who want to improve the object positioning in a sea environment as well as anyone who is interested in improving localisation of an object on the image or video. Furthermore, the thesis will suit a need for automatic correction and can be used to reduce re-annotation costs.

Several works suggest to use improved and more sophisticated representation of objects [2], [3], [4], [5], [6], [7]. Common approaches include the usage of segmentation masks that can give a pixel-wise classification of objects within an image [4], [8] or keypoints that define the borders of objects [9], [10]. Other approaches focus on improving the localization of the bounding box and producing less error at the borders [2], [3], [11], [12], [13]. One of the main limitations of existing approaches is a lack of **computational efficiency**. Since often the use case of object detection is in real or near real life applications it is important to keep the detection fast and with minimal computational resources. Also, training time and cost of such a ML detection model are often quite high for existing state-of-art shapes. The second challenge is the **need for an annotated dataset** with a new representation. Due to the fact that there is a lack of publicly available datasets, manual annotation is required to obtain one for training and testing of a ML model. Annotation or re-annotation is a very costly process.

The hypothesis of the thesis is that the development and integration of an automated semantic segmentation model for ship contour identification, along with the creation of a new bounding box representation capable of accurately delineating the ship's waterline, will result in a significantly improved ship detection process within images. This innovative approach is anticipated to not only reduce the costs associated with dataset annotation but also outperform traditional bounding box methods when assessed using a deep neural network-based predictive model. The primary objectives of the thesis contain the development of this automated semantic segmentation model, the formulation of the waterline-based bounding box representation, the selection of an optimal annotation scheme for enhanced efficiency, and the establishment of a comprehensive evaluation framework for assessing the performance of the predictive model.

The task of accurately detecting and locating ships in a maritime environment is of significant importance in various fields such as maritime security, surveillance, and transportation. Traditional bounding box approaches have limitations in accurately detecting and locating ships due to issues such as sensitivity to scale, aspect ratio, location, and pose. The evaluation of this new approach against the traditional bounding box approach provides an objective comparison of effectiveness and establishes the potential for improved accuracy in ship detection and location. The results of this study can contribute to enhancing maritime safety and security, as well as advancing the field of Computer Vision and ML in object detection and representation.

2. Background

The maritime industry is a critical part of global commerce, with millions of tons of cargo being transported across oceans every day. However, with increasing demand and growing concerns about safety, the need for innovative solutions to enhance the efficiency and safety of maritime transport has become more pressing than ever. One promising approach is the use of object detection technology to provide improved semi-autonomous vessels. Object detection technology uses computer vision algorithms to identify and track objects in real time, enabling vessels to navigate safely and avoid collisions with other ships, buoys, and obstacles. This technology has the potential to significantly reduce the risk of accidents and improve efficiency by allowing vessels to operate at higher speeds and with greater precision.

However, implementing object detection technology in the maritime industry poses some difficulties. One of the challenges of using object detection technology in the maritime industry is the issue of inaccurate bounding boxes. In object detection, bounding boxes are used to enclose the detected object and provide information about its location and size [14]. However, in the marine environment, ships are often irregularly shaped, which can make it difficult to accurately define their boundaries using traditional bounding box representations. In addition to improving the accuracy of object detection, these new ship representations can also have other benefits. For example, they can be used to develop more advanced collision avoidance algorithms that take into account the unique shape and size of each ship.

The thesis is done in cooperation with Groke Technologies which is a Finnish maritime company. Groke Technologies' main mission is to create solutions that will reduce navigation crew stress and ensure peace of mind during operation even in the most challenging situations on board different vessels. Groke's initial solution development is tailored to meet the needs of the Japanese Maritime Industry which is supported by their strategic investor Mitsubishi Corporation. Japan is only the first market and global markets are to follow in upcoming years. Groke's technical capabilities are a combination of a diverse range of expertise coming from different industry backgrounds firmly harnessed by their Founding members who had been leading the Autonomous technology development globally [1].

Object detection is a vital technology that allows computers to identify and locate objects in images or videos. It has a wide range of applications, from autonomous driving to security surveillance. One of the key components of object detection is the use of bounding boxes, which are rectangular shapes that enclose the detected object and provide information about its location and size [15][16][17].

Another common approach is semantic segmentation. Semantic segmentation takes object detection to the next level by not only detecting objects but also dividing them into meaningful segments based on their visual characteristics [18]. This technique allows computers to understand the different parts of an object and their relationships to each other. For example, in maritime applications, semantic segmentation can be used to identify sea, sky, and different ships and backgrounds. Semantic segmentation has many benefits over traditional object detection techniques. It enables more accurate and detailed analysis of images and videos, making it easier for autonomous vessels to make informed decisions and avoid obstacles. Additionally, it can help reduce the amount of data required for processing, as it allows the computer to focus on the relevant parts of the image rather than processing the entire scene [19][20][21]. Training a successful semantic segmentation model requires a large amount of annotated data which typically includes an image and a corresponding mask with classes.

Training a semantic segmentation model on a small dataset can be challenging due to the limited amount of data available for training. This can result in overfitting or poor generalization of the model to new, unseen data. Transfer learning is a technique that can help solve this challenge by using knowledge learned from pre-trained models on large datasets to improve the performance of the model on the smaller dataset. By using a pre-trained model as a starting point and fine-tuning it on the smaller dataset, the model can learn to identify relevant features in the new data more effectively. This can help to improve the accuracy and efficiency of the model while reducing the training time and resources required [22], [23].

Deep learning is a prominent approach in the field of Artificial Intelligence. Deep learning involves training complex computational models known as neural networks to recognize patterns and make predictions from data. These models are inspired by the human brain's interconnected neurons and their ability to process information. Deep learning leverages multiple layers of these interconnected units to automatically learn hierarchical features from raw data, making it particularly adept at handling tasks that involve understanding and extracting intricate patterns from images, text, audio, and other types of data. DeepLabV3, FCN, and LRASPP are popular deep-learning models for the semantic segmentation of images. These models have been extensively used in various applications such as self-driving cars, medical imaging, and robotics [24], [18], [25].

Deep learning models often require a large number of parameters to achieve high accuracy, which can make them slow to train and computationally expensive to run. To address this challenge, researchers have developed a variety of "backbones" that can be used as the foundation of different models. All of the models above come with certain supported backbones like MobileNetV3, Resnet50, and Resnet101. In the context of segmentation models, MobileNetV3 is often used as a lightweight backbone that can be used in mobile and embedded applications, while ResNet50 and ResNet101 are used in applications where higher accuracy is required at the expense of increased computational cost and memory usage [26].

Data augmentation and albumentation are techniques used to artificially increase the amount and diversity of data available for training ML or deep learning models. By applying a variety of transformations to the original dataset, such as rotations, flips, distortions, and other manipulations, the augmented dataset can better capture the range of variability present in real-world data. This can improve the performance and generalization of ML models, especially in cases where the original dataset is limited in size or variety [27].

Hyperparameter tuning refers to the process of finding the optimal set of hyperparameters for a ML or a deep learning model. Hyperparameters are adjustable parameters that are not learned from the data but are set before the training process begins. Examples of hyperparameters include learning rate, batch size, regularization strength, and number of layers in a neural network. The choice of hyperparameters can significantly impact the performance of a model, and selecting the optimal values can be challenging [28].

The learning rate refers to the step size that is used to update the parameters of a model during training. It determines how quickly the model learns from the data and adjusts its internal parameters. If the learning rate is too high, the model may overshoot the optimal solution and fail to converge, while if it is too low, the training may take longer to converge or get stuck in a suboptimal solution. A good learning rate is one that enables the model to converge to the optimal solution quickly and efficiently [29].

Batch size refers to the number of samples that are processed in a single forward/backward pass during training. It determines how many samples are used to calculate the gradient of the loss function and update the model's parameters. A larger batch size can result in faster training times, but it requires more memory and may result in less accurate updates. On the other hand, a smaller batch size can be more accurate, but it requires more time to train and may result in slower convergence [29].

The number of epochs refers to the number of times the entire training dataset is passed through the model during training. Each epoch consists of one forward pass followed by one backward pass (where the parameters are updated). The number of epochs can impact the performance of a model. If the number of epochs is too low, the model may not learn enough from the data, while if it is too high, the model may overfit to the training data and perform poorly on unseen data. A good number of epochs is one that balances the model's ability to learn from the data and its ability to generalize to new data [29].

Another concept useful for semantic segmentation model evaluation is Intersection over Union (IoU) which measures the overlap between the predicted object and the ground truth object. Specifically, IoU is defined as the ratio between the area of overlap and the area of union between the predicted object and the ground truth object. In simpler terms, IoU can be thought of as the amount of overlap between the predicted object and the ground truth object, relative to their total area. A higher IoU score indicates better object detection performance. IoU was calculated separately for each predicted class: background, sea, and sky. Pixel accuracy measures the percentage of pixels in the image that are classified correctly by the algorithm. Pixel accuracy is calculated as the ratio between the number of correctly classified pixels and the total number of pixels in the image [30].

Linear regression is a statistical method for modeling the relationship between a dependent variable and one or more independent variables by fitting a linear equation to the observed data. The goal of linear regression is to find the best fit line that describes the relationship between the variables. The equation takes the form of Y = a + bX, where Y is the dependent variable being predicted, X is the independent variable being used to make the prediction, a is the intercept (the predicted value of Y when X is zero), and b is the slope (the change in Y for each unit change in X) [31].

During training, the model aims to minimize loss by adjusting its parameters through techniques like backpropagation and gradient descent. By iteratively optimizing the loss function, the model learns to improve its object detection capabilities, leading to better object localization and classification performance. The overlap loss measures the agreement between predicted and ground truth bounding boxes based on their intersection over union (IoU). It encourages the predicted bounding boxes to closely match the ground truth boxes in terms of spatial overlap. The Mean Absolute Error (MAE) loss measures the absolute difference between the predicted and ground truth bounding box coordinates. It emphasizes minimizing the discrepancy between the predicted and true values, regardless of their specific relationship or spatial overlap. MAE loss is useful for ensuring precise localization of objects without relying solely on the IoU metric [32], [33].

3. Related Work

There are several state-of-the-art methods that have been developed to overcome the limitations of using bounding boxes for object detection. Some of the most popular and widely used methods are focusing on improving the detection of edges of the bounding box and give quite good results. Some of the existing approaches are Yolov7 [2], Yolov8 [3], CenterNet [11], FCOS [12], and Meta R-CNN [13]. However, these methods cannot resolve the problem of detection of bounding boxes containing more sea than the actual boat location since they focus on improving a bounding box detection, but not on detecting an exact ship location.

There are methods that focus on identifying and segmenting individual objects within an image. These methods appear to be more suitable for detecting the object itself instead of focusing just on bounding boxes. The first methods are object detection methods with masks that include Mask R-CNN [7], Panoptic FPN [34], and Hybrid Task Cascade [35]. These methods use instance segmentation to detect objects in an image by generating a mask for each object in an image. The second methods are object detection with semantic segmentation that includes DeepLabv3+ [8] and Segment Anything [4]. These methods combine object detection and semantic segmentation to improve the accuracy of object detection. The third methods are object detection method with keypoints that include Deformable DetNet [9] and ExtremeNet [10]. These methods detect objects by identifying keypoints or landmarks on the objects. Forth methods are object detection with the attention that includes DETR [6], and Foveabox [36]. These methods use attention mechanisms to focus on important regions of the image for object detection. Mentioned approaches show good performance results and have the potential to solve bounding boxes issues. On the other hand, training such models requires a large dataset with high-quality annotations, and it can be computationally expensive.

One of the most used, researched methods is semantic segmentation which has high dataset availability. Indeed, semantic segmentation can handle arbitrary shape objects as well as overlapping objects, it has high precision and provides rich information about the contents of an image [37],[38]. Additionally, some recent frameworks like Mask R-CNN [7] and Hybrid Task Cascade [35] use semantic segmentation masks in conjunction with bounding boxes to improve object detection performance. Additionally, Segment Anything [4] is a powerful and flexible tool for performing instance segmentation on a wide range of visual tasks. Nevertheless, while semantic segmentation can provide a more detailed and accurate representation of objects in an image, it can also be computationally expensive and may not be suitable for real-time applications. Some of the problems of using semantic segmentation in a real-time environment include high computational cost, memory constraints, latency, and limited hardware support [39].

There is a Poly-YOLO [5] model that uses line annotations for objects. This model tends to improve the detection of objects with complex shapes. However, the model has not been used in a maritime environment and there is a lack of available datasets to train one. Furthermore, there is a lack of knowledge in the usage of a combination of bounding boxes and line annotations for object detection. Related works either use bounding boxes or line annotations (segmentation masks). Combining bounding boxes and line annotations for object detection is an open research area.

There has been research on improving object detection and classification by reducing the number of faulty annotations and proposing corrections to existing datasets. Mixed results were found in a preliminary study where performance improved for one dataset, but got worse for another [40]. The Singapore Maritime Dataset was improved by correcting the noisy labels and imprecisely located bounding boxes which resulted in better detection and classification performance [41]. Another paper developed an automatic approach to correct imperfect seizure annotations on wearable EEG data and could improve the performance of automated seizure detection models [42]. Results show the potential for improvement of annotations, however, the problem of sea-pixel correction within the bounding box was not researched.

Related works provide a range variety of methods that can indeed provide a more accurate representation of a ship in a maritime environment. However, the main limitations are computational cost and lack of available datasets. Some of the related approaches showed an improvement in performance on a corrected dataset. Nevertheless, the automatic transformation of bounding boxes is an open research area. Therefore, the thesis approach will use the existing approach of semantic segmentation and the datasets available for it and produce a new object representation at a reduced cost, and resources and without dataset limitation.

Related work summary is presented in the table 1 below:

Purpose	Methods	Advantages	Limitations
To improve the detection of edges of the bounding box	Yolov7 [2], Yolov8 [3], CenterNet [11], FCOS [12], and Meta R-CNN [13]	Efficient object detection, high accuracy, good scalability, ability to handle complex scenes	The focus is on improving a bounding box detection, but not on detecting an exact ship location within an image
To identify and localize objects within an image, as well as provide a pixel-level segmentation of the object's boundaries	Mask R-CNN [7], Panoptic FPN [34], and Hybrid Task Cascade [35]	Multi-task capability, high accuracy, end-to-end training, scalability, improved contextual awareness	Are computationally expensive and may not be suitable for real-time applications
To perform pixel-wise classification of the objects to produce a more precise segmentation of the object of interest (e.g., a boat)	DeepLabv3+ [8], Segment Anything [4]	Can handle various types of objects and structures inside images, is robust to diverse environmental conditions and occlusions, can generalize to new images	Are computationally expensive and may not be suitable for real-time applications, training and inference can be time-consuming. Require a lot of training data
To identify keypoints or landmarks on the objects	Deformable DetNet[9] and ExtremeNet[10]	Have improved object detection accuracy, faster inference speed, and are able to handle objects with complex shapes	Are computationally expensive and may not be suitable for real-time applications, have difficulty in generalizing to new environments and situations
To focus on important regions of the image for object detection	DETR [6], and Foveabox [36]	DETR can handle complex and variable-sized objects in an image, Foveabox is fast and accurate	DETR requires large amounts of training data and computation, Foveabox has problems with objects that have complex shapes or small sizes, is computationally intensive, and may not be suitable for real-time applications
To improve the detection of objects with complex shapes	Poly-YOLO [5]	Fast, making it suitable for real-time object detection tasks, improved accuracy, easy integration	Has not been used in a maritime environment and there is a lack of available datasets to train one, can be complex, may not generalize well for new data
To correct existing datasets and improve performance	Dataset correction methods [40], [41], [42]	Corrected datasets that are available for public use, improved accuracy, and model performance, easy integration	The problem of sea-pixel correction within the bounding box was not researched.

Table 1: Related work summary.

There are several commonalities among the introduced approaches and existing methodologies. For instance, both exploit the potential of semantic segmentation to produce richer and more detailed object representations. Additionally, both sets of approaches acknowledge the challenges related to computational complexity and the need for high-quality annotated datasets. Existing methods have demonstrated improvements by addressing annotation issues and refining object detection performance.

Nonetheless, notable differences exist between the new approach and existing methods. While the introduced methodology adopts the semantic segmentation foundation and publicly available datasets, it innovatively combines bounding boxes and line annotations to create a refined object representation tailored to maritime environments. This novel representation bridges the gap between bounding boxes and detailed object shapes, addressing the sea-pixel correction problem while utilizing available resources efficiently. In contrast, existing methods often focus solely on one aspect of object detection, such as bounding box improvement or instance segmentation, without directly addressing the maritime context's unique challenges.

The goal of the thesis is to use a combination of the bounding boxes with line annotations to propose a new object representation. This new object representation will consist of existing bounding boxes that are corrected using additional line annotations. Already existing work on the usage of line annotations includes annotated dataset which does not exist for maritime conditions for public use. Therefore, in the scope of the thesis, line annotations will be generated using the semantic segmentation approach which is another common approach that has publicly available datasets. An approach to creating line annotations automatically when necessary will be developed. New object representation will be constructed using bounding boxes and line annotations. The new representation will be defined after conducting research on existing shapes and producing and comparing representations. Proof of concept on the benefits of new representation compared to the bounding boxes will be derived. As a result, the thesis proposes an automated annotation scheme to compose a new object representation from existing bounding boxes.

4. Problem Formulation

The purpose of the thesis is to improve the accuracy of ships' locations in maritime environments. Bounding boxes are the most commonly used representation for object detection, including ship detection. As a result, many existing datasets for ship detection using bounding boxes to represent the location and extent of ships in images or videos. There are several reasons why bounding boxes are typically used for object detection purposes. Bounding boxes are simple and easy to annotate, they are computationally efficient and can represent objects of different shapes and sizes that can be used for both single-object and multi-object detection tasks. However, bounding boxes are a quite limited representation of objects that have a complex shape or that are positioned at an angle. Bounding boxes may not be able to accurately capture the full extent of the ship or its orientation. This can result in imprecise localization of the ship in the image or video.

While focusing on improving ship localization on the image or video it is important to consider computational efficiency. New representation should give more information on the ship's location as well as keep the computational cost low. In applications where ship detection and localization need to be performed in real-time, such as in maritime surveillance systems or autonomous navigation, computational efficiency is crucial. Slow or computationally intensive algorithms may not be able to meet the real-time constraints of these applications. Moreover, computationally efficient algorithms require less expensive hardware or cloud computing resources, resulting in cost savings.

Another key aspect of the thesis is to provide a detection algorithm to automatically find and correct the representation of ships that have a faulty localization. Reannotation of existing computer vision datasets to handle these specific situations is a costly and unwanted expense. An automated detection algorithm that can identify and correct faulty localizations can help to improve the accuracy of ship detection without the need for expensive reannotation. This approach can be particularly useful in applications where real-time or near-real-time processing is required, and where manual intervention to correct errors is not practical. In addition, the development of such an algorithm can contribute to the broader field of computer vision and object detection, as it may be applicable to other types of objects beyond ships. By providing a more robust and automated approach to handling faulty localizations, the thesis may help to improve the accuracy and efficiency of object detection algorithms in general.

Lastly, to be able to evaluate the new ship's representation, a comparison between old and new localizations has to be made. For that, a new object detection model that finds and classifies vessels on the image has to be trained and tuned. As a result, several arguments can be taken into consideration. Some of them include: how fast is the inference of the models, what is the model's accuracy, what is the percentage of images that got new representation, how many ships' locations were improved, and how many faulty representations were generated. Furthermore, a metric to compare a bounding box and a new representation can be developed. This metric can capture a difference between those localizations and evaluate how relevant new annotations are compared to existing solutions.

It is possible to formulate a research question on a given problem formulation and extend them using sub-questions:

1. How to use a semantic segmentation model to find bounding boxes that need correction?
   (a) How to train a semantic segmentation model for a maritime dataset?
   (b) How to develop a method that finds border points between water and ship inside a bounding box?
2. How to develop a method to generate a new bounding box representation that will draw a waterline between the ship and sea?
   (a) How to produce an algorithm that creates a waterline based on found border points?
   (b) How can relevant ship representations be produced through a set of filters?
3. How to choose the best annotation scheme for found bounding boxes?
   (a) What ship representations are currently available in the literature?
   (b) What are the ways to evaluate a ship representation?

4. How to evaluate a model that predicts new bounding box representation?
   (a) How to train an object detection model using new ship representations?
   (b) How does the object detection model trained on new ship representations compare to the model trained on old bounding boxes?

It is important to keep in focus the limitations that this thesis holds:

- New representation may result in unnecessary reannotation.

  Since the generation of new object representations is automatic, it is possible that some of the suggestions will be meaningless and won't improve the localization and representation of an object.

- Evaluation of new annotations requires manual work.

  While the proposed method can speed up the process of re-annotation and correction of faulty bounding boxes, the results still need to be validated to be trustful.

- There is a lack of available datasets for ship locations other than bounding boxes or segmentation masks in a maritime environment.

  For a successful object detection model, an available dataset is crucial since manual annotations of a new representation are a long and costly process and are quite undesired.

- There is a lack of previous studies of different object representations for ship's locations.

  Object detection in the maritime environment is challenging due to various factors such as the complexity of the maritime scene, the presence of water reflections, and the variability of object shapes and sizes. This field is constantly developing but lacks research on various representations of the ship's features and localizations.

From the research sub-questions and a problem formulation a set of goals are derived:

1. Train semantic segmentation model for a maritime dataset.
2. Develop a method that finds border points between water and ship inside a bounding box.
3. Produce waterlines based on border points.
4. Derive a set of filters to produce relevant representations.
5. Create several ship representations and criteria to evaluate them.
6. Train an object detection model using new ship representation
7. Compare model with new representations and old bounding boxes

An accurate representation of objects in computer vision is crucial for various applications, especially in domains like maritime safety where decisions made based on the output of the computer vision system can have serious consequences. The new representations should give more information on the ship's location while keeping the computational cost low. A detection algorithm that can identify and correct faulty localizations can help to improve the accuracy of ship detection without the need for expensive reannotation. Furthermore, new representation should be able to provide more information about the ship's location while maintaining computational efficiency. This is especially important in applications where ship detection and localization need to be performed in real-time, such as in maritime surveillance systems or autonomous navigation. Slow or computationally intensive algorithms may not be able to meet the real-time constraints of these applications. The results of this project can contribute to the development of more effective maritime awareness systems as well as give general guidelines on how to approach the issue in other computer vision related domains. Moreover, the development of this thesis may help to improve the accuracy and efficiency of object detection algorithms in general.

5. Research methodology and methods

Here is a detailed description of a set of goals defined in the section above.

Goal 1. Train semantic segmentation model for a maritime dataset.

This goal was developed using the experimental method (inductive approach).

For a defined research question "How to use a semantic segmentation model to find bounding boxes that need correction?" it is possible to create a sub-question: "How to train a semantic segmentation model for a maritime dataset?". Experiments were designed by selecting an appropriate pre-trained convolutional neural network architecture from the pytorch library, training the model on the maritime dataset, and evaluating its accuracy using metrics such as IoU (Intersection over Union) and pixel accuracy. Multiple datasets were utilized, including a publicly available dataset and a private dataset from a company. The performance of the model on these datasets was analyzed, and conclusions were drawn based on the observed data. The study aimed to assess the effectiveness of the model and identify potential areas for improvement.

Goal 2. Develop a method that finds border points between water and ship inside a bounding box.

This goal was developed using the experimental method (inductive approach).

For a defined research question "How to use a semantic segmentation model to find bounding boxes that need correction?" it is possible to create a sub-question: "How to develop a method that finds border points between water and ship inside a bounding box?". The experiments are using edge detection algorithms and contour analysis to identify border points between water and ship inside a bounding box. Designed experiments will test different edge detection algorithms and contour analysis methods on sample images with annotated bounding boxes, and will evaluate their effectiveness in identifying border points. Data collection will include acquiring sample images with annotated bounding boxes. Then, the performance of different edge detection algorithms and contour analysis methods in identifying border points will be analyzed, and the most effective method will be selected. Finally, conclusions about the effectiveness of the selected method and areas for improvement will be derived.

Goal 3. Produce an algorithm that creates a waterline based on found border points.

This goal was developed using the experimental method (inductive approach).

For a defined research question "How to develop a method to generate new bounding boxes representation that will draw a waterline between the ship and sea?" it is possible to create a sub-question: "How to produce an algorithm that creates a waterline based on found border points?". Linear regression is used to produce a waterline due to the linearity of the data and the simplicity of the approach. The designed experiments include testing the Linear Regression algorithm on sample images with annotated bounding boxes and evaluating the accuracy of the produced waterlines. For data collection same sample images as in Goal 2 are used. Moreover, the accuracy of the produced waterline areas will be analyzed using ship evaluation criteria developed in goal 4. The effectiveness of the algorithm and areas for improvement will be derived.

Goal 4. Derive a set of filters to produce relevant representations.

This goal was developed using the experimental method (inductive approach).

For a defined research question "How to develop a method to generate new bounding boxes representation that will draw a waterline between the ship and sea?" it is possible to create a sub-question: "How can relevant ship representations be produced through a set of filters?" A set of filters based on important features and attributes of the ship will be created to produce relevant ship representations. The set of experiments includes developing filters based on important features and attributes of the ship such as size, shape, location of the waterline, and other relevant factors. Then, these filters will be applied to the ship representation produced in Goal 4 and their effectiveness in producing relevant representations will be evaluated. For the data ship representations produced in Goal 4 will be used and the set of filters created in Step 3 will be applied to produce new ship representations. Next, the resulting ship representations will be analyzed as well as areas for improvement. A comparison of the performance of different filters will be made and based on it conclusions about the effectiveness of the filters in producing relevant ship representations will be drawn.

Goal 5. Create several ship representations.

This goal was developed using the survey method.

For a defined research question "How to choose the best annotation scheme for found bounding boxes?" it is possible to create a sub-question: "What ship representations are currently available in the literature?" Then, relevant sources and search keywords will be identified. After conducting a search, the results will be screened to create a list of possible ship representations. As a result, each found representation will be evaluated within the project requirements in the next goal.

Goal 6. Define criteria to evaluate ship representations and select the best one based on them.

This goal was developed using the experimental method (inductive approach).

For a defined research question "How to choose the best annotation scheme for found bounding boxes?" it is possible to create a sub-question: "What are the ways to evaluate a ship representation?". Experiments will be created to evaluate ship representations using a scoring system that is based on important features and attributes of the ship and statistical analysis. Designed experiments will test the usage of different scores based on sets of various important features and attributes. They will also include statistical analysis of the produced representations to compare their relevance. Then, the performance of each scoring system will be evaluated and the best one will be selected based on the results of the statistical analysis. Finally, conclusions about the effectiveness of the selected method and areas for improvement will be produced.

Goal 7. Train an object detection model using new ship representation.

This goal was developed using the experimental method (inductive approach).

For a defined research question "How to evaluate a model that predicts new bounding box representation?" it is possible to create a sub-question: "How to train an object detection model using new ship representations?" A transfer learning approach will be used and the new loss function will be composed to train an object detection model with improved accuracy. Then, experiments can be designed to use transfer learning with a pre-trained object detection model and test the performance of the tuned loss function for a new object representation. The same maritime dataset as in Goal 1 and the ship representations produced in Goal 4 and Goal 6 is used. The next step includes the evaluation of the performance of the trained model on a test set using metrics such as precision, recall, and F1 score. Lastly, conclusions about the effectiveness of the model can be made as well as future improvements.

Goal 8. Compare the model with new representations and old bounding boxes.

This goal was developed using the experimental method (inductive approach).

For a defined research question "How to evaluate a model that predicts new bounding box representation?" it is possible to create a sub-question: "How does the object detection model trained on new ship representations compare to the model trained on old bounding boxes?" The experiments will include an evaluation of the performance of the model trained on new ship representations and the model trained on old bounding boxes using a test set, and a comparison of their performance using metrics such as precision, recall, and F1 score. Data used for the experiments will consist of the same maritime dataset as in Goal 1 and the ship representations produced in Goal 5 and Goal 6. As a result, the effectiveness of the new ship representations and the performance of the object detection model trained on them will be concluded.

A summary of research questions and their corresponding goals and methods is presented in table 2:

Research question	Goals	Research methodology
How to use a semantic segmentation model to find bounding boxes that need correction	Train semantic segmentation model for a maritime dataset	A pre-trained convolutional neural network architecture from a python library pytorch
	Develop a method that finds border points between water and ship inside a bounding box	Edge detection algorithms and contour analysis
How to develop a method to generate new bounding box representation that will draw a waterline between the ship and sea	Produce an algorithm that creates a waterline based on found border points	Linear Regression algorithm
	Derive a set of filters to produce relevant representations	Evaluation of filters based on relevant features of the ship
How to describe an annotation scheme for found bounding boxes	Create several ship representations	Research of existing representations
	Define criteria to evaluate ship representations and select the best one based on them	Development of scoring methods and usage of statistical analysis
How to evaluate a model that predicts new bounding box representation	Train an object detection model using new ship representation	Transfer learning approach and composing of new loss function
	Compare model with new representations and old bounding boxes	Comparison of their performance using metrics such as precision, recall, and F1 score

Table 2: Research questions and corresponding goals and methods.

Step-by-step process of the research work (see figure 1) includes:

1. Semantic segmentation model training. This step includes training, hyper-parameter tuning, and evaluation of different existing semantic segmentation models. The best segmentation model is selected to be used in the next steps.

2. Finding water points that separate ship and sea. This step covers running a segmentation model on images to generate a background-sea-sky mask. Then, for each bounding box on the image, a set of water points that are separating background pixels from sea pixels inside of the bounding box is found.

3. Produce waterline based on found water points. Here water points are transformed into a waterline that is drawn between the sea and the ship for each bounding box.

4. Filter produced waterlines. In this step, only waterlines that are relevant are kept. Filters help to decrease the number of faulty representations that can be caused by misclassification during segmentation mask predictions.

5. Create several ship representations. Several new ship representations are picked using state-of-art research on existing geometrical shapes to represent an object. Shapes are developed using existing bounding boxes and produced waterlines.

6. Evaluate ship representations with a derived set of criteria. A set of criteria is defined based on relevant features of the ship. Then, based on them the best ship representation is picked for the future trainings in the next step.

7. Train an object detection model with new ship representation. An object detection model is trained and tuned using a new ship representation.

8. Derive a proof of concept for a new ship representation. Lastly, a decision is made on how well the model performs based on the evaluation of the performance. A comparison is made with a baseline model trained on bounding boxes.

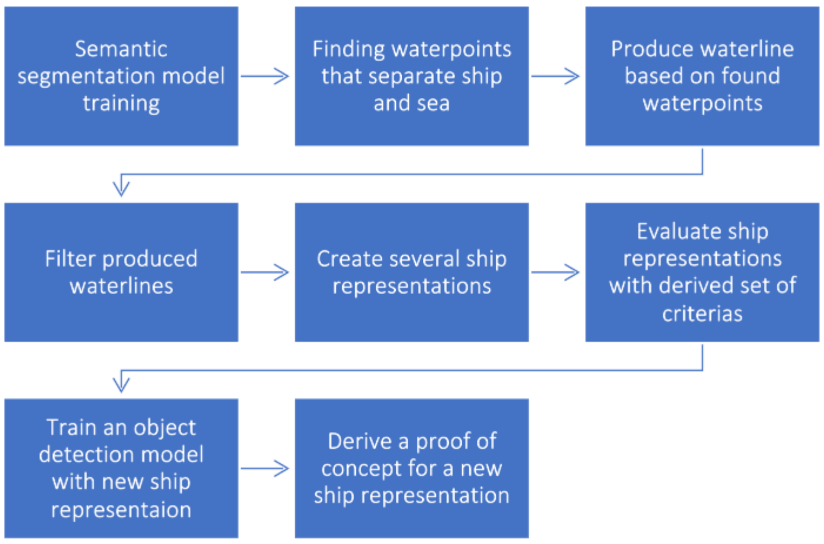

Semantic segmentation model training
Finding waterpoints that separate ship and sea
Produce waterline based on found waterpoints
Filter produced waterlines
Create several ship representations
Evaluate ship representations with derived set of criterias
Train an object detection model with new ship representaion
Derive a proof of concept for a new ship representation

Figure 1: Step-by-step process of the research work.

6. Ethical and Societal Considerations

In terms of research ethics, this study recognizes the societal aspects of the work. It aims to contribute to economic, social, and ecologically sustainable development by developing a method to generate new bounding box representations for drawing a waterline between the ship and the sea. By accurately identifying the border points and creating an effective waterline, it can potentially enhance maritime safety, facilitate efficient navigation, and support environmental preservation efforts.

Additionally, legal and political aspects will be considered in this work. Compliance with applicable laws, regulations, and guidelines related to data protection and privacy will be ensured. The research will be conducted in alignment with ethical standards and will respect the rights and interests of all stakeholders involved.

It is important to highlight that, based on the nature of the research and the precautions taken, this study believes that there are no significant research ethics issues associated with the work. The focus is on conducting the research ethically, respecting privacy, and adhering to relevant legal and societal considerations.

In terms of research ethics, it is important to note that certain data can be shared, such as a selection of relevant images, but the original object detection model is Groke Technologies' intellectual property and therefore any details about it cannot be shared. Any modifications to the object detection model's code should not be shared, but its functionality can be described.

7. Detecting faulty bounding boxes using semantic segmentation

First step in generating new bounding box representation is ML approach semantic segmentation. The main challenge of the segmentation model training is image resolution. MaSTr1325 has 512x384 resolution whereas Groke300 has images with 3840x216 and 3840x640 resolutions. There are two main approaches in cases when training data consists of several datasets with different image resolutions. The first approach suggests inputting data with different scales without prior re-scaling. This approach works only for fully convolutional models, which can handle input data of varying sizes and requires training batches to consist only of images and masks of the same scale. The second approach includes using patches of images and masks. This approach involves dividing the larger images into smaller patches, which can then be used to train the segmentation model. The first approach is chosen as the main approach and training parameters are tuned based on it. The second approach is suggested as an alternative approach as it provides some pre and post-processing of the data. The main benefit of the alternative approach is that it does not require high computational power.

7.1. Semantic segmentation datasets

There is a lack of publicly available datasets for segmentation in the maritime environment. The suitable dataset will include a classification of sea and ship on the image since those two classes need to be separated visually using a new annotation scheme. One of the most famous datasets is COCO [43]. COCO (Common Objects in Context) is a large-scale image recognition, segmentation, and captioning dataset that is widely used in computer vision research. The COCO dataset is particularly useful for training and evaluating deep learning models for object detection, segmentation, and image captioning tasks. However, it classifies a separate object on the image and not background pixels. It makes a COCO dataset not suitable for sea/ship pixel classification.

Chosen dataset is Marine Semantic Segmentation Training Dataset (MaSTr1325) [44]. This dataset includes 1325 images that were manually annotated. Segmentation masks include 3 classes of pixels: background, sea, and sky. There is a label of ignore region / unknown category that covers uncertainty regions between semantically different classes. The dataset's annotations were checked with an expert to ensure high quality. The dataset includes images of various ships that can be found in sea conditions. Nevertheless, the main limitation of the dataset is the image size of 512x384 which is quite a low resolution when scaling up to real-time applications. Moreover, manual annotation can introduce bias to the resulting masks. Also, the dataset might lack a diversity of environmental conditions due to its size. An example of the dataset can be seen in figure 2 where the mask is put on the corresponding image and the blue color represents sea class, yellow - sky class, and violet - background class.

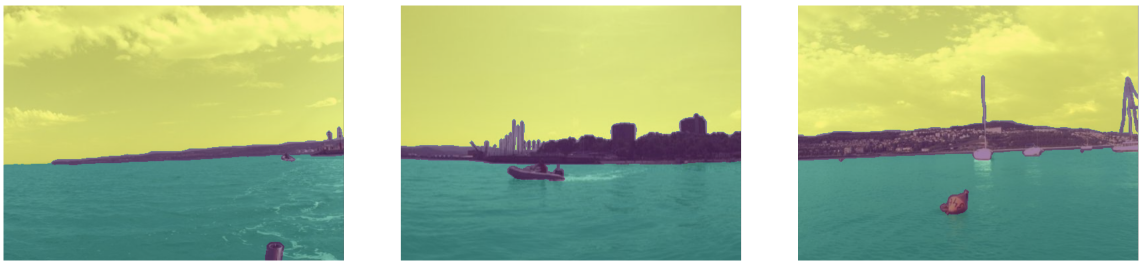

(a) Image 1. (b) Image 2. (c) Image 3.

Figure 2: MaSTr1325 segmentation dataset example.

In order to make the results scalable to real-time applications that use high image resolutions, Groke has provided a small subset of annotated images that includes 200 images of high-resolution annotations of 3840x2160 and 100 images of the high resolution of 3840x640. Additionally, the small subset of high-resolution images adds to the diversity of the training data, making the models more robust and generalizable. An example of the dataset can be seen in figure 3 where the mask is put on the corresponding image and the blue color represents the sea class, yellow - sky class, and violet - background class.

Datasets summary is presented in the table 3:

Dataset name	**Dataset size**	**Image and mask resolution**	**Number of classes**
MaSTr1325	1325	384x512	4
Groke300	300	3840x2160, 3840x640	3

Table 3: Datasets summary.

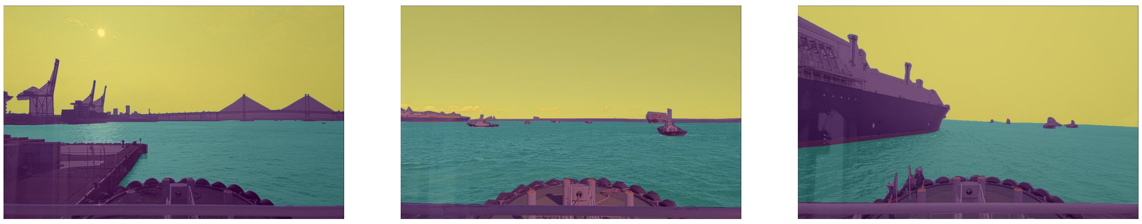

(a) Image 1. (b) Image 2. (c) Image 3.

Figure 3: Groke segmentation dataset example.

7.2. Semantic segmentation training

Transfer learning approach is used to prevent overfitting and to work with a limited amount of data which can be implemented by using pre-trained models. PyTorch provides a variety of pre-trained semantic segmentation models, such as DeepLabV3 [24], FCN [18], and LRASPP [25], which can be fine-tuned on the MaSTr1325 and Groke dataset to improve their performance on the specific task of marine obstacle detection.

In the thesis context, 4 models were selected for training and comparison due to their high benchmark performance: DeepLabv3 with Resnet50 and Resnet101 backbones and FCN with Resnet50 and Resnet101 backbones [45].

Experimental Parameters:

- Learning Rate: A learning rate of 0.001 was selected. Too small learning rate results in a very slow training process and more epochs are needed to reach a certain optimal validation loss. Furthermore, if the learning rate is too high the model does not reach a certain optimal validation loss.
- Batch Size: A batch size of 16 was selected, as recommended in a MaSTr1325 dataset paper [44]. Very small batch size was found to make performance worse. Too big batch size results in very high computational time and memory as well as results in decreased generalization.
- Epochs: The number of epochs was set to 100, and the best three models were saved based on the lowest validation loss. Typically, around 50-90 epochs were found to be sufficient, and sometimes as few as 30 epochs. A too big number of epochs resulted in overfitting and too small in underfitting. Moreover, the efficient number of epochs also depends on the model size since the bigger model will require a smaller number of epochs. This occurs because bigger models have more training parameters and therefore learn faster from the input data. Overall, 100 epochs are chosen for training and 3 best models are saved based on the lowest validation loss. In that way, it is possible to ensure the model is fitted well with the data.

First, the learning rate is tuned. Generally, a learning rate between 0.001 to 0.1 is considered a reasonable starting point for many applications [46]. Therefore, 0.1, 0.01, and 0.001 are tested and compared. The learning rate was tested with set batch parameter 16 which is recommended in the maritime dataset paper [44] and with a number of epochs equal to 100. This is due to the fact that typically, for a small dataset, the number of epochs can range from 50 to 100. After, tuning the learning rate, a number of epochs is tuned. They are selected to be 50, 100, and 200. After analyzing the validation loss graph, the number of epochs is set to 100 with saving the models with the lowest validation loss. From figure 4 it is visible that training and validation losses are going up and down through the training and therefore it is not possible to estimate an optimal number of epochs beforehand. Commonly, this number is located between 50 and 100. A number of epoch tuning is done with a learning rate set to 0.001 and batch size to 16. Lastly, the batch size parameter is tuned. In general, a smaller batch size allows for more frequent updates of the model weights and can help avoid overfitting, but it may result in slower convergence and longer training times. For a small dataset, a typical batch size may range from 4 to 16 and is set to a number that is a power of 2. Chosen batch size parameters are 4, 8, 16, and 32. Batch size tuning is done with a learning rate set to 0.001 and a number of epochs set to 100.

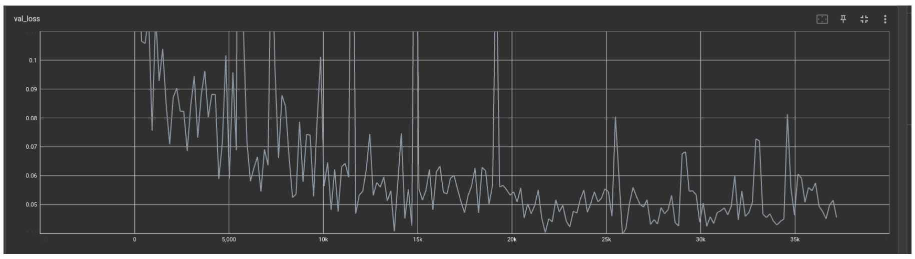


Figure 4: Validation loss graph during training example.

A table was created to document the different parameter tuning and optimization steps taken during training. This table includes details on the different learning rates, batch sizes, and numbers of epochs tested, as well as the resulting training and validation loss for each combination of parameters (table 4). For the learning rate table displays the best training and validation loss values and taken time of training to reach those losses. For the number of epochs, the table represents final training and validation losses as well as total training time. For the batch size, the best training and validation losses are shown and the corresponding training time.

The creation of the train, validation, and test sets was done randomly to ensure a representative sample of the data. The random split was performed with a seed value to ensure the reproducibility of the results. The data were preprocessed before the split to ensure that each set had a balanced representation of the classes. Furthermore, the data augmentation techniques were applied to increase the size of the training set and reduce overfitting. A standard 70-20-10 split is used for the train, validation, and test sets, respectively.

The data was normalized using the mean and standard deviation parameters provided by the PyTorch documentation: (0.485, 0.456, 0.406), (0.229, 0.224, 0.225) [47]. The masks were stored with 0-1-2 pixel values, where 0 represented the background, 1 represented water, and 2 represented sky. Input data is input with different scales without prior re-scaling to the same scale. In this way, training data consisted of three categories of batches: MaSTr with an image scale of 384x512, Groke with a 1280x720 scale, and Groke with 1152x640. Groke300 data is rescaled prior to training into a smaller scale to be an appropriate size for training. Too big image resolution results in slow convergence and high computation need for resources. Data was rescaled into 3 times smaller scale and for 3840x2160 image resolution, the resulting resolution is 1280x720, whereas for 3840x640 image resolution data was also shrunk in width and resulted in 1152x640 scale.

Moreover, data augmentations are used to improve the diversity and generalization of the training. A commonly used library for augmentations in Python is Albumentations which is designed to be fast, flexible, and easy to use [?]. The MaSTr1325 dataset underwent two types of augmentations to expand its size and improve the accuracy of the ML model. These augmentations were used to create two additional versions of the original dataset, resulting in a total of three datasets for training and testing. An example of image augmentation is presented on image 12 where 5a represents the original example image before augmentations, 5b, 5c example images where augmentations were applied, 5d represents original example mask before augmentations, and 5e, 5f example masks where augmentations were applied. Similarly, the Groke dataset was extended using 5 sets of augmentations and resulted in 6 times bigger dataset. A summary of used augmentation is presented in the table 5:

Parameter name	**Chosen parameter value**	**Experimental parameter value**	**Training loss**	**Validation loss**	**Training time**
Learning rate	0.001	0.1	0.02742	0.06903	6.156 h
		0.01	0.02051	0.05721	6.154 h
		0.001	0.02008	0.04109	4.671 h
Number of epochs	50-100	50	0.03508	0.04497	3.053 h
		100	0.01996	0.0622	6.166 h
		200	0.008374	0.04554	12.39 h
Batch Size	16	4	0.03944	0.06194	3.432 h
		8	0.0509	0.05402	2.323 h
		16	0.04399	0.04509	2.742 h
		32	0.03616	0.04459	2.606 h

Table 4: Training parameters tuning.

Dataset name	**Augmentation name**	**Augmentation description**
MaSTr1325	Transformation	HorizontalFlip, VerticalFlip, ElasticTransform, GridDistortion, OpticalDistortion, RandomBrightnessContrast, ColorJitter, SafeRotate
	Weather augmentation	ColorJitter, HorizontalFlip, SafeRotate, RandomSunFlare, RandomShadow, RandomFog
Groke300	Transformation 1	HorizontalFlip, VerticalFlip, ElasticTransform, GridDistortion, OpticalDistortion, RandomBrightnessContrast, ColorJitter. SafeRotate
	Transformation 2	HorizontalFlip, VerticalFlip, RandomBrightnessContrast, ColorJitter, SafeRotate
	Transformation 3	HorizontalFlip, VerticalFlip, SafeRotate
	Weather augmentation 1	ColorJitter, HorizontalFlip, SafeRotate, RandomSunFlare, RandomShadow, RandomFog
	Weather augmentation 2	HorizontalFlip, VerticalFlip, SafeRotate, RandomSunFlare, RandomShadow, RandomFog

Table 5: Augmentations summary.

7.3. Segmentation model patches

This approach is suggesting to use patches of images and masks during training. This involves dividing each image into smaller patches of a fixed size and then using these patches to train the segmentation model. By doing so, the model is able to learn from the high-resolution details of the original images, without losing accuracy due to re-scaling.

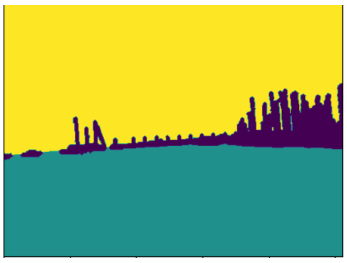
Patches are created using patch size and stride size. The patch size should be large enough to capture the relevant features of the object of interest, but not so large that it includes irrelevant background information. Chosen patch size for images with a 3840x2160 scale is 1950x1500 which is almost 2 times smaller than the original image. The patch size for images with a 3840x640 scale is 1950x650. The stride size determines the overlap between adjacent patches. A smaller stride size results in more overlap between patches and more computations, while a larger stride size reduces the number of patches and computational costs but may result in imbalanced coverage of the image. The stride is chosen as 80 percent of the patch size.

(a) Original image.

(b) Augmented image 1.

(c) Augmented image 2.

(d) Original mask.


(e) Augmented mask 1.

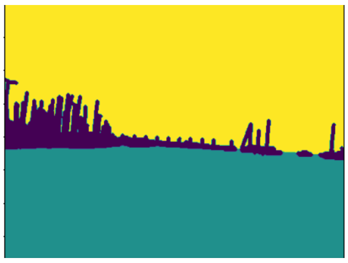

(f) Augmented mask 2.

Figure 5: Augmentations example.

Images and corresponding masks are divided into a fixed number of patches for the training and then are all re-scaled into 512x384 resolution to match the MaSTr1325 scale. In that way, image resolution is kept low and the number of patches per image is not high which results in fast training and inference.

To perform patch segmentation, the following steps can be followed:

1. Define the patch size, stride size, and output shape for the input image and its corresponding mask.
2. Implement the function, which takes the input image and mask and generates a list of image and mask patches.
3. Resize patches into 512x384 scale recognized by segmentation model.
4. Input each patch into a segmentation model and generate the corresponding mask.
5. Resize patches into original scale.
6. Implement the function, which takes the list of generated masks and combines them to form the final output mask for the original image.
7. Repeat a process for each image

To make the process faster, it is possible to run inference of patches simultaneously using GPU cores and PyTorch tensors. An example of the process is displayed in figure 6.

For training a patched model same parameters optimised for original segmentation model are used. Similarly, train, validation and test split are 70-20-10. Original set of data normalization and data augmentation is kept for patched model training. Patched segmentation model tries to reproduce original segmentation model results with a lightweight training process. Usage of patches allows to perform the trainings and inference on less powerful processor and does not require usage of high CPU and GPU clusters.

7.4. Water borderpoints detection

In order to find bounding boxes that need correction it is important to focus on water pixels inside the particular bounding box. The key challenge lies in determining whether water points are beneficial to bounding box correction.

The algorithm to detect relevant water border points is as follows. The border points are detected between water and the background in an image within each bounding box using the bounding box and classified mask from the segmentation method. The bounding box contains the coordinates of the region of interest in the image. The water border point is considered irrelevant if water points are detected higher than half of the bounding box' width. This is due to the fact that a meaningful borderline cannot start at the higher part of the bounding box. If the water is higher than mentioned distance, it is not considered in the border detection. Next, the function checks if the water is on two sides of the bounding box. In that case, the borderline can be drawn from both sides of the bounding box. The function then determines the iteration flow, which is the direction in which the function should search for the border pixels. It calculates this by finding the distance of the breakpoints from the bounding box's sides and determining which breakpoint is farther. If both breakpoints exist, it keeps the breakpoint that is closer to the bounding box's side with the longer waterline. In this way, only one borderline will be kept for each bounding box and the algorithm will continue with a longer one.

At this stage, the relevance of the water border points is only defined by their location within a particular bounding box. Water points are getting filtered when located too high within the image. Furthermore, while looking for border points of water the algorithm checks only for the water pixels that are located at the border of the image, starting with the lowest row of pixels. Another important filter includes checking for a background pixel located after each water pixel to ensure that the water point is indeed a water point.

To perform waterpoints detection, the following steps can be followed:

1. Iterate through images and bounding boxes.
2. Run a segmentation model on an image (which results in 3 class classifications: background, sea, sky).
3. Iterate pixel by pixel starting from the left lower part of the bounding box.
4. Find class sea pixels that come after the background pixels in the bounding box.
5. Ignore sea pixels that are too high in a bounding box (the accepted height of sea pixels is 1/2 of the bounding box height).
6. In case sea pixels are found from both sides of the bounding box, calculate the length of resulting water pixels and keep the longer one.
7. Repeat the process for the bounding box on each image.

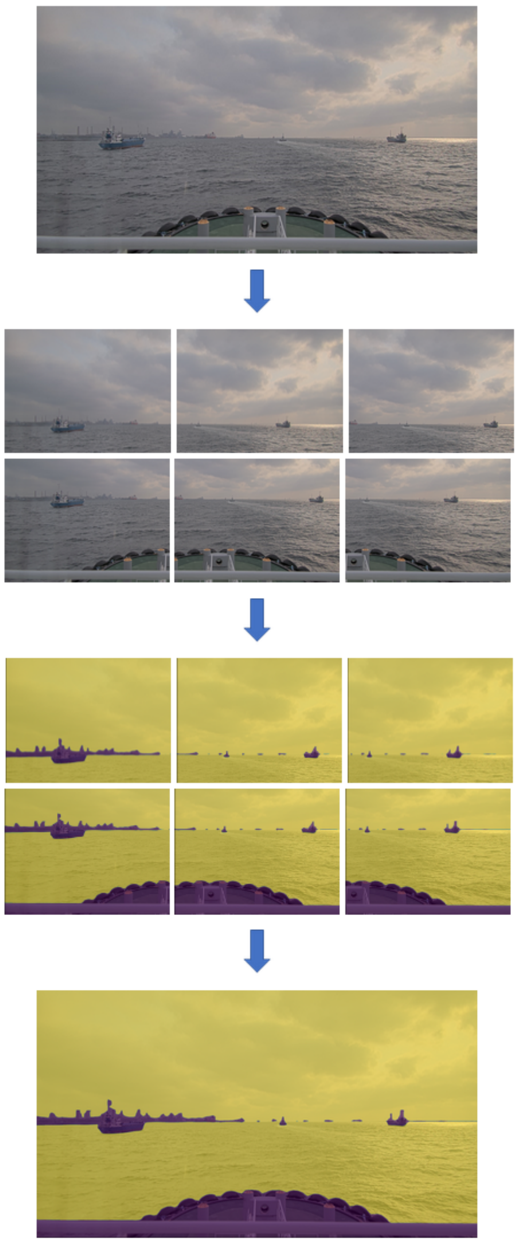


Figure 6: Patches detection example.

8. Generating waterline inside a bounding box

8.1. Linear Regression

To build a line from detected water points between ship and sea Linear Regression was used. Linear Regression is fed with detected water points. Linear Regression is used as a part of sklearn Python package. Default values available in that package are used for training. The x-coordinates of water points are predicted values and the y-coordinates are input values. First, the model is getting fit with (x, y) pairs. After the model has been trained, the x-coordinates of detected water points are input into the fitted model to get predictions of corresponding y-coordinates. In that way, the resulting (x, y) pairs compose a line.

To simplify the resulting line it should be represented by 2 points and form an interval: (x, y) coordinates of the start of the interval and end of the interval. Those 2 points should be located on the bounding box. To find interval points intersection between the bounding box and the resulting line is found. In computational geometry, finding the intersection point of two lines is a common task. One way to do this is by using linear algebra and vector calculus. An implementation of this method is done in Python using the NumPy package. The function takes in the start and end points of two lines, represented as vectors, and computes the intersection point of the lines. It first converts the inputs to NumPy arrays and computes the direction vectors of the lines. It then uses the vector cross product and the dot product to compute the intersection point of the lines. The function checks for the case where the lines are parallel or overlapping and returns the existing bounding box interval in that case since produced waterline did not result in any improvement.

To perform waterlines detection, the following steps can be followed:

1. Iterate through images and bounding boxes.
2. Run a segmentation model on an image (which results in 3 class classifications: background, sea, sky).
3. Generate water points.
4. Train Linear Regression model with x-coordinates of water points as predicted values and y-coordinates as input values.
5. Predict x-coordinates using the trained model by inputting y-coordinates of water points (Waterline is represented by predicted x-coordinates and input y-coordinates).
6. Find the intersection between the waterline and bounding box (found points represent a waterline in an interval format).
7. In case the intersection could not be found do not produce any waterline (no improvement was made).
8. Repeat the process for the bounding box on each image.

8.2. Filtering detected waterlines

Filters are applied to resulting waterlines in order to minimize the number of meaningless cases for a new representation. Those cases will most likely not need a different representation than a bounding box since will result in quite a similar or worse representation than a bounding box.

A size filter is applied to filter out ships that are too small for the segmentation model to produce high-quality masks. As a result, the waterline algorithm will most likely not find any water points or will result in a faulty representation. Area filter decreases the number of faulty representations since if the area of the bounding box that is getting cut off is too big, it means that the waterline algorithm or segmentation model failed on that image. Similarly, if the water points are detected too high on the image, it is likely that those water points are located between the ship and the border of the bounding box, but are higher than the bounding box.

To find a threshold that is good for filtering meaningless waterlines, but also does not remove successful reannotations combination of statistical analysis and experimental method are used.

First, data distribution is calculated and visualized to see what are the common values and outliers for each of the described issues above. The statistical analysis involved calculating various descriptive statistics such as mean, median, standard deviation, and quartiles. These measures provided a quantitative understanding of the central tendency and dispersion of the data. Additionally, visualizations such as histograms, box plots, or density plots were created to visualize the distribution shape and identify any skewed or outlier data points. The experimental method complemented the statistical analysis by validating the effectiveness of different threshold values. This involved systematically applying different thresholds to the dataset and evaluating the resulting annotations. Evaluation is done using a visual inspection of the generated waterlines to determine their quality. The output is examined and compared with the original bounding box and the surrounding context. The waterline should accurately separates the ship and the sea, and assess whether any water regions inside the bounding box are erroneously included. Groke Technologies are actively used as a domain expert in those evaluations.

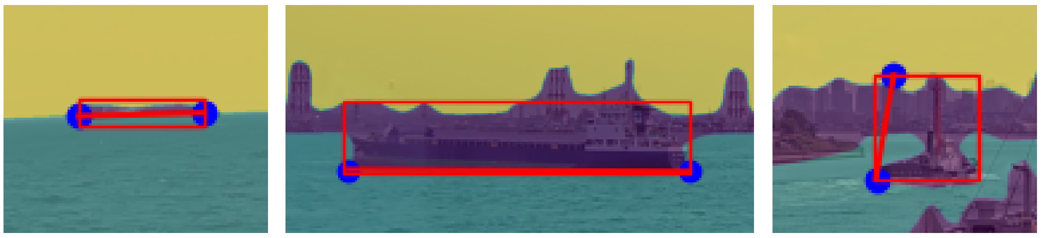
The resulting filters are described below.

Waterlines are not generated if:

1. Size of the bounding box is less than 4 percent of the image size.
2. No water was detected inside the bounding box using the water points detection algorithm.

Produced waterlines are filtered if:

1. Resulting waterline is identical to the lower side of the bounding box.
2. Area that is getting cut by a waterline is more than 25 percent of the bounding box area (fig. 7a).
3. If the area cannot be computed (waterline is almost identical with the lower side of the bounding box, fig. 7b).
4. Interval length that is getting cut from a lower side of the bounding box is less than 40 percent of the total length of the lower side (fig. 7c).



(a) Image 1. (b) Image 2. (c) Image 3.

Figure 7: Waterlines filtering example.

9. Annotation scheme to outperform bounding boxes

In the field of computer vision, the accurate detection and representation of ships in images is a crucial task for a wide range of applications, such as maritime surveillance, navigation, and remote sensing. Traditionally, bounding boxes have been widely used to represent ships in images due to their simplicity and ease of implementation. However, bounding boxes have limitations in accurately capturing the shape and orientation of ships, particularly for irregularly shaped or partially occluded objects. To address this issue, various alternative ship representations have been proposed in the literature, including trapezoids, cuboids, and polygons In this context, this work aims to provide a comprehensive overview of the existing solutions for representing ships in images, discussing their advantages and limitations.

The work aims to provide an automated approach to generate selected representations. For that waterlines that are produced by predicting segmentation masks and then running water points and waterlines algorithms can be used to produce a new annotation scheme.

9.1. Polygons

A polygon can be used to represent the shape of the ship with more accuracy than a bounding box. This approach is useful for ships with complex shapes, such as those with multiple edges or those with sharp features. The polygon can be defined by a series of points that represent the shape of the ship.

Polygon representations have been widely used in object detection and tracking tasks. In recent years, deep learning methods have been applied to improve the accuracy of object detection and classification using polygon representations. For example, a convex-plane-polygon representation was used for robot occlusion calculation in real-time[48]. Another example is a method for polygonal mapping of buildings from satellite imagery using deep convolutional neural networks [49]. In addition to this, polygons are quite useful when performing instance segmentation of objects in real-time [50] or combining it with multi-tracking of the objects [51].

However, one limitation of polygon representations is that they can be computationally expensive, especially when using a large number of points to accurately represent complex ship shapes. In addition, the accuracy of polygon representations can depend on the quality and resolution of the input images. Despite these limitations, polygon representations remain a promising approach for ship detection and classification tasks, especially for ships with complex shapes. Moreover, it is possible to variate a number of points to balance computational costs and meaningful representation.

Generation of polygons is done automatically for each bounding box where a waterline is detected. Polygon is formed from the interval points proposed in the waterline approach. The waterline cuts either one or two lower points of the bounding box. In case where a waterline cuts one lower point (see images 15a, 15b), that one point will be substituted with two interval points. In the case where the waterline cuts two lower points of the bounding box (see image 15c), both lower points are substituted by interval points). As a result, a polygon is formed using 4 or 5 points which are stored in a clockwise manner starting from an upper left point.

9.2. Trapezoids

Instead of using a bounding box to enclose the entire ship, a trapezoid can be used to represent the shape of the ship more accurately. This is particularly useful for ships with irregular shapes or those that are viewed from an angle. The trapezoid can be defined by four points, where the top points represent the width of the ship at the front and back, and the bottom points represent the width of the ship at the waterline.

One alternative representation that has gained popularity in recent years is the use of trapezoid-shaped regions of interest (ROIs) to represent objects in the image. The trapezoid-shaped ROIs provide a more accurate representation of objects with irregular shapes or those viewed from an angle. The use of trapezoid-shaped ROIs has shown to improve the accuracy of object recognition and tracking in various research studies. For example, in lane recognition [52], [53] or in medical field [54], [55].

Although trapezoid-shaped ROIs have shown to be effective in representing ships in various image-based applications, they also have some limitations. One limitation is that they require additional computational resources for processing and training compared to simpler representations like bounding boxes. Additionally, the accuracy of trapezoid representations can depend on the quality and resolution of the input images. Nonetheless, trapezoid-shaped ROIs have proven to be a promising alternative to traditional representations in ship detection and classification.

Generation of trapezoids is done automatically for each bounding box where a waterline is detected. Trapezoid is formed using the interval points proposed in the waterline approach. Trapezoid is always formed using 4 points. The top points of the trapezoid represent the width of the ship at the front and back, and the bottom points represent the width of the ship at the waterline. One way to produce a trapezoid is to continue the waterline until it intersects both sides of the bounding box (see image 8a). That approach will include unnecessary water pixels inside a trapezoid. The second approach is to connect starting waterline point with the opposite lower side of the bounding box (see image 8b). In that way, the resulting lower line will sometimes cut a part of the bounding box. However, in most cases the harm is minimal, and produce trapezoids represent a ship sufficiently.

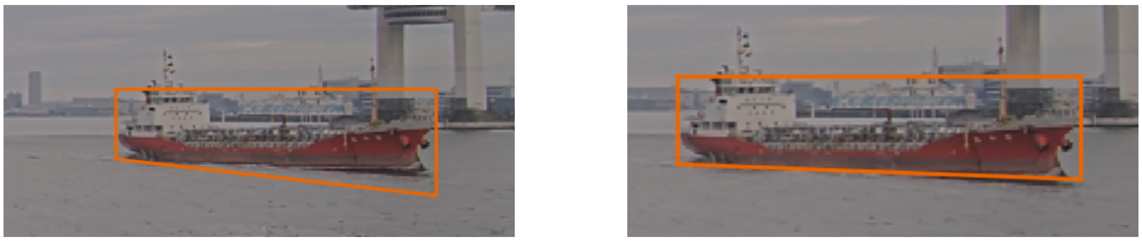

(a) Image 1. (b) Image 2.

Figure 8: Trapezoid generation example.

9.3. Cuboids

A cuboid can be used to represent the 3D shape of the ship in the image. This approach is useful for tasks that require estimating the volume of the ship or its cargo. The cuboid can be defined by six faces, where the top and bottom faces represent the length and width of the ship, and the four vertical faces represent the height of the ship.

Cuboid objects are widely used in object detection and tracking algorithms. For example, in logistics and manufacturing due to their suitability for automated handling. The reason for their widespread use lies in their regular shape and uniform dimensions, which make them easy to stack, store, and transport with minimal wastage of space. As a result, cuboid objects are an efficient and cost-effective choice for mass production and distribution processes [56], [57]. Moreover, cuboids are useful when detecting anomalies in crowded scenes [58].

Despite the advantages of using cuboids for ship representation, there are some limitations to consider. One challenge is the computational cost of processing and training with cuboids, which can be higher than with simpler representations such as bounding boxes or trapezoids. Additionally, accurately defining the six faces of the cuboid can be difficult, especially for ships with irregular shapes or those viewed from an angle. Furthermore, the generation of cuboids is a time and resource-consuming process, and it often requires a manual check of the resulting cuboids. Lastly, the success rate of automatic cuboid generation is not high, making it less practical compared to other simpler representations such as trapezoids or polygons.

As a result, while cuboids can provide informative representations, their usage can overcomplicate the underlying concept of waterline extraction in remote sensing, which aims to extract the waterline and classify objects in the image based on their location relative to the waterline. Therefore, in this thesis, the usage of cuboids was not employed in favor of simpler representations such as trapezoids or polygons, which require less computation and can still provide accurate object detection and classification results. This approach aims to streamline the object detection process and simplify the interpretation of the results, ultimately improving the efficiency and effectiveness of remote sensing applications.

9.4. Comparison metric

When deciding on the comparison metric two main things are taken into account: accuracy of the new representation and its efficiency. To measure accuracy within the problem of detection of irrelevant sea pixels following matrics are developed: percentage of pixel count of background pixels and percentage of pixel count of sea pixels. In that way it this possible to see how many irrelevant pixels are reduced using new representations compared between representations and existing bounding boxes. These metrics are developed by using ground-truth segmentation masks from the Groke300 dataset where each pixel is classified into one of 3 classes: background, sea, and sky. Then a number of background/sea pixels is calculated based on those masks within the representation. The final percentage is computed by dividing the number of background/sea pixels by the total number of pixels within the representation.

To measure the efficiency of new representations following metrics are developed: computational efficiency, training efficiency, and ease of annotation. Computational efficiency is used to measure how much time each representation takes to be produced and how much memory each representation takes. Those metrics are taken on average for 300 images and corresponding annotations. Training efficiency defines how easy are the new representations to use during training and how many new points should be predicted for each representation. Lastly, ease of annotations corresponds to manual annotation requirements and guidelines: how easy is it to learn to annotate new representation, and how easy is it to correct and validate new representation.

The summary of used comparison metrics is presented in table 6:

Metric purpose		**Metric name**
Measure accuracy	Measure how many relevant pixels are within the representation	Percentage of pixel count of background pixel
	Measure how many irrelevant pixels are reduced	Percentage of pixel count of sea pixels
Measure efficiency	Measure time to produce, memory the representation takes	Computational efficiency
	Measure how easy to use during training	Training efficiency
	Measure how easy to manually annotate	Ease of annotation

Table 6: Comparison metrics summary.

10. Object detection model training and evaluation using new annotation scheme

10.1. Training the model

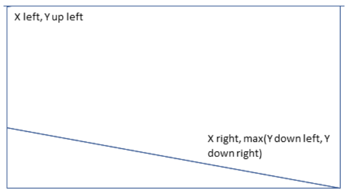
Training object detection model using new annotation scheme will include adjusting Groke Technologies' existing model parameters to support new representation. All existing model configurations are not accessible for public use and will be omitted. Only generic information will be described to demonstrate the usage of the new representation.

Training of a new object detection model based on existing configuration includes the following steps:

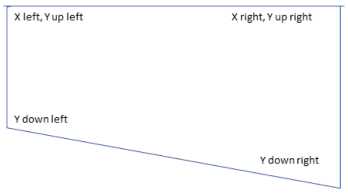
1. Modifying representation to support 3 input formats: bounding box, trapezoid, keypoints.
2. Adding transformation functions for 3 input formats
3. Changing augmentations
4. Adjusting loss
5. Training the model with the new input format

Bounding box representation uses 4 points: *X left coordinate, Y left-top coordinate, Y left-bottom coordinate, X right coordinate, Y right-top coordinate, Y right-bottom coordinate* (see figure 10a). Trapezoid representation uses 6 points: *X left coordinate, Y left-top coordinate, Y left-bottom coordinate, X right coordinate, Y right-top coordinate, Y right-bottom coordinate* (see figure 9b). Keypoints representation uses 8 points or 4 points pairs: *[X left coordinate, Y left-top coordinate], [X right coordinate, Y right-top coordinate], [X right coordinate, Y right-bottom coordinate], [X left coordinate, Y left-bottom coordinate]* (see figure 9c). Trapezoid can be represented as a bounding box by defining the bottom Y coordinate as a maximum between Y bottom left and Y bottom right coordinates: *X left coordinate, Y left coordinate, X right coordinate, maximum (Y bottom left coordinates, Y bottom right coordinate)* (see figure 9a).

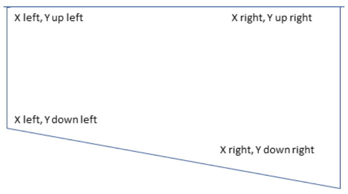
Some of the bounding boxes are not transformed into trapezoids in the waterline and trapezoid creation algorithms. Nevertheless, to use all of the data for training, it is possible to transform bounding boxes into trapezoids. If the bounding box is represented as *X left coordinate, Y left*



(a) As bounding box.



(b) As trapezoid.

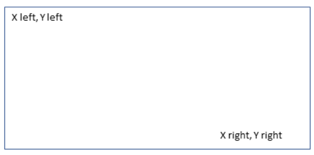


(c) As keypoints.

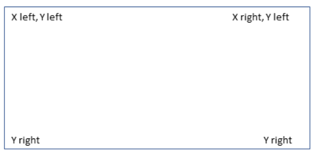
Figure 9: Trapezoid representation example.

coordinate, X right coordinate, Y right coordinate (see figure 10a), then it can be transformed in the following way: *X left coordinate, Y left coordinate, Y left coordinate, X right coordinate, Y right coordinate, Y right coordinate* (see figure 10b). And transformed into keypoints as *[X left coordinate, Y left coordinate], [X right coordinate, Y right coordinate], [X right coordinate, Y right coordinate], [X left coordinate, Y left coordinate]* (see figure 10c).

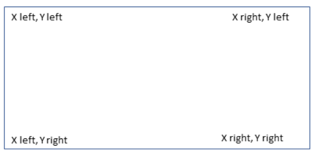
Existing augmentations are supporting bounding box input format which supports 4 points transformation. The chosen option is to use keypoints augmentations which are available in albumentations [27] module. To support augmentations for a new object representation, the trapezoids and bounding boxes are transformed into keypoints and keypoints input format is used.



(a) As bounding box.



(b) As trapezoid.



(c) As keypoints.

Figure 10: Bounding box representation example.

To support training of detections of a trapezoid representation a new loss function is needed. Several experiments are conducted to compare and demonstrate different loss options for trapezoid training. The overlap loss is a default loss commonly used in object detection tasks, it is used in the existing model and therefore is kept for trapezoid trainings. Other losses are composed using overlap loss.

The motivation behind constructing a new loss is to keep it as similar to the existing bounding box loss as possible. Since a trapezoid can be represented as a bounding box, the overlap loss is kept and corresponds to the bounding box part of a trapezoid (9a). To represent a trapezoid, an absolute difference between Y down left and Y down right points (9b) is calculated. To propagate this error, it is represented by a mean absolute error (MAE) loss that is summed up to the overlap loss. MAE loss is then added to the overlap loss to form a total loss. A modification of such loss includes multiplying MAE by 0.25 to reduce its relative contribution compared to the overlap loss. Setting the MAE coefficient to a high value makes the total loss big and the model focuses more on optimizing the coordinate accuracy rather than spatial alignment.

Another modification includes taking into account a sign difference between Y down left and Y down right points to capture the true angle of the trapezoid baseline (9a). A sign penalty is a negative penalty coefficient for negative predicted values. The penalty coefficient can be set to a number of choice, here it is set to 2. By setting the coefficient to 2, it indicates a relatively stronger penalty, but not too high in order to lower the total loss value. This choice suggests that the model should be incentivized to align its predictions more closely with the ground truth in terms of the sign of the errors. A sign penalty is added to the original MAE loss to obtain the total loss.

Experiment loss variations are described below and summarized in a table 7:

1. **Overlap + MAE**. Combining the overlap loss and MAE loss allows for a comprehensive evaluation of both spatial overlap and coordinate accuracy simultaneously. This combined loss captures both localization accuracy and spatial alignment between predicted and ground

truth bounding boxes. It helps achieve a balanced optimization objective, considering both aspects of the problem.

2. **Overlap + 0.25 MAE**. Using this loss in object detection training provides a balanced approach that considers both spatial overlap and coordinate accuracy. The inclusion of factor 0.25 plays a crucial role in ensuring an appropriate weighting between these two components.

3. **Overlap + 0.25 (MAE + sign penalty)**. This loss extends the previous combination by incorporating a sign penalty. The sign penalty takes into account the directionality of the error in bounding box predictions. It penalizes predictions that deviate in opposite directions from the ground truth, encouraging predictions that align in the same direction. Dividing the sum of overlap loss and MAE loss by 4 helps balance the overall contribution of the sign penalty.

Loss name	**Description**	**Benefit**	**Limitation**
Overlap + MAE	Sum of overlap loss and MAE loss	Simple loss that captures both bounding box and trapezoid alignment	MAE loss makes a total loss very big and hard to tune
Overlap + 0.25 MAE	Sum of overlap loss and MAE loss divided by 4	A balanced approach between overlap and MAE losses	Trapezoid baseline angle is not represented
Overlap + 0.25 (MAE + sign penalty)	Sum of overlap loss and MAE loss that is first sum up with sign loss and then divided by 4	A balanced approach that includes trapezoid baseline angle evaluation	Additional complexity to the optimization process

Table 7: Losses summary.

10.2. Evaluating the model

In order to evaluate a new annotation scheme representations it is compared to a baseline model that uses old representations. The baseline model is created by using the same subset of data for which the trapezoid representation is created but with an old bounding box representation. Since trapezoids are not created for the whole bounding box dataset (not all bounding boxes benefit from such transformation) it is important to ensure that the evaluation is fair and consistent when comparing the new annotation scheme (trapezoid representation) to the baseline model (old bounding box representation).

Models that are compared are presented in a summary in the table 8:

Model name	**Dataset description**	**Representation used for training**	**Representation used for augmentations**
Baseline model	Bounding box old annotations	Bounding box	Bounding box
New scheme model	Trapezoid new annotations	Trapezoid	Keypoints

Table 8: Models summary.

In the evaluation of a new annotation scheme representation, it is crucial to establish a fair and consistent comparison between the new approach (trapezoid representation) and the baseline model (old bounding box representation). To ensure the integrity and sensitivity of the evaluation results, it is essential to focus primarily on comparing the training, validation, and test losses. This approach enables us to directly assess the impact of the new annotation scheme on the core components of the model's training and evaluation process, providing reliable insights into its efficacy and performance as well as protecting the result data that is sensitive for Groke Technologies. The main purpose of this evaluation is to establish positive feedback on training new representation models rather than give training results. Fine-tuning of the trapezoid object detection model lies out of the scope of this thesis.

11. Results

The results of the thesis include a demonstration and evaluation of 4 research questions that can be summarised into the following categories:

1. Semantic segmentation model results
2. Waterline generation results
3. Annotation scheme results
4. Object detection model training and evaluation using new annotation scheme results

11.1. Semantic segmentation model results

11.1..1 Segmentation model results

Training experiments were performed for a combination of two datasets: MaSTr1325 and Groke300. Two metrics were used when comparing models' performance IoU and pixel accuracy.

IoU and pixel accuracy are the evaluation metrics used for segmentation models. Mean IoU of 3 classes: background, sea, and sky, and pixel accuracy is computed for the test data. Test data consists of two datasets: MaSTr1325 and Groke300. Evaluation metrics are presented for both test sets separately and the performance on the Groke300 dataset is more valuable since this is the dataset on which the segmentation model is used in the future. The best metrics are selected between each of the models for 2 test sets and therefore each test set has a best-performing model depending on a metric of choice. For the calculation of IoU and pixel accuracy, the ground-truth segmentation masks are compared with predicted masks.

A comparison of four models DeepLabv3 with Resnet50 and Resnet101 backbones and FCN with Resnet50 and Resnet101 backbone can be found in the table 9. Highlighted metrics represent the highest value in the column for each dataset and therefore indicate which model has the best performance by a particular metric.

Model name	**Test set**	**Evaluation metrics**			
		IoU background	**IoU sea**	**IoU sky**	**Pixel accuracy**
DeepLabv3 with Resnet50	MaSTr1325	0.9268	0.9914	0.9917	0.9929
	Groke300	**0.9220**	0.9560	**0.9679**	**0.9784**
DeepLabv3 with Resnet101	MaSTr1325	**0.9272**	**0.9916**	**0.9918**	**0.9930**
	Groke300	0.9178	0.9561	0.9603	0.9754
FCN with Resnet50	MaSTr1325	0.9243	0.9913	0.9913	0.9927
	Groke300	0.9122	0.9554	0.9593	0.9745
FCN with Resnet101	MaSTr1325	0.9248	0.9915	0.9916	0.9929
	Groke300	0.9210	**0.9583**	0.9659	0.9778

Table 9: Segmentation model results metrics comparison.

Example performance of the segmentation model with the best performance DeepLabv3 with Resnet101 backbone can be seen in figure 11.

Next, the DeepLabv3 with Resnet101 performance was visualized using histograms and boxplots. Histograms and boxplots are a graphical representation of the distribution of a dataset. Histograms display the frequency of data points falling within certain ranges. Boxplots, on the other hand, provide a way to visualize the spread and central tendency of a dataset, containing the median, quartiles, and outliers of a dataset. Both graphs show outliers of the data, and its general tendency and can provide insights into areas where the model shows accurate or inaccurate results. Figures 12a, 12b show histogram and boxplot of the class 0 which corresponds to the background class in semantic segmentation. Figures 12c, 12d display histogram and boxplot of class 1 which

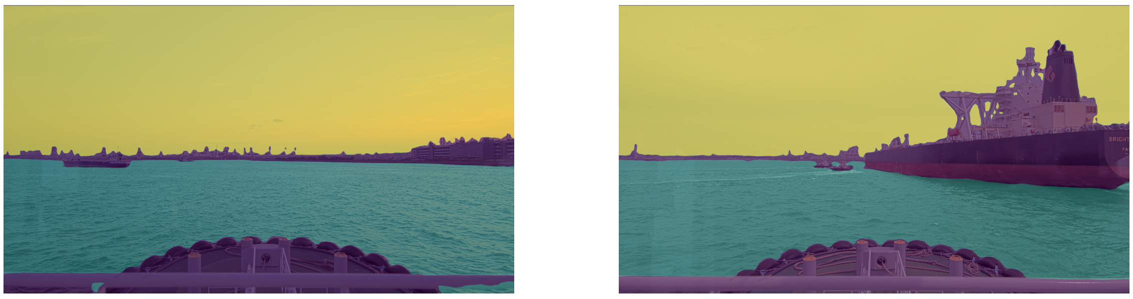
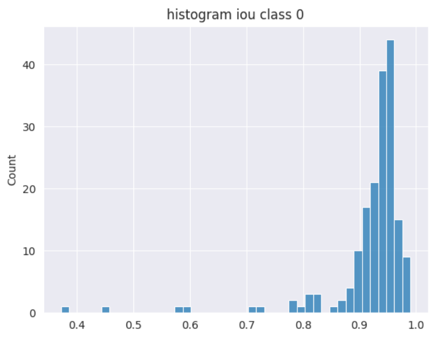

(a) Image 1. (b) Image 2.

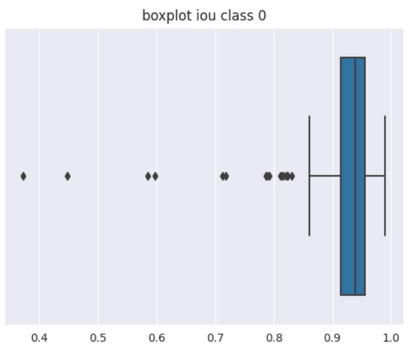
Figure 11: Predicted segmentation classes example.

corresponds to the sea class and histogram and boxplot on figures 12e, 12f correspond to class 2 or sky class. Lastly, figures 12g, 12h contain histogram and boxplot for pixel accuracy.

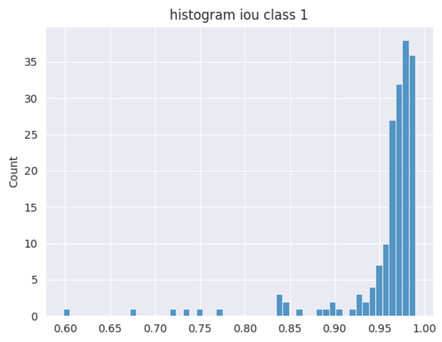
Boxplots and histograms are another evaluation metric and are representing the overall performance of the models. The mean value can hide the potential loss performance of the model, whereas boxplot and histogram show performance for each image. The best-performing model according to boxplot and histogram analysis includes plots where most of the data is distributed in the 0.8-1.0/1.0 accuracy range and that has a minimal number of outliers. Outliers correspond to a low accuracy for several images in a test set.



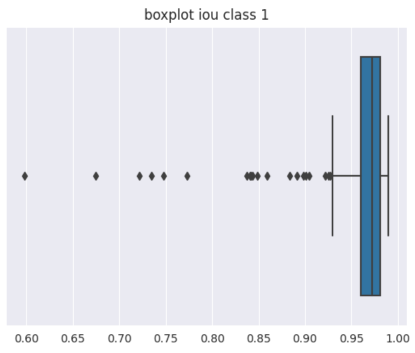
(a) Histogram class background.



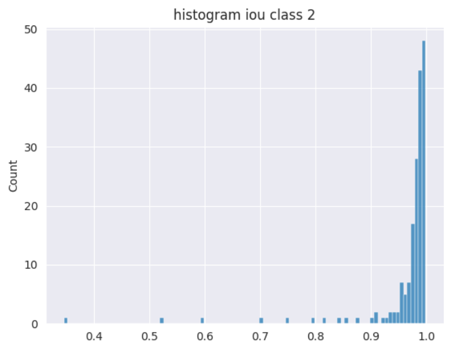
(b) Boxplot class background.



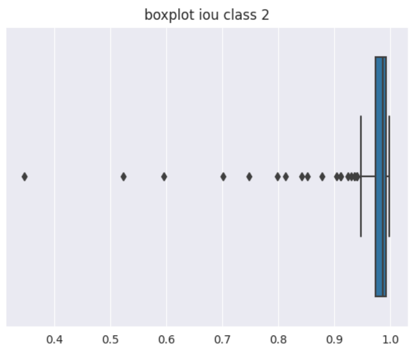
(c) Histogram class sea.



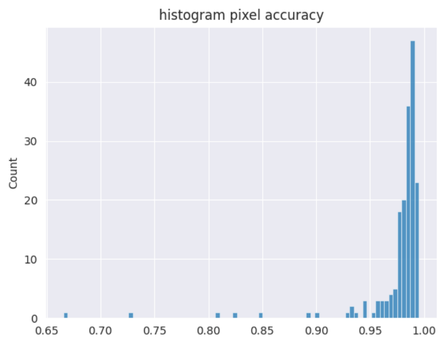
(d) Boxplot class sea.



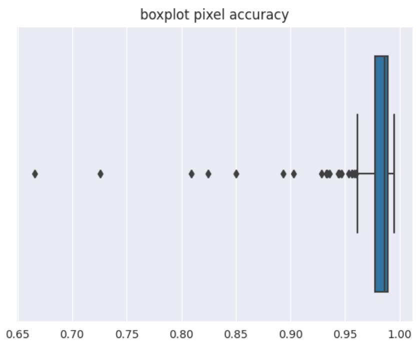
(e) Histogram class sky.



(f) Boxplot class sky.



(g) Histogram for pixel accuracy.



(h) Boxplot for pixel accuracy.

Figure 12: Histograms and boxplots for DeepLabv3 with Resnet101.

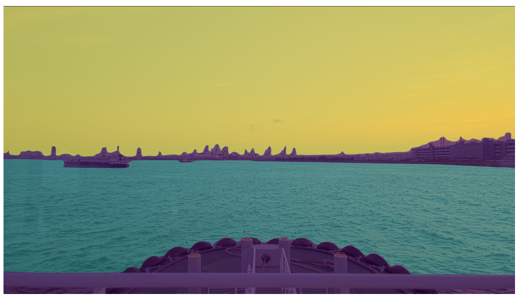
11.1..2 Segmentation model patched results

The segmentation model patched evaluation includes the same metrics and logic as the original segmentation model. A comparison of four models DeepLabv3 with Resnet50 and Resnet101 backbones and FCN with Resnet50 and Resnet101 backbone can be found in the table 10. Highlighted metrics represent the highest value in the column for each dataset and therefore indicate which model has the best performance by a particular metric.

Model name	**Test set**	**Evaluation metrics**			
		IoU background	**IoU sea**	**IoU sky**	**Pixel accuracy**
DeepLabv3 with Resnet50	MaSTr1325	**0.9166**	**0.9868**	**0.9898**	**0.9908**
	Groke300	**0.9669**	**0.9838**	**0.9929**	**0.9925**
DeepLabv3 with Resnet101	MaSTr1325	0.9106	0.9860	0.9894	0.9899
	Groke300	0.9648	0.9829	0.9923	0.9919
FCN with Resnet50	MaSTr1325	0.9034	0.9837	0.9886	0.9889
	Groke300	0.9497	0.9715	0.9814	0.9838
FCN with Resnet101	MaSTr1325	0.9094	0.9860	0.9893	0.9899
	Groke300	0.9606	0.9783	0.9896	0.9900

Table 10: Segmentation model patched results metrics comparison.

Example performance of the segmentation model with the best performance DeepLabv3 with Resnet101 backbone can be seen in figure 13 where pixels are classified into 3 classes (purple pixels - background, blue - sea/water, yellow - sky).


(a) Image 1.

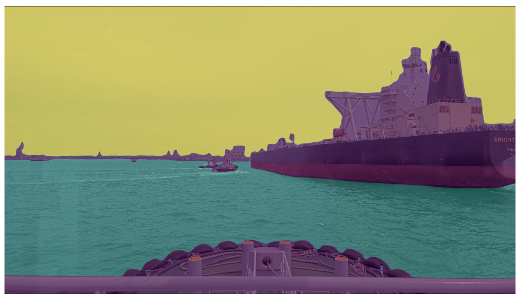

(b) Image 2.

Figure 13: Predicted segmentation classes example.

11.2. Waterline generation results

The proposed waterline generation algorithm consists of three parts:

1. Detecting waterpoints. This step relies on the original bounding box and segmentation mask detected by the computer vision model. Visualization of detected waterpoints between ship and sea within a bounding box is displayed in figure 14.
2. Generating waterlines. This step relies on previously detected waterpoints and uses Linear Regression to draw a line. Visualization of detected waterlines between ship and sea within a bounding box is displayed in figure 15.
3. Filtering generated waterlines. This step relies on statistical analysis and experiments evaluation. Visualization of filtered cases is provided in figure 7.

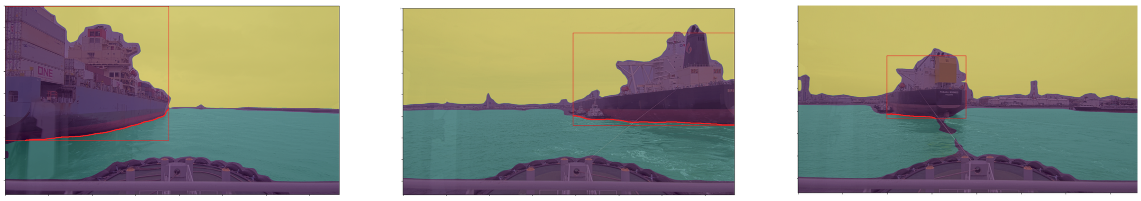

(a) Image 1. (b) Image 2. (c) Image 3.

Figure 14: Water points detection example.

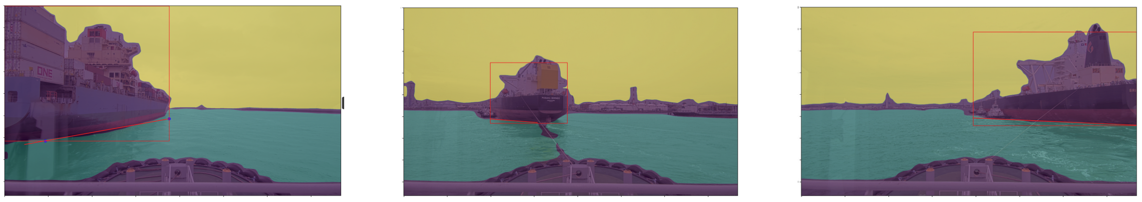
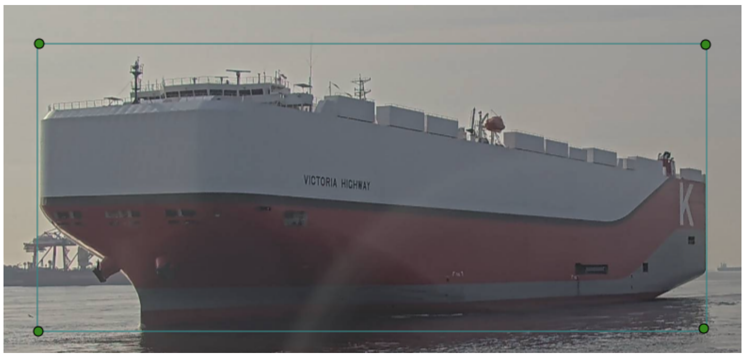

(a) Image 1. (b) Image 2. (c) Image 3.

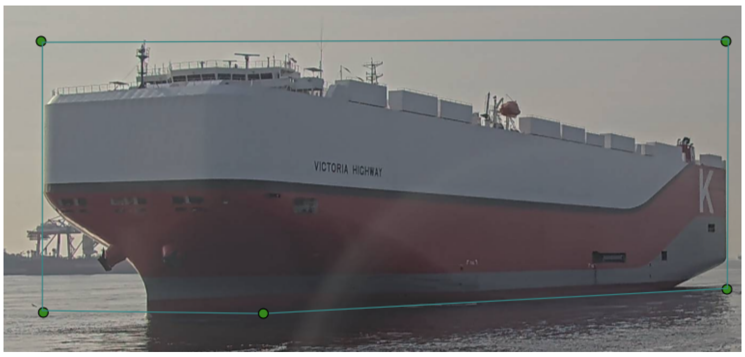
Figure 15: Waterlines detection example.

11.3. Annotation scheme results

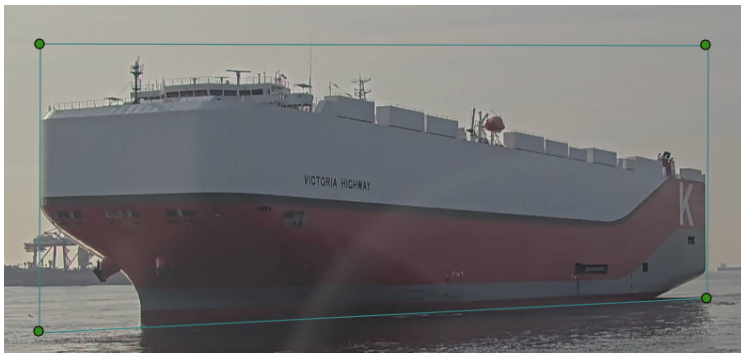
Three new object representations are generated: polygon, trapezoid, and cuboid. Polygon and trapezoids are generated automatically based on a specific algorithm that relies on detected waterlines. Due to their complexity, cuboids are not generated automatically, but manually for demonstration purposes. Visual comparison of the original bounding box, automatically generated polygon and trapezoid and manually drawn cuboid are presented in figure 16.



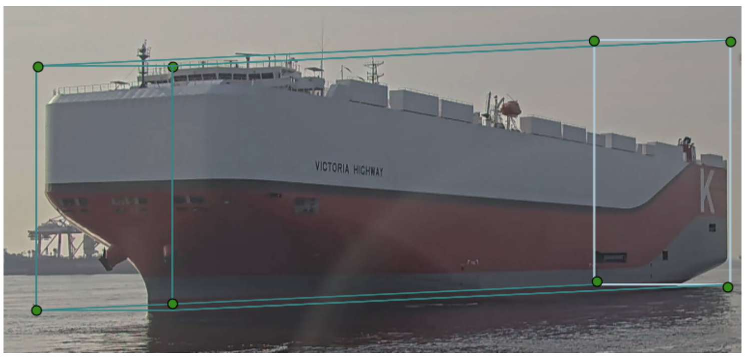
(a) Bounding box.



(b) Polygon.



(c) Trapezoid.



(d) Cuboid.

Figure 16: Annotation schemes results.

The comparison metric results are obtained by calculating mean and median values for each of the metrics. In the evaluation of the new bounding box representation, computational efficiency is assessed by measuring the time and memory required for its generation. These metrics are calculated as averages across 300 images and their annotations. Training efficiency refers to the ease of using the new representation during the training process, including the number of points that need to be predicted. Ease of annotations relates to the manual annotation process, considering factors such as the ease of learning to annotate the new representation, as well as correcting and validating it. Bounding box generation time is not taken into account since it is not generated automatically. Training efficiency and ease of use and annotation are generalized based on Groke Technologies' domain expertise. Results are presented in a table 11 where highlighted values represent the best result among columns (object representations) for each row.

Annotation scheme name		**Bounding box**	**Polygon**	**Trapezoid**
Percentage of background pixels	Mean	79.8826	82.9713	**83.0446**
	Median	80.9908	84.4427	**85.1863**
Percentage of sea pixels	Mean	9.9476	6.5387	**6.2761**
	Median	6.3148	1.7831	**1.4262**
Computational efficiency	Memory	**3.5078 Mb**	3.7159 Mb	3.5711 Mb
	Time	-	1.03036 sec	**1.0283 sec**
Training efficiency		**Easy: 4 points**	Hard: 8 points	Medium: 5-6 points
Ease of use/annotation		**Easy-Medium**	Medium-Hard	Medium

Table 11: Annotation scheme comparison metrics.

11.4. Object detection model training and evaluation using new annotation scheme results

Results of model evaluation include train, validation, and test loss comparison. This comparison aims to establish an evaluation of the training efficiency of the new object representation compared to the baseline model for bounding boxes. First, experimental models are compared and results are presented in table 12. Then, two models are compared: the baseline model that is trained with bounding boxes and Overlap + 0.25 MAE as a model with the lowest validation and test losses. The results are displayed in table 13.

Loss name	**Train loss**	**Validation loss**	**Test loss**
Overlap + MAE	344.9 (overlap 334.4)	953.5 (overlap 895.4)	917.8(overlap 864.3)
Overlap + 0.25 MAE	326.2 (overlap 318.7)	554.6 (overlap 493.4)	549.7 (overlap 490.3)
Overlap + 0.25 * (MAE + sign penalty)	353.8 (overlap 346.3)	636.8 (overlap 575.5)	635.5 (overlap 576.2)

Table 12: Loss results comparison.

Model name	**Loss name**	**Train loss**	**Validation loss**	**Test loss**
Baseline model	Overlap	204.6 (overlap 140)	407.1 (overlap 194.9)	391(overlap 189)
Trapezoid model	MAE + sign penalty	353.8 (overlap 346.3)	636.8 (overlap 575.5)	635.5 (overlap 576.2)

Table 13: Model comparison.

12. Discussion

In this thesis, the research questions were divided into four categories, namely semantic segmentation model results, waterline generation results, annotation scheme results, and object detection model training and evaluation using the new annotation scheme.

Reflecting on the research questions:

1. How to use a semantic segmentation model to find bounding boxes that need correction?

   The results of the thesis demonstrate the successful training of a semantic segmentation model specifically designed for maritime data. This achievement is crucial for accurately identifying ship regions within the dataset. The results demonstrate that segmentation models can be successfully trained even with a limited amount of data. Patched segmentation model training demonstrated a sufficient way to train a model with limited computational power. The biggest limitation here is unseen data and the generalization of results. Since the detection model is trained on a specific relatively small dataset, the performance on unseen data can be unpredictable. Moreover, this approach might not work for a different segmentation task and can require additional experiments.

2. How to develop a method to generate a new bounding box representation that will draw a waterline between the ship and sea?

   The thesis presents a method that effectively identifies the border points between water and ship within bounding boxes. This step is essential for generating accurate waterlines. Using waterpoints the thesis successfully generates waterlines based on the identified border points. This achievement allows for the creation of a new representation that visually separates the ship from the sea. The thesis establishes a set of filters that ensure the relevance and quality of the generated representations. These filters help eliminate meaningless waterlines and improve the accuracy of the representation. Waterline generation is a new approach in the field of ship representation and holds significant potential for various applications. The successful production of waterlines based on the identified border points showcases the feasibility and practicality of this approach. The limitation lies in existing bounding boxes requirement and the algorithm might not perform well on poor bounding box annotations. In addition to this, scaling of this method might require further fine-tuning of the algorithm.

3. How to choose the best annotation scheme for found bounding boxes?

   Multiple ship representations are created, and specific criteria are defined to evaluate their effectiveness. This comprehensive analysis allows for the assessment and comparison of different representation approaches. It is demonstrated that usage of the new annotation scheme can reduce water pixels percentage up to less than 1.5 percent while not requiring much computational power and extra points predictions. Several ship representations are presented to be picked for various implications. Moreover, presented annotations schemes can be used outside the maritime industry field for any object that needs shape correction. The main bottleneck of this approach is the simplicity of chosen representations. While they are suitable to solve ship distance assessment challenge, this approach might not be applicable to a more complex task. Also, new representations still require manual visual assessment and possible retraining of the data annotators to be able to use those annotation schemes in practice.

4. How to evaluate a model that predicts new bounding box representation?

   The thesis successfully trains an object detection model utilizing the new ship representation. This achievement demonstrates the compatibility and applicability of the generated representation in the training process. The thesis compares the performance of the object detection model using the new ship representations against the traditional bounding box approach. The evaluation is done using loss comparison between new bounding box representation model and old bounding box model. This evaluation allows for a thorough assessment of the training process offered by the new representations. However, the object detection model is not optimized for new representation detection and this step lies out of the scope of the thesis.

Based on the achieved goals and the results obtained, it can be concluded that the research questions have been adequately addressed. The thesis successfully develops a method to generate waterlines, establishes evaluation criteria, and demonstrates the effectiveness of the new representation approach. These findings contribute to enhancing the accuracy and relevance of ship detection and analysis in maritime datasets.

The implications of these results are significant, as they offer improved techniques for ship representation and detection in various maritime applications. The accurate identification of ship regions and the precise detection of waterlines provide valuable insights for tasks such as maritime surveillance, vessel tracking, and maritime research.

Recommendations for future work could include exploring additional variations of the representation approach, investigating the generalization of the method to other maritime datasets, and further optimizing the object detection model using the new representations. Additionally, considering the potential impact of the research, collaboration with industry stakeholders or relevant authorities can help facilitate the integration of the proposed techniques into practical maritime systems or applications.

13. Conclusions and future work

The focus of this thesis is to address the limitations of using bounding boxes in computer vision for ship detection in the maritime environment using machine learning techniques. Throughout the research, various goals were achieved, and the results have significant implications in the field of ship representation and detection.

The method successfully identifies the border points between water and ship within bounding boxes, which is a crucial step in generating accurate waterlines. The production of waterlines based on these border points demonstrates the feasibility and practicality of this approach. The derived set of filters ensures that the generated representations are relevant and reliable, filtering out erroneous or irrelevant annotations. The evaluation of the object detection model using the new ship representations compared to the traditional bounding box approach revealed training possibilities, highlighting the effectiveness of the proposed method in enhancing ship detection accuracy and overall system performance.

The significance of these results lies in their potential impact on various maritime applications. Accurate waterline depiction enables precise analysis of maritime data, such as ship behavior tracking, maritime boundary delineation, and environmental monitoring. This has implications for industries such as maritime security, coastal management, and marine research, where accurate ship representation is crucial for informed decision-making and situational awareness.

In conclusion, the research has successfully addressed the stated research questions and achieved the set goals. The method for generating the new bounding box representation has demonstrated its effectiveness and applicability in accurately separating the ship from the sea with waterlines.

Moving forward, there are several possibilities for further exploration. New representations can be refined and optimized to handle diverse maritime datasets and environmental conditions. Additionally, optimizing training for new representation is a whole new area for experiments. Collaborations with domain experts and industry practitioners can provide valuable insights and facilitate the practical implementation of the proposed method in real-world maritime applications.

References

[1] Groke technologies. Accessed on: 2023-03-17. [Online]. Available: https://www.groke-tech.com/

[2] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," 2022.

[3] Ultralytics. (2023) Yolov8. Accessed on: 2023-04-23. [Online]. Available: https://github.com/ultralytics/ultralytics

[4] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," 2023.

[5] P. Hurtik, V. Molek, J. Hula, M. Vajgl, P. Vlasanek, and T. Nejezchleba. (2022) Poly-yolo: higher speed, more precise detection and instance segmentation for yolov3. Accessed on: 2023-01-31. [Online]. Available: https://doi.org/10.1007/s00521-021-05978-9

[6] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. (2021) Deformable detr: Deformable transformers for end-to-end object detection. Accessed on: 2023-01-31. [Online]. Available: http://export.arxiv.org/pdf/2010.04159

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

[8] B. Baheti, S. Innani, S. Gajre, and S. Talbar, "Semantic scene segmentation in unstructured environment with modified deeplabv3+," *Pattern Recognition Letters*, vol. 138, pp. 223–229, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865520302750

[9] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun. (2018) Detnet: A backbone network for object detection. Accessed on: 2023-01-31. [Online]. Available: http://export.arxiv.org/pdf/1804.06215

[10] X. Zhou, J. Zhuo, and P. Krähenbühl, "Bottom-up object detection by grouping extreme and center points," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 850–859.

[11] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 6568–6577.

[12] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9626–9635.

[13] X. Yan, Z. Chen, A. Xu, X. Wang, X. Liang, and L. Lin, "Meta r-cnn: Towards general solver for instance-level low-shot learning," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9576–9585.

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2016.

[16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016.

[17] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, pp. 21–37. [Online]. Available: https://doi.org/10.1007%2F978-3-319-46448-0_2

[18] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *CoRR*, vol. abs/1411.4038, 2014. [Online]. Available: http://arxiv.org/abs/1411.4038

[19] ——, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.

[20] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, "Fully convolutional instance-aware semantic segmentation," 2017.

[21] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," 2016.

[22] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021.

[23] S. Niu, Y. Liu, J. Wang, and H. Song, "A decade survey of transfer learning (2010–2020)," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 151–166, 2020.

[24] L. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *CoRR*, vol. abs/1706.05587, 2017. [Online]. Available: http://arxiv.org/abs/1706.05587

[25] A. Howard, M. Sandler, G. Chu, L. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," *CoRR*, vol. abs/1905.02244, 2019. [Online]. Available: http://arxiv.org/abs/1905.02244

[26] O. Elharrouss, Y. Akbari, N. Almaadeed, and S. Al-Maadeed, "Backbones-review: Feature extraction networks for deep learning and deep reinforcement learning approaches," 2022.

[27] Albumentations. Accessed on: 2023-03-17. [Online]. Available: https://albumentations.ai/

[28] F. Hutter, H. Hoos, and K. Leyton-Brown, "An efficient approach for assessing hyperparameter importance," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 1. Bejing, China: PMLR, 22–24 Jun 2014, pp. 754–762. [Online]. Available: https://proceedings.mlr.press/v32/hutter14.html

[29] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016, http://www.deeplearningbook.org.

[30] E. Tiu. (2019) Metrics to evaluate your semantic segmentation model. Accessed on: 2023-05-28. [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model-6bcb99639aa2

[31] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013. [Online]. Available: https://faculty.marshall.usc.edu/gareth-james/ISL/

[32] J. Yu, Y. Zhang, S. Liu, S. Du, and X. Lan, "A review of object detection based on deep learning." *Multimed Tools Appl*, no. 79, 2020, accessed on: 2023-06-14. [Online]. Available: https://doi.org/10.1007/s11042-020-08976-6

[33] S. Wu, J. Yang, X. Wang, and X. Li, "Iou-balanced loss functions for single-stage object detection," *Pattern Recognition Letters*, vol. 156, pp. 96–103, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865522000289

[34] A. Kirillov, R. Girshick, K. He, and P. Dollár, “Panoptic feature pyramid networks,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 6392–6401.

[35] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, “Hybrid task cascade for instance segmentation,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4969–4978.

[36] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li, and J. Shi, “Foveabox: Beyound anchor-based object detection,” *IEEE Transactions on Image Processing*, vol. 29, pp. 7389–7398, 2020.

[37] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez. (2017) A review on deep learning techniques applied to semantic segmentation. Accessed on: 2023-01-31. [Online]. Available: http://export.arxiv.org/pdf/1704.06857

[38] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, “Understanding convolution for semantic segmentation,” in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 1451–1460.

[39] C. Holder and M. Shafique. (2022) On efficient real-time semantic segmentation: A survey. Accessed on: 2023-01-31. [Online]. Available: https://arxiv.org/abs/2206.08605

[40] J. Ma, Y. Ushiku, and M. Sagara, “The effect of improving annotation quality on object detection datasets: A preliminary study,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2022, pp. 4850–4859.

[41] J.-H. Kim, N. Kim, Y. W. Park, and C. S. Won, “Object detection and classification based on yolo-v5 with improved maritime dataset,” *Journal of Marine Science and Engineering*, vol. 10, no. 3, 2022. [Online]. Available: https://www.mdpi.com/2077-1312/10/3/377

[42] J. Zhang, C. Chatzichristos, K. Vandecasteele, L. Swinnen, V. Broux, E. Cleeren, W. V. Paesschen, and M. D. Vos, “Automatic annotation correction for wearable eeg based epileptic seizure detection,” *Journal of Neural Engineering*, vol. 19, no. 1, p. 016038, feb 2022. [Online]. Available: https://dx.doi.org/10.1088/1741-2552/ac54c1

[43] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: common objects in context,” *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[44] B. Bovcon, J. Muhovič, J. Perš, and M. Kristan, “The mastr1325 dataset for training deep usv obstacle detection models,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019.

[45] Models and pre-trained weights - torchvision 0.15 documentation. Accessed on: 2023-03-17. [Online]. Available: https://pytorch.org/vision/stable/models.html

[46] J. Brownlee. (2019) How to configure the learning rate when training deep learning neural networks. Accessed on: 2023-05-28. [Online]. Available: https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks/

[47] Deeplabv3 example. Accessed on: 2023-03-17. [Online]. Available: https://pytorch.org/hub/pytorch_vision_deeplabv3_resnet101/

[48] J. Xu, Z. Liu, C. Yang, L. Li, and Y. Pei, “A pseudo-distance algorithm for collision detection of manipulators using convex-plane-polygons-based representation,” *Robotics and Computer-Integrated Manufacturing*, vol. 66, p. 101993, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584520302040

[49] B. Xu, J. Xu, N. Xue, and G.-S. Xia, “Hisup: Accurate polygonal mapping of buildings in satellite imagery with hierarchical supervision,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 198, pp. 284–296, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0924271623000667

[50] H. Perreault, G.-A. Bilodeau, N. Saunier, and M. Héritier, "Centerpoly: Real-time instance segmentation using bounding polygons," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, October 2021, pp. 2982–2991.

[51] G. Faure, H. Perreault, G. Bilodeau, and N. Saunier, "Polytrack: Tracking with bounding polygons," *CoRR*, vol. abs/2111.01606, 2021. [Online]. Available: https://arxiv.org/abs/2111.01606

[52] Tanviruzzama and S. Mehfuz, "Review: Lane detection for autonomous vehicles using image processing techniques," in *2023 International Conference on Power, Instrumentation, Energy and Control (PIECON)*, 2023, pp. 1–6.

[53] R. Agrawal and N. Singh, "Lane detection and collision prevention system for automated vehicles," in *Applied Computer Vision and Image Processing*, B. Iyer, A. M. Rajurkar, and V. Gudivada, Eds. Singapore: Springer Singapore, 2020, pp. 46–59.

[54] S. M. Lee, M. J. Kim, J. H. Yoon, W. Hong, and H. I. Ha, "Comparison of point and 2-dimensional shear wave elastography for the evaluation of liver fibrosis." *Ultrasonography*, 2020. [Online]. Available: https://doi.org/10.14366/usg.19090

[55] S. Mohaghegh, H. Mohammad-Rahimi, L. Eslamian, A. Ebadifar, and M. R. Badiee, "Effect of mesenchymal stem cells injection and low-level laser therapy on bone formation after rapid maxillary expansion: an animal study." *American journal of stem cells*, 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7811931/

[56] H. Hu, F. Immel, J. Janosovits, M. Lauer, and C. Stiller, "A cuboid detection and tracking system using a multi rgbd camera setup for intelligent manipulation and logistics," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, 2021, pp. 1097–1103.

[57] A. Naumann, L. Dörr, N. Ole Salscheider, and K. Furmans, "Refined plane segmentation for cuboid-shaped objects by leveraging edge detection," in *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2020, pp. 432–437.

[58] Z. ping Hu, L. Zhang, S. fang Li, and D. gang Sun, "Parallel spatial-temporal convolutional neural networks for anomaly detection and location in crowded scenes," *Journal of Visual Communication and Image Representation*, vol. 67, p. 102765, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1047320320300158