Real-time Ship Recognition and Georeferencing for the Improvement of Maritime Situational Awareness

Borja Carrillo Perez

A Thesis submitted for the degree of Doctor of Engineering (Dr.-Ing.)
Faculty 3 – Mathematics and Computer Science, University of Bremen
2024

This doctoral thesis is a compilation of six articles that I authored and co-authored between 2021 and 2024. The research was conducted at the Institute for the Protection of Maritime Infrastructures of the German Aerospace Center (DLR), in Bremerhaven, Germany.

Date of the Defense: 09.08.2024

Examiners:

Prof. Dr.-Ing. Udo Frese (University of Bremen)

Prof. Dr.-Ing. Tobias Meisen (University of Wuppertal)

Further members of the examination committee:

Prof. Dr. Ralf Bachmayer (University of Bremen)

Dr. rer. nat. Enno Peters (German Aerospace Center)

Arne Hasselbring (University of Bremen)

Ayleen Lührsen (University of Bremen)

Declaration

I, Borja Carrillo Perez, declare that this thesis, titled "Real-time Ship Recognition and Georeferencing for the Improvement of Maritime Situational Awareness", is my own work and has not been submitted for any other degree or qualification at this or any other institution. All sources are acknowledged and referenced.

Borja Carrillo Perez

June 2024

Abstract

In an era where maritime infrastructures are paramount supports of societies, the need for advanced maritime situational awareness solutions has become increasingly important. Existing ship monitoring procedures, such as the Automatic Identification System (AIS), have limitations, suffer from delayed updates and are vulnerable to cyberattacks. Other technologies, such as satellite imagery and radar, face challenges in real-time applications due to delays in acquiring and processing data. The use of optical camera systems and image processing can improve situational awareness, allowing real-time usage of maritime infrastructure footage. However, the number of video streams available poses a challenge for maritime operators, who could be helped by summarized spatial information of recognized ships, irrespective of their size and type, and presented on a map in real-time. This motivates the development of automated ship recognition and georeferencing technologies. Moreover, the deployment of such camera systems, equipped with an embedded device, allows for local data processing on the edge to minimize network demand, energy usage, decrease latency, cut costs, and enhance data protection.

This thesis, integrating six of my publications, presents a comprehensive investigation into leveraging deep learning and computer vision to advance the research in real-time ship recognition and georeferencing for the improvement of maritime situational awareness. I present a novel dataset for ship recognition and georeferencing, ShipSG, which facilitates the development and validation of recognition and georeferencing methodologies. The dataset contains 3505 images and 11625 ship masks with their corresponding class, geographic position and length. Through a series of studies of state-of-the-art deep-learning-based object recognition algorithms, I introduce a custom real-time segmentation architecture, ScatYOLOv8+CBAM. This architecture was created and optimized for the NVIDIA Jetson AGX Xavier as embedded system. ScatYOLOv8+CBAM incorporates the 2D scattering transform, a novel addition that enhances YOLOv8 in real-world applications such as ship segmentation. Additionally, the performance is further improved with the integration of attention mechanisms. The proposed architecture exceeds in more than 5% the performance of state-of-the-art methods, achieving a mean Average Precision (mAP) of 75.46%. The inference speed, once the customized architecture is deployed on the embedded system using TensorRT, is of 25.3 ms per frame. Furthermore, I address the need for precision in recognizing small and distant ships and their real-time processing of full-resolution

images on embedded systems, with an enhanced slicing mechanism that performs batch inference and merges predictions, achieving mAP improvements ranging from 8% to 11%. The recognized ships are georeferenced using my proposed method, which automatically calculates the georeferencing pixel of the recognized ships, and uses homographies to provide the geographic position of ships from single images, without prior camera knowledge. In the quantitative analysis, the georeferencing method achieved a positioning error of 18 m $\pm$ 10 m for ranges inside the port basin (up to 400 m) and 44 m $\pm$ 27 m outside (from 400 m to 1200 m). The main findings reveal significant advancements in maritime situational awareness with the practical demonstration of the applicability of the methodologies in real-world scenarios, such as the detection of abnormal ship behaviour, camera integrity assessment and 3D reconstruction. The approach not only outperforms existing methods in terms of accuracy and processing speed but also provides a framework for seamlessly integrating recognized and georeferenced ships into real-time systems, enhancing operational effectiveness and decision-making for maritime authorities. The integration of these methodologies into embedded systems represents a pivotal advancement in the domain, offering a scalable and efficient solution for improving maritime situational awareness and response capabilities. This thesis contributes to the maritime computer vision field by establishing a benchmark for ship segmentation and georeferencing research, demonstrating the viability of deep-learning-based recognition and georeferencing methods for real-time maritime monitoring.

Zusammenfassung

In einer Ära, in der maritime Infrastrukturen von größter Bedeutung für Gesellschaften sind, ist der Bedarf an fortschrittlichen Lösungen zur maritimen Lageerkennung zunehmend wichtiger geworden. Bestehende Verfahrung zur Schiffsbeobachtung wie das Automatic Identification System (AIS) haben Einschränkungen, leiden unter verzögerten Aktualisierungen und sind anfällig für Cyberangriffe. Andere Technologien, wie Satellitenbilder und Radar, haben Schwierigkeiten bei Echtzeitanwendungen aufgrund von Verzögerungen bei der Erfassung und Verarbeitung von Daten. Der Einsatz von optischen Kamerasystemen und Computer Vision kann das Situationsbewusstsein verbessern, indem sie die Echtzeitnutzung von Aufnahmen direkt an marititmen Infrastrukturen ermöglichen. Die große Anzahl verfügbarer Videostreams stellt jedoch eine Herausforderung für maritime Betreiber dar, die durch zusammengefasste, räumliche Informationen über erkannte Schiffe, unabhängig von ihrer Größe und Art, auf einer Karte in Echtzeit unterstützt werden könnten. Dies motiviert die Entwicklung automatisierter Technologien zur Schiffsidentifikation und Georeferenzierung. Darüber hinaus ermöglicht der Einsatz von Kamerasystemen zusammen mit Embedded Systems, die lokale Datenverarbeitung in situ, um den Netzwerkbedarf zu minimieren, den Energieverbrauch zu senken, die Latenz zu verringern, die Kosten zu senken und den Datenschutz zu verbessern.

Diese Dissertation, die sechs meiner Veröffentlichungen bündelt, präsentiert eine umfassende Untersuchung zur Nutzung von Deep Learning und Computer Vision, um die Forschung zur Echtzeit-Schiffsidentifikation und Georeferenzierung zur Verbesserung des maritimen Situationsbewusstseins voranzutreiben. Ich präsentiere einen neuartigen Datensatz für die Schiffsidentifikation und Georeferenzierung, ShipSG, der die Entwicklung und Validierung von Identifikations- und Georeferenzierungsmethoden erleichtert. Der Datensatz enthält 3505 Bilder und 11625 Schiffsmasken mit entsprechenden Klassen, geografischer Position und Länge. Durch eine Reihe von Studien zu den neuesten Objekterkennungsalgorithmen basierend auf Deep-Learning stelle ich eine neuartige Echtzeit-Segmentierungsarchitektur vor, ScatYOLOv8+CBAM. Diese Architektur wurde speziell für den NVIDIA Jetson AGX Xavier als eingebettetes System entwickelt und optimiert. ScatYOLOv8+CBAM integriert die 2D-Streutransformation, eine neuartige Ergänzung, die YOLOv8 in realen Anwendungen wie der Schiffsegmentierung verbessert. Zudem wird die Leistung durch die Integration von Attention-Mechanismen weiter gesteigert. Die vorgeschlagene Architek-

tur übertrifft neueste Methoden um mehr als 5% und erreicht eine Mean-Average-Precision (mAP) von 75.46%. Die Inferenzlaufzeit beträgt 25.3 ms pro Frame, sobald die angepasste Architektur auf dem eingebetteten System mit TensorRT bereitgestellt ist. Darüber hinaus gehe ich auf die Notwendigkeit der Präzision bei der Erkennung kleiner und entfernter Schiffe und ihrer Echtzeitverarbeitung von Bildern in voller Auflösung auf eingebetteten Systemen ein. Hierzu betrachte ich einen verbesserten Slicing-Mechanismus, der Batch-Inferenz durchführt und Vorhersagen zusammenführt, was letztlich in mean Average Precision (mAP)-Verbesserungen von 8% bis 11% resultiert. Die erkannten Schiffe werden mittels meiner vorgeschlagenen Methode georeferenziert. Dazu wird automatisch das Georeferenzierungspixel der erkannten Schiffe berechnet und Homographien verwendet, um die geografische Position der Schiffe aus Einzelbildern ohne vorherige Kamerakenntnisse zu bestimmen. In der quantitativen Analyse erreichte die Georeferenzierungsmethode einen Positionierungsfehler von 18 $m \pm 10$ m für Entfernungen innerhalb des betrachtetetn Hafenbeckens (bis zu 400 m) und 44 $m \pm 27$ m außerhalb (von 400 m bis 1200 m). Die Hauptresultate zeigen erhebliche Fortschritte im maritimen Situationsbewusstsein, welche anhand von praktischen Beispielen der Anwendbarkeit der Methoden in realen Szenarien demonstriert werden. Zu diesen Beispielen gehören die Erkennung von abnormalem Schiffsverhalten, die Bewertung der Kameraintegrität als auch die 3D-Rekonstruktion von Schiffen. Der Ansatz übertrifft nicht nur bestehende Methoden in Bezug auf Genauigkeit und Laufzeit, sondern bietet auch ein Framework für die nahtlose Integration erkannter und georeferenzierter Schiffe in Echtzeitsysteme, wodurch die operative Effizienz und Entscheidungsfindung für maritime Behörden verbessert werden kann. Die Integration dieser Methoden in eingebettete Systeme stellt einen entscheidenden Fortschritt in diesem Anwendungsbereich dar und bietet eine skalierbare und effiziente Lösung zur Verbesserung des maritimen Situationsbewusstseins und der Reaktionsfähigkeiten. Diese Dissertation trägt zum Bereich der maritimen Computer Vision bei, indem sie eine Benchmark für die Forschung zur Schiffssegmentierung und Georeferenzierung etabliert und die Machbarkeit von auf Deep-Learning basierenden Erkennungs- und Georeferenzierungsmethoden für die Echtzeitüberwachung maritimer Umgebungen demonstriert.

Acknowledgements

This academic and personal achievement would not have been possible without the support I have received over these past years. I am profoundly grateful to everyone who has contributed.

Firstly, I extend my deepest thanks to Dr. Sarah Barnes and Dr. Maurice Stephan from the German Aerospace Center for their invaluable guidance, insightful discussions, commitment, and time. I feel very lucky to have had the opportunity to learn from them, and their mentorship has profoundly shaped my professional and personal growth.

I am also grateful to Prof. Dr.-Ing. Udo Frese from the University of Bremen for his guidance and advice, especially during the final stages of this work, which significantly elevated the quality of my research. I appreciate Prof. Dr.-Ing. Tobias Meisen from the University of Wuppertal for his time and willingness to review this thesis, and I thank the committee members for their participation in the defense.

I would like to thank Jens-Michael Schlüter and Michael Busack from the Alfred Wegener Institute for Polar and Marine Research, as well as Marco Gawehns and Tino Flenker from the German Aerospace Center, for their technical support in the data acquisition process. Additionally, I am grateful to Dr. Michael Stadermann and Dr. Frank Sill Torres for their support in the publication of the data.

My sincere thanks to my colleagues at the Institute for the Protection of Maritime Infrastructures of the German Aerospace Center for their incredible support, scientific contributions, and the joy they brought to our work environment. Special thanks go to Dr. Angel Bueno, Dr. Edgardo Solano, Felix Sattler and the Methods and Processing group members. I am also grateful to the Machine Learning group of MI and PI for their valuable input and discussions. And big thanks to those who read the articles and thesis and provided such helpful feedback.

I am grateful to the friends I made in Bremen along these years for their encouragement and support. They have always found kind words that made the journey lighter, even when it felt heavier.

Me gustaría agradecer a mis amigos de España por su constante apoyo.

Muchas gracias a Paloma, por haber sido mi apoyo durante años, por enseñarme tanto y por inspirarme querer alcanzar este objetivo.

Y por último a mi familia, por su amor, por hacerme quien soy y empujarme a luchar por superarme.

Contents

List of Figures

List of Tables

Acronyms

AI Artificial Intelligence
AIS Automatic Identification System
ARM Advanced Reduced instruction set computer Machine

CBAM Convolutional Block Attention Module
CNN Convolutional Neural Network
CSP Cross Stage Partial
CUDA Compute Unified Device Architecture

DLR German Aerospace Center
DLT Direct Linear Transformation
DTCWT Dual-Tree Complex Wavelet Transform

GAN Generative Adversarial Network
GDE Georeferencing Distance Error
GIS Geographic Information System
GPU Graphics Processing Unit

H Homography

IMO International Maritime Organization
IoU Intersection over Union

LS Least Squares

mAP mean Average Precision
MLP Multi-layer Perceptron

NLP Natural Language Processing
NMS Non-Maximum Suppression

ONNX Open Neural Network Exchange

RANSAC Random Sample Consensus
ReLU Rectified Linear Unit

SAHI Slicing Aided Hyper Inference
SiLU Sigmoid-Weighted Linear Unit
SMD Singapore Maritime Dataset

UTM Universal Transverse Mercator

VTS Vessel Traffic Services

Chapter 1

Introduction

Maritime infrastructures are an essential component towards the support of societal needs, economic activities, mobility, and the advancement of renewable energy sources [1]. This highlights the reason why their security, integrity, and operational safety are crucial. In response, maritime research is aiming at developing, testing, and validating systems to thoroughly assess and operate these infrastructures [2]. Such initiatives aim to cultivate a proactive and informed understanding of maritime contexts, essential for accurately determining the protection status of infrastructures in real-time and enabling prompt action against various threats, including major accidents, natural disasters, and organized crime [1]. Maritime situational awareness, facilitated by advanced technologies and data integration, is critical for a proactive and informed understanding of maritime environments [3]. It encompasses real-time monitoring and drives innovative solutions to enhance the security, safety, structural integrity, and operational reliability of infrastructures against various threats [4].

In the improvement of maritime situational awareness the introduction of advanced instruments and sensors plays a key role, which should be designed not only to recognize elements of interest but also to suggest practical measures to both users, for operational decisions, and authorities, for regulatory compliance and emergency response [5]. Enhancing maritime situational awareness with technology represents a significant advancement for smart ports, exemplifying the potential for improved maritime operations [6].

The International Maritime Organization (IMO) mandates that vessels exceeding 300 gross tonnage are equipped with Automatic Identification System (AIS) transceivers, which broadcast crucial data including identification numbers, type, position, course, and speed through encoded radio messages [7]. This system is pivotal for Vessel Traffic

Services (VTS) and nearby vessels, facilitating marine traffic awareness, critical operations such as collision avoidance, and search and rescue missions. AIS transmissions occur at intervals ranging from 2 to 10 seconds when ships are underway, which can be extended by up to 6 minutes when stationary[1]. Such intervals may leave gaps in real-time monitoring, highlighting the need for systems capable of analyzing situations with a significantly shorter interval to aid in the prevention and response to potential maritime complications [8]. Furthermore, the open standards employed by the AIS exposes it to various cyber threats, including spoofing, hijacking attacks, and denial of service, underscoring the vulnerability of the system [9–13]. Therefore, despite significant efforts, real-time ship monitoring for improved maritime situational awareness only using AIS continues to pose a challenge for VTS [14].

Other available sources for the improvement of maritime situational awareness are satellite imagery and radar systems [15]. However, their processing for real-time maritime situational awareness faces challenges due to the time-sensitive nature of data acquisition, periodic satellite overpasses, and processing delays ($\sim$15 minutes per data cycle), impacting the immediacy and utility of the information [16]. Moreover, revisit times of satellites can range from hours to days.

Optical camera systems, on the other hand, due to their accessibility, cost-efficiency, and ease of deployment, are key in rapidly assessing ship traffic, enhancing maritime situational awareness through views of the infrastructure from strategic positions [17]. The vast number of video streams available can present a challenge for operators [18]. The efficiency of real-time recognition is significantly boosted by image processing technologies applied to optical monitoring [17]. This motivates the use of computer vision and deep learning to automatically recognize and locate geographically (georeference) ships on a map, irrespective of their type or size. This process can support the operational decision-making procedures of maritime authorities by providing them with spatial information in a timely manner [19].

Early detection of potential threats and the prevention of accidents are significantly enhanced by employing optical cameras when used at full-resolution, ensuring detailed and precise imagery for monitoring purposes [20, 21]. Therefore, a key challenge in maritime monitoring is the recognition of small and distant ships, which is crucial for safety and security at maritime infrastructures, as it helps in early threat detection and accident prevention [20].

Object georeferencing involves linking physical objects to specific locations on the Earth's surface for spatial integration and analysis, a process that can extend to ob-

[1]https://www.navcen.uscg.gov/ais-messages

Figure 1.1: Conceptual representation of ship detection, segmentation and georeferencing from maritime footage. (a) Tanker being detected. Point 1 represents the bounding box center, and 2, the bottom-center point. (b) Tanker being segmented. Point 3 represents the intersection of the navigation antenna with the water. (c) Representation of the goereferenced tanker displayed on a map. The georeference from the mask, 3, provides the most accurate ship location of the three points.

jects captured within an image [22]. For the improvement of maritime situational awareness, once ships are recognized in real-time from monitoring images, the display of their geographical positions on maps using georeferencing enables a better spatial understanding of the situation [19].

Ship detection methods which use monitoring footage to present a bounding box surrounding the detected ship, can be used to improve maritime situational awareness [23, 24]. However, ship segmentation provides a more accurate georeference for the ships using the segmented masks, as shown in Figure 1.1. The georeferenced pixel can be better inferred from the segmented mask of an object than from a surrounding bounding box, which usually contains unnecessary background. The center and bottom-center of the bounding box, given the perspective of the image, provide a more erroneous georeference compared to the point that lies at the intersection point between the ship hull and the water below the bridge or wheelhouse, where the naviga-

tion antenna is located. This rationale motivates the exploration of ship segmentation methods beyond bounding box detection, and paves the way for the development of a method to automatically identify the pixel for georeferencing within this thesis.

Furthermore, the processing using embedded devices powered with a Graphics Processing Unit (GPU), equipped with monitoring cameras and placed at maritime infrastructures, represents a step forward in maritime situational awareness [25, 26]. These systems can enable on-site deep-learning-based ship recognition, offering significant advantages such as reduced network bandwidth and energy usage, minimized latency, and enhanced security [27]. The local processing of images using embedded systems directly at the infrastructure facilitates the spatial understanding of the maritime situation [28]. Recognized and georeferenced ships using an embedded system can then be seamlessly integrated into web services, allowing their display (e.g. on maps) within the situational awareness system [19]. This enhances real-time visualization and enriches the overall situational awareness by providing operators with accurate and timely spatial information [19].

Real-time and accurate ship recognition, classification, and georeferencing are essential, not just for improved spatial visualization. Beyond visualization, its combination with other data sources, such as AIS, satellite imagery and radar systems can further enhance the overall situational understanding [19]. Therefore, the faster the image processing occurs, the better it supports the creation of a comprehensive real-time situational picture by fusing with additional maritime data, thereby elevating the operational effectiveness of maritime situational awareness efforts [19].

We have seen, that in the context of enhancing maritime situational awareness, optimized real-time processing is paramount. The objective, therefore should be to ensure that the developed ship recognition and georeferencing system operates with the highest possible accuracy and the shortest inference times on embedded systems. This dual focus on speed and accuracy is critical for facilitating the fusion of the developed methodologies with other sensor data and services, thereby enabling safer, more secure, and more efficient maritime operations.

This thesis presents a compilation of explorations, methods and results proposed in the publications shown in Chapter 10, which will be referenced throughout the manuscript from [BCP-I] to [BCP-VI]. The goals and contributions of this thesis, achieved within these publications, are summarized as follows:

- Production of a real-world maritime dataset for ship recognition and georeferencing, advancing the research field of maritime situational awareness.

Contribution: Creation and publication of the ShipSG dataset, which provides a comprehensive set of annotated images for ship recognition and georeferencing [BCP-II].

- Investigation and development of a ship recognition architecture that seeks for enhanced real-time ship recognition, able to run in real-time embedded systems.

  Contribution: In-depth study of state-of-the-art methods for ship recognition, using ShipSG and other datasets [BCP-I][BCP-II][BCP-III][BCP-IV].

  Contribution: Introduced ScatYOLOv8+CBAM, an innovative ship recognition architecture optimized for real-time processing on embedded systems [BCP-V].

- Proposal of an efficient solution for processing full-resolution images on embedded systems, allowing the recognition of small and distant ships.

  Contribution: Introduced an improved slicing method that enables the processing of full-resolution images for the recognition of small and distant ships on embedded systems [BCP-VI].

- Innovation in the field of ship georeferencing using monocular images by developing a methodology that does not rely on prior camera knowledge.

  Contribution: Developed a novel ship georeferencing methodology using homographies that operates without requiring prior camera calibration [BCP-I][BCP-II][BCP-V].

- Optimization of real-time ship recognition and georeferencing methodologies for their deployment on embedded systems, balancing performance with computational efficiency.

  Contribution: Further improvement of the ScatYOLOv8+CBAM architecture for efficient deployment on embedded systems, balancing computational efficiency with high performance [BCP-VI].

- Demonstration of the practical application of the methodologies by integrating them into systems for improved maritime situational awareness in a variety of applications.

  Contribution: Successfully integrated the developed methodologies into applications such as ship georeferencing displays including map-based visualization, abnormal ship behavior detection, camera integrity assessment, and 3D ship reconstruction, showcasing their effectiveness in enhancing maritime situational awareness [BCP-I][BCP-II][BCP-III][BCP-IV][BCP-V].

The remaining chapters of this thesis are organized as follows:

Chapter 2 Fundamentals of Modern Object Recognition

This chapter dives into the technical background of modern object recognition, introducing key concepts. It explores how deep learning has transformed computer vision for object recognition and how it can be leveraged.

Chapter 3 Relevant State of the Art

This chapter focuses on relevant state-of-the-art, highlighting areas in ship recognition and georeferencing where current research falls short. The chapter presents maritime datasets and object recognition methods essential for this thesis as well as potential improvements, prior ship georeferencing research and deployment on embedded systems.

Chapter 4 ShipSG: Ship Segmentation and Georeferencing Dataset

This chapter presents the creation of ShipSG[2], a novel dataset for ship recognition and georeferencing. ShipSG provides the foundation for this thesis, enabling the development and evaluation of the methods presented in the subsequent chapters.

Chapter 5 Ship Recognition for Improved Maritime Awareness

This chapter shows the initial exploration of deep-learning-based methods for ship detection and segmentation, revealing their potential applications, such as ship georeferencing, abnormal ship behavior detection, camera integrity assessment, and 3D ship reconstruction. The study of state-of-the-art instance segmentation methods sets the stage for the custom developments and analysis proposed in subsequent chapters.

Chapter 6 Advanced Ship Recognition for Real-time Operation

This chapter addresses the need for fast and accurate algorithms on embedded systems for real-world use. While ship detection was proven to perform well, deploying instance segmentation (better for ship georeferencing) on embedded systems was shown to be a significant challenge. This chapter addresses this gap by proposing a customized real-time segmentation method (ScatYOLOv8+CBAM), deployed on an embedded system. It also proposes a method to improve the segmentation accuracy for small and distant ships by processing full-resolution images, crucial for better maritime situational awareness.

[2]https://dlr.de/mi/shipsg

Chapter 7 Ship Georeferencing for Maritime Situational Awareness

This chapter focuses on the georeferencing of the ships recognized using monocular cameras to improve maritime situational awareness. This involves the development of a method to present ships on a global scale using only single images without prior camera knowledge. The chapter first explains homographies and then details the proposed method for georeferencing ship bounding boxes, along with the calculation of ship heading direction from optical flow. Finally, the chapter quantitatively analyses how this monocular ship georeferencing improves maritime situational awareness.

Chapter 8 Summary and Conclusion

This chapter summarizes the contributions and key findings of this thesis, and concludes the outcome of the produced results.

Chapter 9 Future Work

This chapter presents the challenges encountered throughout the thesis and proposes new research lines to approach future work.

Chapter 10 Publications by the Author for this Thesis

This chapter presents the list of publications used in this compilation thesis and includes a short summary of my contributions to each publication.

Chapter 2

Fundamentals of Modern Object Recognition

As motivated in Chapter 1, the combination of computer vision and deep learning offers a potent solution for automatic ship recognition using optical monitoring cameras. This chapter provides a technical overview of concepts to understand modern object recognition, starting with the role of supervised learning in computer vision, and followed by the use of deep learning for the two computer vision tasks of interest in this thesis, object detection and instance segmentation.

2.1 Supervised Learning in Computer Vision

Computer vision is a discipline within Artificial Intelligence (AI) that allows machines to process and interpret visual data. By harnessing algorithms and data, computer vision systems can identify and classify objects, and make decisions based on visual inputs similar to the way humans do [29]. The field of computer vision has significantly advanced with deep learning, a subfield of machine learning, particularly through the use of Convolutional Neural Networks (CNNs) [29]. Preceding computer vision approaches relied on hand-engineered feature extraction. Deep learning, on the other hand, utilizes vast amounts of visual data to train hierarchical structures of neurons that excel at identifying patterns to therefore perform automatic feature extraction [30]. Thanks to the use of GPUs, deep learning with CNNs have significantly surpassed the performance of traditional algorithms in tasks like image classification, object detection, and instance and semantic segmentation [31]. The computational power of GPUs, due to the parallel processing capabilities, enabled the training of

deep networks with millions of parameters, allowing for the extraction of complex features from large datasets.

Machine learning algorithms are commonly categorized into supervised learning, which requires labeled data for training, unsupervised learning, which operates on unlabeled data to find patterns, and semi-supervised learning, which uses a combination of labeled and unlabeled data to train models [32]. In certain real-world applications, supervised learning is preferred for training models with real-world annotated datasets, ensuring accurate identification and categorization of objects represented in the data [33].

In supervised learning, models are trained on datasets labeled by human experts [34]. During training, the supervised model adjusts its parameters by measuring the deviation from the actual labels [32]. Therefore, the annotation process involves pairing each training sample with its corresponding output labels, serving as a learning guide for the model. In computer vision tasks, such as image classification, labeled training images are used to predict classes on validation images [35]. In object detection tasks, the annotations and training extends for the classification and localization of objects in the image within bounding boxes. Segmentation tasks demand detailed annotations, labeling each pixel by class [36].

In summary, supervised learning has greatly advanced computer vision tasks, while also highlighting the continuous need for models that can learn effectively from annotated data.

2.2 Deep-Learning-Based Object Recognition

Object recognition in computer vision involves identifying and classifying objects in images [29]. Two main tasks in the field are object detection and instance segmentation, which are essential for machine interpretation of visual data and widely used in autonomous driving, monitoring, surveillance and medical imaging applications, among others [33, 36]. Figure 2.1 depicts the difference between object detection and instance segmentation.

Object Detection aims to locate and classify objects within an image, including the determination of their presence and exact location within bounding boxes.

Instance Segmentation advances beyond detection with bounding box by identifying each object instance in an image at the pixel level, delineating its shape with a mask. Unlike semantic segmentation, which classifies each pixel within the image

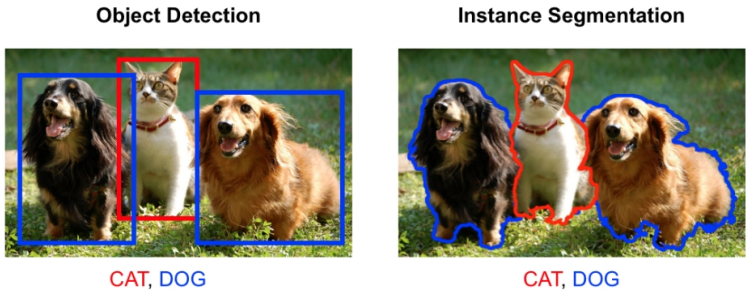

Figure 2.1: Example of object detection and instance segmentation on an image. Object detection involves bounding box localization and classification, whereas instance segmentation goes beyond that to provide a mask outlining the exact shape of each individual object instance. Adapted from [37].

as belonging to a certain class, instance segmentation recognizes each object instance separately and the rest is considered background.

2.2.1 Standard Architecture Description

Modern deep learning architectures for detection and segmentation tasks extensively use CNNs, featuring a combination of a backbone, a neck and head structure [29].

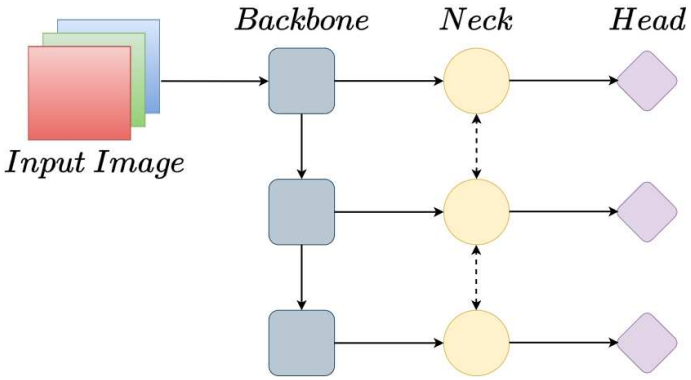

Figure 2.2: Standard deep learning object recognition architecture. See text for details.

Figure 2.2 illustrates a standard deep learning object recognition architecture. In CNN-like architectures, a feature map is the output of one filter (also known as kernel) applied across the previous layer to detect specific features [29]. In the case of an object recognition architecture, the backbone focuses on extracting features by learning to recognize task-relevant patterns in visual data, performing changes in feature map resolution (width, height and channel number). The arrows in Figure 2.2 represent the flow of data through the network layers. As data progresses through the backbone,

the resolution typically decreases to reduce the spatial dimensions while increasing the depth (number of channels) to create more abstract and complex feature representations [29]. Conversely, in the neck, the resolution can either decrease or increase. Typically, in object detection and instance segmentation tasks, the neck often includes upsampling to increase spatial information, allowing a more accurate location of objects of interest [29]. The neck fuses and aggregates features from different resolutions, acting as a bridge between the backbone and the head. The head performs specific tasks based on these features, such as detection, segmentation, and classification. Although they are different tasks, detection and segmentation share similarities in terms their architecture, with each head designed to perform the desired task. However, the design of backbone and neck can be tailored to perform better for the task of interest.

As illustrated by Figure 2.2, deep learning architectures for object recognition that use CNNs typically comprise blocks that represent combinations of structured layers to process the visual data. These blocks include include a combination of convolutional, pooling, upsampling, activation and regularization layers [38].

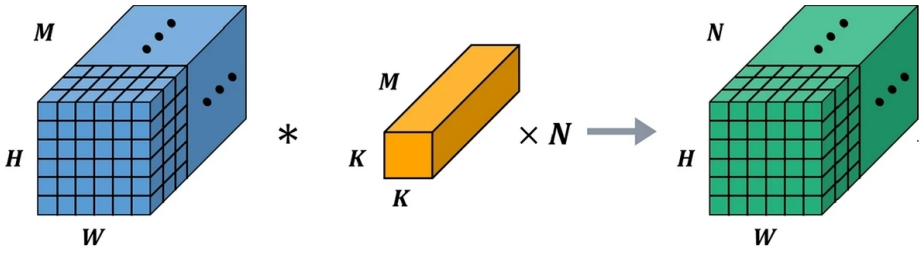

Figure 2.3: Illustration of a standard convolution operation, taken from [39]. The input volume has dimensions $H \times W \times M$ (height, width, and number of channels). A filter, also named kernel, of size $K \times K \times M$ is convolved with the input, producing an output volume of dimensions $H \times W \times N$, where N is the number of filters. This process involves sliding the filter over the input and computing the dot products between the filter weights and local regions of the input.

Convolutional layers apply filters (kernels) to the input to create feature maps. These filters contain learned weights, which are adjusted during training to optimize feature extraction (see Figure 2.3). The convolution operation involves sliding the filter over the input to compute dot products between the filter weights and local regions of the input, generating feature maps that capture different aspects of the input data. Though not depicted in Figure 2.3, to further adjust the output, a learnable bias term is normally also added to each output element.

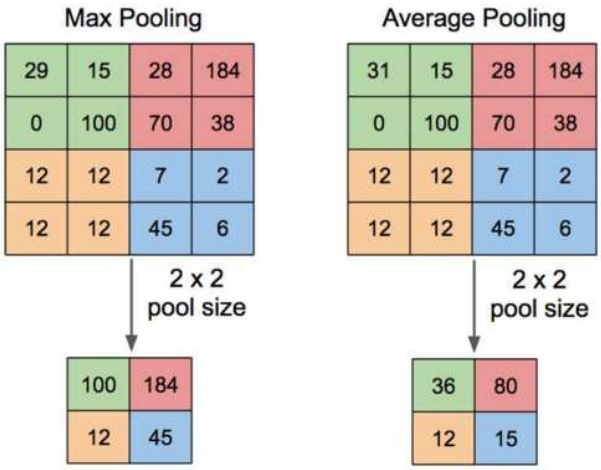

Figure 2.4: Illustration of max pooling and average pooling operations with a 2×2 pool size, taken from [39]. Max pooling selects the maximum value from each 2×2 block, while average pooling computes the average value from each 2×2 block, reducing the spatial dimensions of the input feature maps.

Pooling layers perform a fixed operation to reduce the spatial dimensions of the feature maps, down-sampling the input to reduce computational load and enhance invariance to small translations. As seen in Figure 2.4, there are two common types of pooling: max pooling and average pooling. Both types of pooling effectively reduce the spatial dimensions while preserving important spatial features. Moreover, pooling layers reduce the spatial resolution of feature maps to combat overfitting, which happens when a model memorizes training data, failing to generalize to new, unseen data.

In contrast to pooling, upsampling layers perform the opposite operation by increasing the spatial dimensions of the feature maps. Upsampling can be achieved through various methods, such as nearest-neighbor interpolation, bilinear interpolation, or transposed convolutions [29]. Upsampling layers are used to increase resolution when finer detail is necessary.

Activation layers introduce non-linearity, enabling the network to capture complex patterns [38]. Typical activation functions include Rectified Linear Unit (ReLU) and Sigmoid-Weighted Linear Unit (SiLU). The ReLU, given by $f(x) = \max(0, x)$, function provides a linear output that is zero for negative inputs and linear with a slope of 1 for positive inputs. SiLU, given by $f(x) = x \cdot \sigma(x)$, incorporates a sigmoid function, described as

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

allowing it to handle negative inputs more dynamically by scaling the output in a non-linear fashion.

Regularization layers, such as dropout and batch normalization, are also integrated into CNN architectures. Dropout randomly omits neurons during training to enhance generalization, while batch normalization scales the output of a layer to have a mean of zero and a variance of one, which can expedite training and improve overall performance [38].

State-of-the-art deep learning architectures, such as those that will be described in Chapter 3, often combine the above-presented layers in innovative ways to enhance model performance and efficiency. For instance, methods like ResNet, normally used as backbone in object recognition architectures, introduce skip connections that allow gradients to flow more easily through very deep networks, facilitating training [40]. Other advancements involve combining the layers in specific configurations to achieve desired properties. For example, Feature Pyramid Networks (FPNs) employ a combination of convolutional and upsampling layers to create feature maps at different scales, allowing the model to better handle objects of varying sizes within an image [41]. Additionally, various forms of attention mechanisms (see Section 2.2.2) can be integrated using these building blocks to selectively focus on relevant parts of the feature maps, leading to improved performance [42]. In summary, the combination of layers are foundational and their interactions is crucial for the advancements in contemporary deep learning models.

2.2.2 Attention Mechanisms

Neural networks use attention mechanisms to allow models to dynamically focus on the most relevant parts of the input data, enhancing their ability to process complex information [42, 43]. Initially introduced in Natural Language Processing (NLP) for tasks like language translation [43], attention mechanisms have since become integral to various deep learning applications, including computer vision. At the core of these mechanisms are attention weights, learned during training, which determine the importance of different parts of the input [44]. For example, in language translation, this results in an $n \times n$ attention matrix or map, where n is the number of words [43].

The implementation of attention mechanisms involves the calculation of the attention weights using a score function [43]. This is achieved by applying the input

to a learned weighted matrix that computes relevance scores using functions like dot product [42], or pooling operations (e.g., max pooling, average pooling) [45]. These scores create an attention map, highlighting the importance of different input parts. The network uses this map to prioritize crucial information, learning to distribute focus effectively across the input data.

In computer vision, the term attention translates to individual pixels attending to other pixels, or patches of pixels attending to other patches, leading to an attention map that captures the relationships across different regions of the image [46]. However, this poses significant computational challenges. To address these challenges, various strategies have been proposed in the literature:

- **Dimensionality reduction** with convolutional layers and pooling operations are often used to reduce the dimensions of the input before applying attention, decreasing the computational load by working with smaller feature maps [45, 47].

- **Hierarchical attention** mechanisms apply attention at different scales or hierarchies, allowing the model to first focus on broad, coarse details and progressively refine its attention to finer details, thus significantly reducing complexity [48].

- **Local attention** restricts the attention mechanism to a local neighborhood around each pixel, limiting the number of interactions and thereby reducing the computational burden [49].

- **Spatial attention** mechanisms identify important spatial locations within an image, thus allowing the model to concentrate on critical regions [50].

- **Channel attention** mechanisms focus on identifying important feature channels within a CNN, thereby improving the model's feature representation [47].

Attention mechanisms are, therefore, used to enhance the ability of models to focus on key parts of an image. Specifically for computer vision tasks, they have been incorporated to CNNs to improve performance in large-scale image classification tasks [47, 51]. Additionally, they have been applied in object detection and instance segmentation tasks by incorporating spatial and channel-wise attention, which improves feature representation and accuracy [45, 47, 52, 53].

2.2.3 Object Classification and Postprocessing

To accurately classify objects, after they are detected or segmented, the final layer of the head, usually a fully connected layer [30], provides a confidence score for each potential prediction, which reflects the likelihood that a recognized object belongs to a specific class. Multiple prediction proposals (bounding boxes or masks) for the same object require further postprocessing beyond the head to enhance the accuracy and reliability of a prediction. Standard object recognition architectures include postprocessing after the head to refine the bounding boxes or segmentation masks recognized by the model and to eliminate redundant predictions [29]. A key component of this postprocessing phase is Non-Maximum Suppression (NMS) [54], a technique designed to eliminate redundant bounding boxes or segmentation masks that pertain to the same object. Essentially, NMS ensures that each detected object is represented exclusively by the single, most accurate bounding box or mask, thereby preventing clutter and providing a clearer output.

To decide which bounding boxes or masks to keep and which to discard, NMS relies on the confidence scores of the predictions, together with a metric known as Intersection over Union (IoU). This metric measures the overlap between two areas—in this case, the area of overlap between a predicted bounding box or mask and the ground truth, as shown in Figure 2.5. The IoU helps in determining the accuracy of the predictions by quantifying how closely the predicted bounding boxes or masks align with the actual objects in the image.

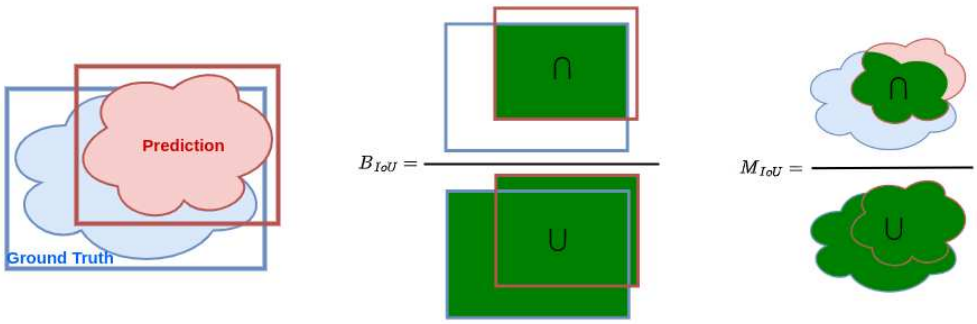

Figure 2.5: Joint illustration of Intersection over Union calculation for boxes and masks. On the left, the ground truth, and predicted bounding boxes and masks are shown. In the middle, the IoU for the bounding box is visualized. On the right, in the case of instance segmentation, the IoU for the mask is calculated.

Mathematically, B_{IoU} (bounding box) and M_{IoU} (mask) can be denoted as:

$$B_{IoU} = \frac{\text{area}(B_{pred} \cap B_{gt})}{\text{area}(B_{pred} \cup B_{gt})} \quad (2.2) \qquad M_{IoU} = \frac{\text{area}(M_{pred} \cap M_{gt})}{\text{area}(M_{pred} \cup M_{gt})} \quad (2.3)$$

This ratio ranges from 0 to 1, where 0 indicates no overlap and 1 indicates perfect overlap. In practice, an IoU threshold is set (e.g., 0.5 or 50%) to classify predictions as true positives or false positives. The NMS filters the best final prediction from the possible proposals, represented as $P_{final} = NMS(P, S, \tau)$, for a set of predictions P (either masks or bounding boxes) with associated confidence scores S and an IoU threshold τ.

Following the discussion of object recognition architectures and postprocessing, it becomes relevant to address the practical aspects of implementing these frameworks. PyTorch [55] is a popular deep learning library for computer vision, valued for its dynamic computation graph and efficient GPU memory management. Its straightforward syntax simplifies the implementation of supervised CNNs, making it ideal for research and development. This thesis leverages PyTorch to develop models for ship recognition.

2.2.4 Training Process

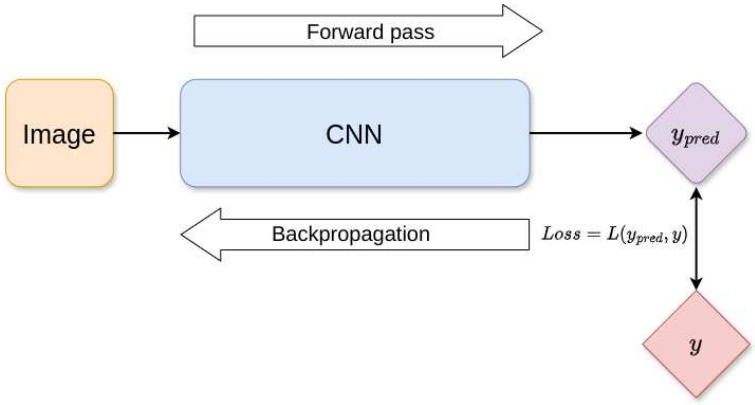

Figure 2.6: Schematic of the Convolutional Neural Network (CNN) Training Process. An input image is passed through the CNN during the forward pass, resulting in a predicted output (y_{pred}). The prediction is compared to the true value (y) using a loss function ($L(y_{pred}, y)$), and the error is propagated back through the network during backpropagation to adjust and improve the model weights.

The training process of CNNs for object detection and segmentation includes forward and backward propagation, as illustrated by Figure 2.6.

During forward propagation, the input data is fed through the network to output a prediction. The choice of loss function is crucial for model performance. Commonly used loss functions for different tasks include Binary Cross-Entropy Loss, Focal Loss, Bounding Box Loss, Objectness Loss, and Pixel-wise Cross-Entropy. Each of these loss functions addresses specific aspects of the prediction problem, such as class imbalance (Focal Loss) or spatial localization accuracy (Bounding Box Loss). These loss functions, sometimes used in combination or with other loss functions, help the model learn from its mistakes and achieve optimal performance [31]. Further description and mathematical definitions of these loss functions can be found in reference [29].

Backward propagation, based on a loss function, then adjusts the network weights to minimize discrepancies between the predictions and the ground truth [38]. This adjustment process involves calculating gradients of the loss function with respect to the network parameters. These gradients indicate the direction and magnitude of the changes needed to reduce the loss.

Optimization algorithms use these gradients to update the network parameters iteratively. Common optimization algorithms include Stochastic Gradient Descent (SGD), Adam, RMSprop, and AdaGrad [30]. Each algorithm has its strengths and is chosen based on the specific requirements of the task. The goal is to minimize the loss function, thereby improving the performance of the model. The learning rate, a key factor in this process, determines the size of the updates. Optimization involves multiple passes through the dataset, known as epochs, where the network parameters are refined to achieve better accuracy and generalization [30].

2.2.5 Evaluation Metrics

Evaluating the performance of object recognition models is critical to understanding their effectiveness and accuracy. The mean Average Precision (mAP) is a commonly-used metric to evaluate object detection and segmentation performance [56]. Expressed in percentage, it is calculated as the mean of all the Average Precisions (AP) for all classes present in the dataset at a given IoU threshold. This is, mathematically:

$$mAP_\tau = \frac{1}{C} \sum_{c=1}^{C} AP_{\tau,c} \tag{2.4}$$

Here, mAP_τ represents the mAP at an IoU threshold τ, calculated by averaging the AP values across all C classes. For the calculation of AP for each class, true positives are counted when the IoU of the prediction exceeds the given threshold τ. In the case of object detection, a true positive is confirmed when the IoU of the predicted

bounding box exceeds the IoU threshold. For instance segmentation, true positives are based on the overlap between the predicted mask and the ground truth mask at the IoU threshold. This distinction in true positive calculation signifies the different evaluation approaches between object detection and instance segmentation.

It is common in the field of object recognition to refer to mAP as a short form of $mAP_{0.5:0.95}$ [29]. The $mAP_{0.5:0.95}$ accounts for mAP values at IoU thresholds that range from 0.5 to 0.95, in increments of 0.05. The formula would therefore be defined as:

$$mAP = \frac{1}{N} \sum_{\tau=0.5}^{0.95} mAP_{\tau} \tag{2.5}$$

Where N represents the number of thresholds, which is 10 in the case of the range $0.5 : 0.95$. This comprehensive evaluation across several IoU thresholds provides insights into the performance of the model at different levels of strictness in object localization against the ground truth.

Additionally, the mAP can accomodate objects of varying sizes by further categorizing it based on the pixel area of the detected objects [56]:

- mAP_s (small) if $area \leq 32^2$ pixels
- mAP_m (medium) if $32^2 < area \leq 96^2$ pixels
- mAP_l (large) if $area > 96^2$ pixels

This distinction per object size allows for a more detailed analysis of performance, especially in datasets with a wide range of object sizes, by highlighting its ability to detect small, medium, and large objects.

In order to compare results with existing standards, datasets such as COCO [56], with over 330,000 images and detailed annotations, are used as a resource for training and evaluating computer vision models in object detection and instance segmentation. Its diverse image collection makes it a valuable resource for researchers and developers. In the literature of experimental general purpose object recognition, it is a standard practice to evaluate general-purpose object recognition models performance using COCO as a benchmark [29] with the mAP as metric.

Chapter 3

Relevant State of the Art

Expanding on the foundations of Chapter 2, this chapter identifies key limitations in approaches to maritime situational awareness prior to this thesis. Specifically, this chapter reviews existing maritime datasets, real-time ship recognition algorithms, georeferencing techniques and gives an overview of the technologies available for deployment on embedded systems. By analyzing limitations, this chapter lays the groundwork for the studies and developments of a novel approach presented in later chapters.

3.1 Real-world Maritime Datasets

The accuracy of a supervised learning model is greatly dependent on the quality and volume of the annotated data it is trained on, especially for real-world applications [34]. As deep-learning-based ship detection and segmentation rely on supervised learning, it is necessary to use domain-specific training datasets [33]. The training set and annotations must accurately represent the variety of ways objects can appear in different conditions [57].

Real-world maritime monitoring requires image data with precise annotations for a broad range of ships and ship classes [58]. General-purpose detection and segmentation datasets, such as COCO [56] or PASCAL VOC [59], therefore, do not suit the task of ship recognition and georeferencing as benchmark datasets for maritime awareness. Relevant datasets in the literature for ship detection on video monitoring cameras are the Singapore Maritime Dataset (SMD) [17], Seaships7000 [60], and a dataset introduced by Chen et al. [61]. Moreover, other private datasets exist [62, 63], however the restricted access makes the experimental validation using them not possible. The accessible datasets, lack a variety of ship classes in their annotations and do not provide ship masks, necessary for ship georeferencing. The MarSyn dataset [64] is a synthetic

ship dataset that contains images rendered from synthetic 3D scenes for instance segmentation in six ship classes, without georeference from the ships annotated.

A literature review on ship detection and localization [65] highlights the fact that while annotations for ship datasets should include more complex data such as latitude and longitude of the ship, available datasets primarily focus on the object classes and bounding boxes, without masks or geographic positions. However, as motivated in Chapter 1, ship segmentation provides a more suitable solution for georeferencing. Therefore, we find the need for a publicly available real-world dataset, for ship segmentation and georeferencing, that includes footage of a maritime infrastructure as well as mask and georeferencing annotations of several classes of ships. This dataset, should aim towards the advancement and evaluation of ship recognition methods for the improvement of maritime situational awareness.

3.2 Ship Recognition Using Maritime Monitoring Footage

To enhance maritime situational awareness, it is crucial to use methods that perfom ship recognition on maritime footage [17]. However, these methods should not only recognize ships but also allow the gathering of essential information about them, such as their class and geographic location (georeference). It is vital to present this information in a simplified format to maritime operators for quick and effective decision-making [19]. Additionally, deploying an embedded system, with a monitoring camera on board, enables deep-learning object recognition directly on-site. This approach reduces network bandwidth, minimizes latency, improves security, and offers cost-efficiency [27], but comes with the trade-off of lower computational power compared to high-end systems [25]. To address these complexities effectively, it is important that ship recognition methodologies not only ensure high accuracy across various ship sizes and types but are also optimized for the constraints of embedded hardware. Furthermore, the inference speed of video-based ship segmentation is paramount, as it significantly contributes to the improvement of data fusion with other sensor data, leading towards more cohesive maritime situational awareness systems [19].

These challenges underline the need to search for effective object recognition methods in the literature. The following list provides a brief overview of state-of-the-art object detection and instance segmentation methods that are particularly relevant:

- **YOLOv4-CSP [67]**. The You-Only-Look-Once (YOLO) algorithm was introduced for real-time object detection [68]. It divides the image into a grid and predicts bounding boxes and class probabilities for each grid cell, based on pre-

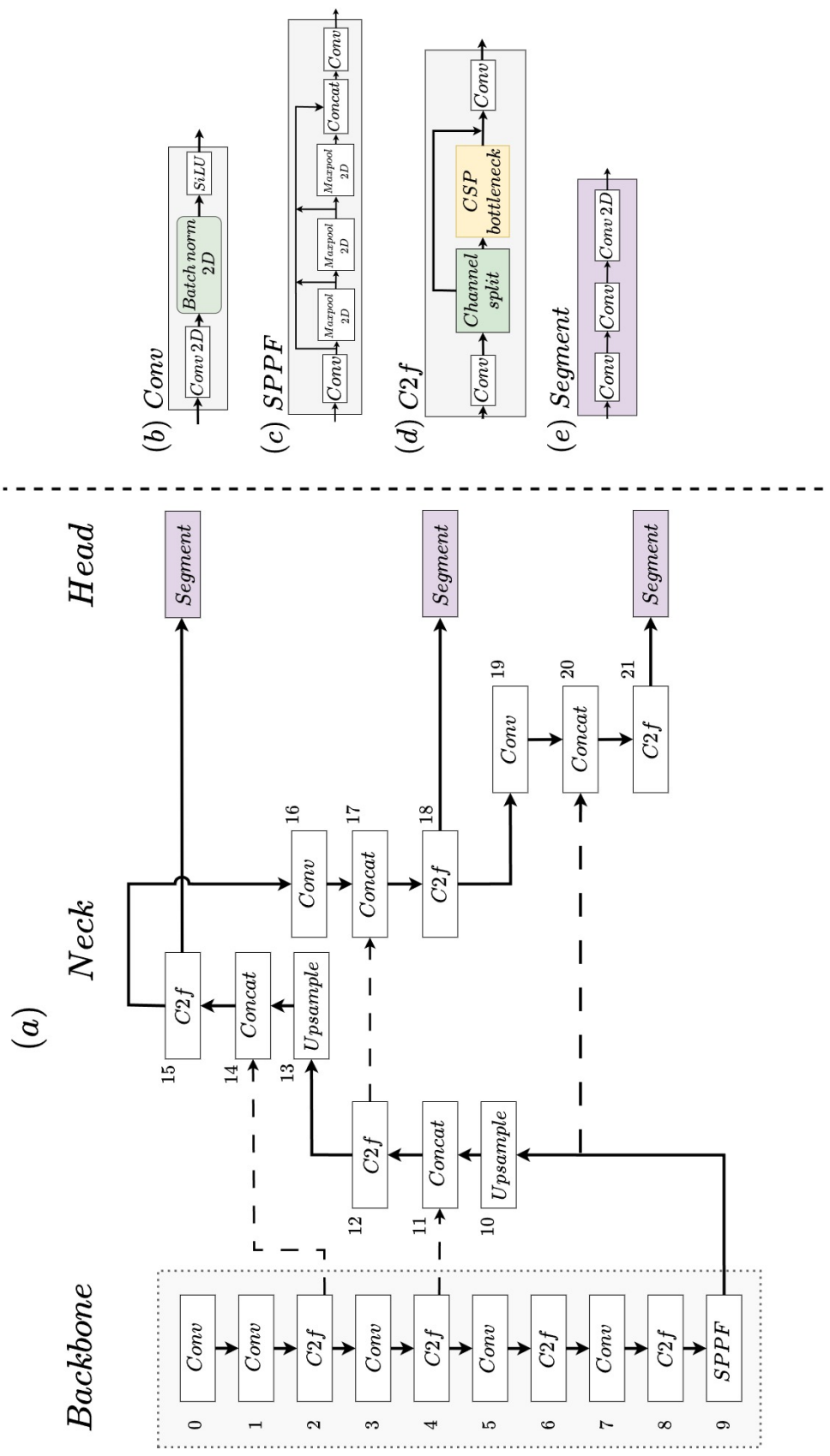

Figure 3.1: The YOLOv8 architecture by Ultralytics [66]. (a) Backbone, neck and head of the architecture. The numbers next to every block represent the sequential order followed by the implementation, from input to output. (b) Standard YOLOv8 convolutional block. (c) Spatial Pyramid Pooling Fast module of YOLOv8. (d) C2f with split channel operation and a Cross Stage Partial (CSP) Bottleneck. (e) Segment block of YOLOv8 that performs segmentation. Modified from [BCP-V] ©2023 IEEE.

defined anchors that act as reference points. These anchors are calculated by applying k-means clustering [32], and serve a set of provisional bounding boxes for the object detection task. YOLOv4-CSP presented a substantial enhancement in speed and accuracy by leveraging two main innovations. The first innovation is the use of CSPDarknet53 [69] as backbone, based on CSP network [70]. The CSP network partitions the feature map into two parts: one part is processed through a series of layers while the other part bypasses these layers, ensuring better gradient flow and capturing patterns more effectively [70]. The second innovation is the Bag of Freebies technique, which includes data augmentation, label smoothing, and additional regularization methods. These improvements reached a detection mAP of 47.5% on COCO.

- **Faster R-CNN [71] and Mask R-CNN [72]**. Mask R-CNN is a two-stage instance segmentation method that was developed as an extension of the object detector Faster R-CNN. They use the Region Proposal Network introduced in [71] to identify object candidates and then refine these detections by classifying them and fitting precise bounding boxes. In Mask R-CNN, a fully convolutional network was added to regress the mask from the detected bounding boxes, reaching a mask mAP of 39.8% on the COCO dataset with the ResNeXt-101 backbone [73].

- **DetectoRS [74]** is a multi-stage instance segmentation method that enhances the use of recursive feature pyramid network [41] and feedback connections [75] for improved performance. It features an atrous convolution, a type of dilated convolution used to expand the receptive field, allowing it to capture larger areas of the input [76]. DetectoRS uses ResNet-50 [40] as its backbone, and achieves a 44.4% mask mAP on the COCO dataset.

- **YOLACT [77]** emerged as one of the first real-time instance segmentation approach, operating in one-stage. It generates prototype masks through an independent fully convolutional network [78] and computes coefficients for adjusting these masks to the predicted bounding boxes. After suppressing overlapping detections with Non-Maximum Suppression (NMS), it filters the masks using anchor boxes. With a ResNet-101 backbone [40], YOLACT attains a mask mAP of 34.1% on the COCO dataset.

- **Centermask-Lite [79]** is a one-stage instance segmentation method, optimized for real-time applications. It utilizes a spatial attention-guided mask branch

within a fully convolutional object detector [80] to refine proposed regions. It incorporates a novel backbone, VoVNet [81], which enhances feature map integration and, with VoVNet-39, achieves a 36.3% mask mAP on the COCO dataset.

- **YOLOv5 [82]**, developed within Ultralytics framework[1], builds on YOLOv4 but utilizes PyTorch and introduces the AutoAnchor algorithm to automatically fine-tune anchor boxes over multiple iterations. It achieves a detection mAP of 50.7% on COCO.

- **YOLOv8 [66]**, also developed by Ultralytics, builds upon previous YOLOv5. With a focus on real-time applications, this version supports a full range of vision tasks, including detection and instance segmentation. In this thesis, YOLOv8 plays a central role in the customized instance segmentation architecture proposed in [BCP-V] and improved in [BCP-VI], ScatYOLOv8+CBAM (Section 6.2). Therefore, YOLOv8 is described further than the previous state-of-the-art methods. The YOLOv8 architecture is divided into three main parts: Backbone, Neck, and Head, as illustrated in Figure 3.1. The model uses the backbone CSPDarknet53 [69] as previous YOLO versions, but includes the novel C2f module (Figure 3.1(a)). The blocks found in the backbone are:

  - **Conv Block (Figure 3.1(b))**: Each Conv block includes a 2D convolution, followed by batch normalization and a Sigmoid-Weighted Linear Unit (SiLU) activation function. This block reduces the spatial dimensions (width and height) and increases the number of channels.
  - **SPPF Block (Figure 3.1(c))**: The Spatial Pyramid Pooling Fast (SPPF) block performs multiple max-pooling operations at different scales, concatenates the results, and then applies a convolution. This block maintains the number of channels.
  - **C2f Module (Figure 3.1(d))**: This module contains a series of convolutional layers, a channel split operation, and a CSPbottleneck. The CSPbottleneck splits the input feature map into two parts: one part goes through a series of convolutional layers (bottleneck), while the other part bypasses these layers. The outputs are then concatenated, which helps in maintaining gradient flow and reducing computational load. This de-

[1]https://github.com/ultralytics/

sign enhances feature extraction and learning efficiency. The C2f module increases the number of channels.

The neck of YOLOv8 (Figure 3.1(a)) is designed to generate feature pyramids that help the model handle objects at different scales. Further blocks of the neck are:

- **Upsample**: This block increases the spatial dimensions (height and width) of the feature maps.
- **Concat**: The Concat blocks merge feature maps from different stages, increasing the number of channels.

The head (Figure 3.1(a)) is responsible for producing the final output of the model. It includes the following:

- **Segment Block (Figure 3.1(e))**: This block consists of a series of convolutional layers and generates segmentation masks and classifies each detected object. The postprocessing of YOLOv8, not shown in Figure 3.1, combines the output of both the pixel-level masks and class labels for each object.

YOLOv8 also offers five model sizes, these being, from the lightest and fastest to the deepest and most accurate: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. It achieves a detection mAP of 53.9% and a mask mAP of 43.4%.

The above-listed methods were originally designed by their authors using COCO [56] as benchmark dataset, serving this as a way of identifying robust and real-time models in the literature that could be suited for ship recognition tasks.

An effective object recognition method for ship recognition should offer potential for integration with additional tasks, such as georeferencing, which, is vital for enhancing maritime situational awareness [19]. This necessitates a move beyond the utilization of existing state-of-the-art object detection and segmentation models, towards developing an advanced approach. Such an approach must be lightweight enough for embedded system deployment while maintaining or enhancing precision and speed. This highlights the need for innovative, efficient solutions capable of meeting the stringent demands of both performance and practicality in maritime situational awareness.

To tackle the challenges mentioned and boost performance in real-world scenarios with scarce data, real-time ship recognition can be improved by leveraging advanced

methods like attention mechanisms and techniques from other fields, such as the 2D scattering transform. The 2D scattering transform, which uses wavelets, has been widely used in signal processing tasks such as speech recognition, time series analysis, astrophysics, and geosciences [83–86]. Prior research has utilized wavelets in the computer vision field to perform multi-resolution geometric processing, multi-scale oriented filtering and image denoising [29]. In combination with deep learning and computer vision, the scattering transform has been used for image classification tasks [87], proving to provide a deep systematic understanding of how invariant features can be captured and utilized. It captures the essence of geometric and structural properties, which are crucial for recognizing complex patterns under various conditions [88]. The integration of the 2D scattering transform and attention to an object recognition method should aim to enhance ship recognition performance, offering an efficient and targeted solution to the previously outlined challenges. The technicalities of the 2D scattering transform are given in Chapter 6.

Enhancing the performance of ship recognition systems through the integration of advanced techniques represents a significant step forward in maritime situational awareness. However, the practical application of these advancements in real-world monitoring scenarios brings to the forefront additional challenges, particularly in the recognition of small and distant ships. The effective recognition of small and distant ships ensures protection at the infrastructure by enabling early threat detection and accident prevention at maritime infrastructure [20]. Using images at their original full-resolution, or even high-resolution cameras, is essential for this task [21]. However, deep learning techniques for object recognition on high-resolution images consume significant memory and necessitate larger neural networks, which complicates their real-time deployment [89]. Moreover, high-resolution processing on embedded systems with limited memory presents additional challenges, impacting performance and latency [90]. Image super-resolution, which consists of the synthetic increase of the input image resolution, has been used in the literature [21]. In reference [91], their work introduced additional blocks and layers using transformers, which leverage self-attention extensively [92] for better small object accuracy from aerial views [91]. These solutions increase computational complexity beyond the capacity of embedded systems for real-time operation. The Slicing Aided Hyper Inference (SAHI) method, introduced in reference [93], splits high-resolution images into slices, enabling detection and segmentation of small objects. Slicing mechanisms, therefore, allow the processing of high-resolution images on embedded systems by dividing the images into manageable sections, thus reducing the computational load and memory requirements.

However, SAHI lacks native batch inference processing, and instead, processes slices sequentially. This highlights the need for a slicing method which can benefit real-time applications such as small ship segmentation, as proposed in this thesis.

3.3 Georeferencing of Recognized Ships

The field of object georeferencing from images extends across various domains, from aerial vehicle tracking using airborne cameras [94] to the vehicle geolocation in urban environments for autonomous driving [95]. These methods, while effective within their respective ranges of operation (dozens of meters), require positioning systems on board for calibration. The method proposed by [96] introduced an alternative that estimates georeferences from surveillance camera videos by aligning video frame points with geographic locations using a homography transformation to project the camera space onto orthophoto maps, which are geometrically corrected aerial photos, and Digital Elevation Models (DEMs). However, the application of high-resolution orthophotos and DEMs for maritime environments presents significant challenges, as these methods primarily model terrestrial elevations and are less effective for water surfaces, where the dynamic nature of water and its reflective properties complicate the creation of DEMs that could be used for ship georeferencing.

Table 3.1: Ship georeferencing accuracy in existing literature. Note: Some entries lack reported uncertainty values for the positioning error.

Source	System	Range to Object	Error (m)
[97]	Radar Antenna + GPS*	1000 m	6.5
[98]	Synthetic Aperture Radar	800 km	13 ± 23
[99]	Opt. Remote Sensing	36000 km	165 ± 109
[24]	Opt. Camera + GPS + IMU**	400 m	20

*Global Positioning System, **Inertial Measurement Unit

To transition from terrestrial to maritime applications, as discussed in Chapter 1, ship georeferencing is a critical aspect. This process involves the assignment of geographic coordinates to ships detected in various data sources. Table 3.1 shows a summary of ship georeferencing accuracies in the literature using different technologies. Radar technologies have been a cornerstone in this field, providing real-time georeferencing at a speed of 1 Hz, as detailed in [97]. Despite their accuracy, radar systems often involve high costs and complex deployment requirements. Parallel to radar, satellite technologies including optical remote sensing [99] and synthetic aper-

ture radar (SAR) [98], extend georeferencing capabilities over larger coverage areas. However, the effectiveness of these methods is constrained by their data cycle times (∼minutes) and the satellite revisit schedules, limiting their temporal resolution. Recent advancements have explored the use of video sequences for ship georeferencing. The work in [24] proposed a method which relies on the pinhole camera model calibration matrix [100] to georeference ships detected in video frames. This approach, while innovative, requires prior knowledge of camera calibration, and its application has been limited to controlled conditions with a single video sequence of two small ships.

The methodologies and technologies reviewed reveal a landscape where accuracy, range, and cost are in constant negotiation. While radar and satellite methods offer comprehensive coverage, their practical deployment is often hindered by high costs and technical complexities. Conversely, camera-based approaches present a cost-effective alternative but are limited by the need for prior calibration or additional sources, such as orthophotos or DEMs. A solution that uses cameras without pose calibration would facilitate scalability in the deployment of the georeferencing method to existing monitoring cameras at the maritime infrastructure.

3.4 Deployment on Embedded Systems

Utilizing a Graphics Processing Unit (GPU) on an embedded system, equipped with a monitoring camera, can allow for on-site deep-learning object recognition, streamlining the process significantly [25]. Processing images directly on the embedded system, rather than transferring them to a cloud or server, produces notable reduction in network bandwidth and latency, alongside cost savings and enhanced security [27]. This integration facilitates real-time access to recognized and georeferenced ships through web services, enabling their display on maps for operators, boosting maritime monitoring [19].

Table 3.2: Comparison of NVIDIA GPU modules, with focus on the Jetson family and high-end GPU-powered systems.

System Type	Module	CUDA Cores	Memory	Max Power Consumption
Edge Computing Device	Jetson Nano	128	4 GB	10 W
	Jetson TX2	256	8 GB	15 W
	Jetson AGX Xavier	512	16GB	30 W
High-End Device	GV100	5120	32 GB	250 W
	A100	6912	80 GB	400 W

Within the spectrum of embedded systems widely utilized for deep learning and computer vision, the NVIDIA Jetson family[2] stands out in the literature, offering both mobile and energy-efficient embedded GPU-systems [101]. Table 3.2 shows a comparison of three modules of the Jetson family compared against high-end server-based GPU systems to contextualize their capabilities. We observe that the Jetson modules provide optimized balance between performance and energy efficiency, marking them as an optimal solution for vision-based systems where the on-site deployment is a requirement. Larger servers, are typically used for the training of the models that are later deployed on the Jetson.

Jetson modules allow the use of GPU computing for deep learning models developed with PyTorch. Additionally, to enhance deep learning efficiency, models can be converted into optimized engines using TensorRT [102], a practice recommended for deploying models on NVIDIA hardware, which leads to faster inference speeds [103]. TensorRT is an NVIDIA library designed for high-performance deep learning inference, which includes optimizations for NVIDIA hardware.

The transition to export weights from PyTorch-trained models to TensorRT involves converting the trained deep learning models into a format that is optimized for inference on NVIDIA GPUs. This process begins with the trained model in PyTorch. The model is then exported to an intermediate representation, often using Open Neural Network Exchange (ONNX)[3], which standardizes the model format for use across different deep learning frameworks [104]. Once in ONNX format, the model is ready to be optimized by TensorRT, which analyzes the network to fuse layers, optimize kernel selection, and apply other enhancements that reduce memory footprint. The optimization process is automatically tailored to the unique architecture of the GPU, making it more effective when it is performed directly on the intended target system. By converting PyTorch models to TensorRT, deep learning models can achieve faster inference times, reduced memory usage, and the ability to choose precision formats (such as FP16 or INT8) that balance speed and accuracy.

Several studies have leveraged NVIDIA Jetson modules for deploying deep learning models in various computer vision applications. For instance, in [26], the Jetson TX2 is employed for ship detection, showcasing the utility of Jetson modules in maritime object detection. The work in [105] explored a comparison of marine object detection methods using the SMD dataset [17] on the NVIDIA Jetson Xavier AGX. In the field of instance segmentation, the work in [106] utilized the NVIDIA Jetson AGX Xavier

[2]https://developer.nvidia.com/embedded/jetson-modules
[3]https://github.com/onnx/onnx

for real-time instance segmentation in driving traffic videos, showing the capability of the module to handle complex vision tasks in real-time scenarios.

While the NVIDIA Jetson modules have been effectively utilized in various object detection and recognition tasks, there is a notable absence of research focusing on tailored architectures for real-time ship segmentation deployable on these embedded systems. This highlights a significant opportunity for innovation in developing efficient, real-time processing solutions specifically designed for maritime monitoring applications.

Chapter 4

ShipSG: Ship Segmentation and Georeferencing Dataset

In Chapter 3, we explored the current state of maritime situational awareness, highlighting the critical need for robust and efficient ship recognition methodologies and the challenges associated with georeferencing ships using maritime monitoring footage. This exploration underlined the limitations of existing datasets in supporting the development and evaluation of advanced ship recognition and georeferencing techniques. Motivated by these insights, the creation of a comprehensive dataset that includes precise annotations for ship segmentation and accurate georeferencing has become paramount. This chapter presents ShipSG, a novel dataset for ship segmentation and georeferencing using images from a fixed oblique perspective at maritime facilities. ShipSG serves as a foundational component of this thesis, enabling the evaluation of existing instance segmentation methods as detailed in Chapters 5 compared against the custom architecture proposed in 6. Additionally, the dataset has been instrumental in the quantitative assessment of our georeferencing approaches, as outlined in Chapter 7. A further description of the dataset is given in [BCP-II]. **The dataset was made public and is accessible upon request**[1].

4.1 Dataset Overview

The ShipSG dataset dataset was introduced in [BCP-II] for the development and evaluation of instance segmentation and georeferencing methods using computer vision and deep learning, thus advancing the research field of ship recognition for maritime

[1] https://dlr.de/mi/shipsg

situational awareness. Some samples of ShipSG with annotated ship masks can be seen in Fig. 4.1. The dataset contains:

- 3505 images (2028 × 1520 pixels) from two cameras with static oblique view to the Doppelschleuse, Bremerhaven, Germany.
- 11625 annotated ship masks grouped in seven classes (see Fig.4.3) with COCO format [56].
- 3505 geographic positions, consisting of the latitude and longitude of one of the masks within each image.
- 3505 Automatic Identification System (AIS) ship types[2], one per geographic position annotated.
- 3505 ship lengths, one per geographic position annotated.

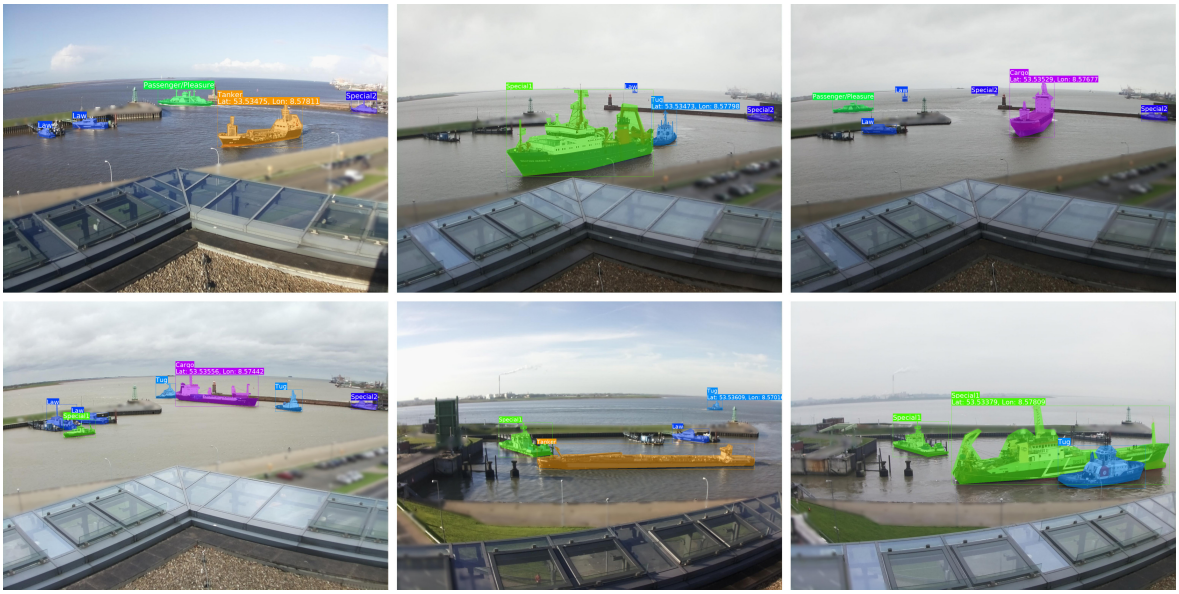

Figure 4.1: Visualisation of ShipSG dataset samples with annotated ship masks, classes, and one ship position per image. Reprinted from the dataset site with permission from German Aerospace Center (DLR).

The dataset was split into two sets: training and validation. The training set contains 80% of the dataset, with 2804 images, and the remaining 20% is used for validation, with 701 images.

[2]https://coast.noaa.gov/data/marinecadastre/ais/VesselTypeCodes2018.pdf

4.2 Acquisition and Annotation

The ShipSG dataset was collected through two strategically positioned cameras at the Fischereihafen-Doppelschleuse in Bremerhaven, Germany, aiming to cover a broad view of the lock's entrance and adjacent Weser river area (see Fig. 4.2). With a height above water level of 23 meters, these cameras captured the dynamic maritime activities within the port basin, ranging distance up to 400 meters from the cameras, and on a distance of up to 1200 meters on the Weser river. The images were captured under various weather conditions including sunny, cloudy, windy, and rainy days during Autumn 2020. The dataset also ensures privacy by anonymizing non-relevant entities like vehicles and people. This comprehensive collection approach ensures a diverse and realistic dataset for maritime research.

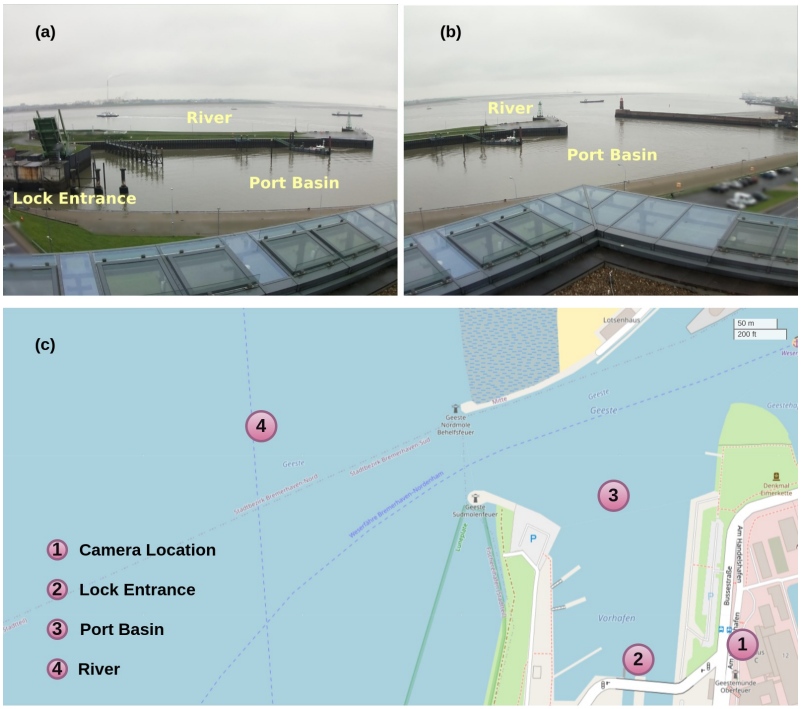

Figure 4.2: View of each camera and identification of important elements in the scene. (a) View of first camera. (b) View of second camera. (c) Notable elements in the scene (OpenStreetMap [107]). Modified from [BCP-II] (CC BY 4.0).

The dataset utilized AIS data to identify ships in each image, accessing real-time positional and static information from ships to annotate images accurately. This included both the exact locations and lengths of the ships. By matching the timestamps of AIS messages with those of captured images, the dataset ensures high precision in ship positioning, limiting the time offset to 100 ms to achieve a close correspondence.

The recommended speed within the port of Bremerhaven is 10 knots (18.5 km/h) to ensure safe and efficient navigation, in accordance with the guidelines provided by the port authority [108], which means a displacement of approximately 0.5 m in 100 ms. Therefore, the impact of the time offset between AIS and image capture can be considered negligible for the precision of the dataset ground truth. This approach allowed for the annotation of 3505 images using accurate AIS data.

Figure 4.3: Examples extracted from the dataset that show the seven ship classes. Each class contains a variety of sizes and orientations of the ships. (a) Cargo, (b) Tug, (c) Special 1, (d) Tanker, (e) Law Enforcement, (f) Passenger/Pleasure, (g) Special 2. Reprinted from [BCP-II] (CC BY 4.0).

By providing AIS ship types with the dataset, we enable users to create their own tailored classes. In our case, we categorized seven ship classes (see Figure 4.3) for the dataset based on an observation of their purpose and visual similarities:

- Cargo: All types of cargo ships.
- Law Enforcement: Police watercrafts and coast guard ships.
- Passenger/Pleasure: Ferries, yachts, pleasure and sailing crafts.
- Special 1: Crane vessels, dredgers and fishing boats.
- Special 2: Research and survey ships, search and rescue ships and pilot vessels.
- Tanker: All types of tankers.
- Tug: All types of tugboats.

To train and validate instance segmentation algorithms, ship masks were manually annotated in each image, identifying the ships and their classes using the LabelMe software [109]. Moreover, ShipSG can also be used not only for the development of ship segmentation but also ship detection algorithms, by considering the surrounding bounding box of annotated masks.

4.3 Summary and Discussion

Featuring 3505 images, 11625 ship masks and the corresponding georeferences, a novel dataset, ShipSG, for ship segmentation and georeferencing using a static oblique view of a port has been presented. This dataset contains images with mask annotations of ships present, and their corresponding class, position and length.

ShipSG stands as a pivotal contribution to the field of maritime research, setting a new benchmark for ship segmentation and georeferencing. The validation of innovative methodologies using ShipSG lays the groundwork for future advancements in maritime situational awareness.

The creation and use of ShipSG is an essential pillar for this thesis, as it allowed the validation of the recognition methods proposed in Chapters 5 and 6. Our proposed georeferencing methods are also quantitatively validated using ShipSG, as presented in Chapter 7. In total, ShipSG has been used in publications [BCP-II], [BCP-III], [BCP-V] and [BCP-VI].

While methods trained on ShipSG were cross-validated with other similar datasets in [BCP-II] to study generalizability to other maritime scenes (see Sec. 5.2), a key limitation is its reliance on only two views of the same area, coupled with the high costs and logistical challenges of new image capture and manual annotation. Therefore, improvements of the dataset will focus on introducing a broader spectrum of data, crucial for mitigating issues caused by the limited variability in real-world annotated data.

Moreover, future iterations of ShipSG could enhance ship recognition algorithms by leveraging AIS data for annotating ship heading, in addition to the ship lengths. The incorporation of this, combined with further annotations such as ship cuboids or keypoints, would offer valuable insights into the development of algorithms that automatically recognize ship heading and dimensions.

Chapter 5

Ship Recognition for Improved Maritime Awareness

We now delve into the initial exploration of deep learning techniques for ship detection and instance segmentation, that allowed further development of tailored solutions for ship recognition in the maritime domain as discussed in subsequent chapters.

Firstly, Section 5.1 shows that ship detection serves as a proof of concept for the feasibility of using deep-learning-based object detection and georeferencing. This proof of concept reveals its potential to be applied to existing problems, as proposed in [BCP-I], [BCP-III] and [BCP-IV]: abnormal vessel behaviour detection, camera integrity assessment and 3D reconstruction. Secondly, the chapter continues with the journey through standard instance segmentation methods shown in 5.2, performed in [BCP-II], setting the stage for the custom developments for real-time ship segmentation and georeferencing provided in Chapter 6 and 7. Therefore, this chapter outlines the impact of ship detection and segmentation in the development of advanced methodologies for the improvement of maritime situational awareness.

5.1 Ship Detection for Maritime Applications

In this section we navigate through the implementations for this thesis in the field of ship detection from monitoring video and images as proposed in [BCP-I], [BCP-III] and [BCP-IV]. We explore how the automatic recognition of the bounding box of ships provides information that can be used by further processes for three different applications. The first application is the detection of abnormal vessel behavior, which is crucial for maritime safety and security, as it enables early identification and mitiga-

tion of potential threats or navigational hazards [110]. The second is the assessment of optical camera obstruction using ship detection, vital to maintain the reliability of surveillance systems, ensuring consistent monitoring quality under various environmental conditions [111]. Finally, the third application discussed in this section is 3D reconstruction of detected ships, which plays a pivotal role in enhancing situational awareness by offering three-dimensional visualizations which improve available semantic information of the situation [112].

5.1.1 Abnormal Vessel Behaviour from Video [BCP-I]

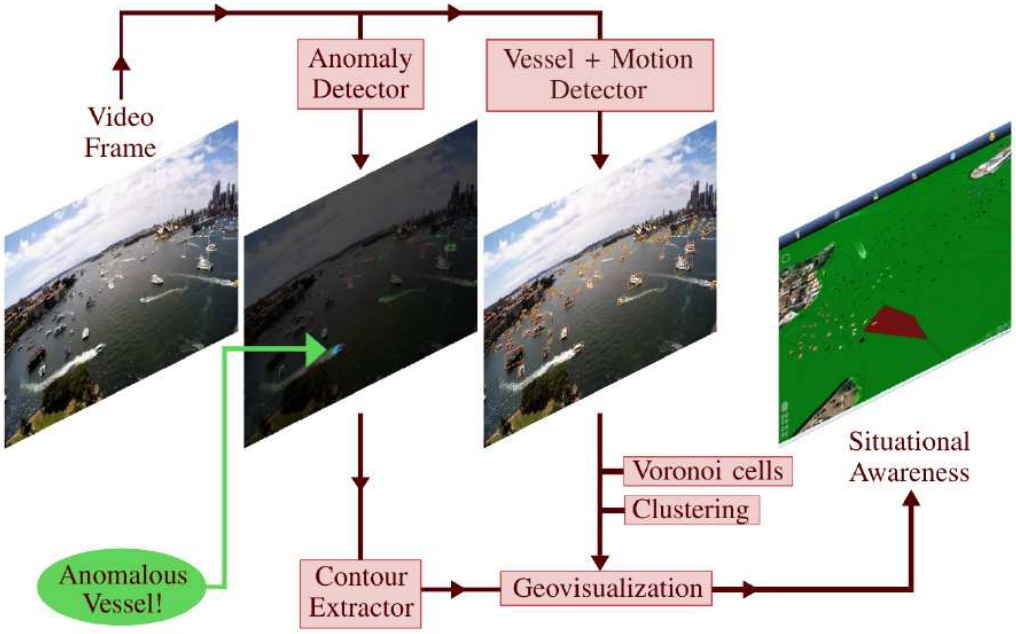

Figure 5.1: Framework proposed in [BCP-I] for maritime anomaly detection from video, including my contributions (Vessel + Motion Detector and Geovisualization). The framework interprets the anomalies using the detections and georeferences for the geovisualization. Reprinted from [BCP-I]. ©2021 IEEE.

The proof of concept for ship detection and georeferencing as tool to support maritime situational awareness, paving the way for this thesis, was conceived in [BCP-I]. The publication presents a framework (see Figure 5.1) for detecting and geovisualizing abnormal vessel behavior using video. This framework aims to enhance maritime situational awareness by offering a tool that leverages Artificial Intelligence (AI) for monitoring and interpreting anomalous vessel activities, thereby improving safety and security in maritime environments.

In my contribution to [BCP-I], the focus was on ship detection, motion analysis, and georeferencing, to facilitate global geospatial localization and visualization of abnormal behavior (in latitudes and longitudes). The georeferenced ships are then represented on a real-world coordinate map, and the motion is used to represent the direction of movement in the form of ship heading. Details for the motion and georeferencing are given in Chapter 7. The anomaly detection module, which was not part of my contribution, uses a Generative Adversarial Network (GAN) for the identification of abnormal behavior. The behaviour is interpreted by combining the output of the anomaly detector with the ship detections, motion and georeferencing. The abnormal vessel behavior was defined by categorizing anomalies based on significant vessel fluxes and depletions, thereby establishing a nuanced criteria that captures a wide range of anomalous patterns without relying on supervised training. Both anomaly detection and ship detection model were trained and validated on an optical sequence video with a resolution of 1280×720 pixels and high density of vessels at the port of Sydney. The scene can be seen in Figure 5.2.

Figure 5.2: Inference of the YOLOv4-CSP based vessel detector. The orange bounding boxes correspond to the detected vessels. Reprinted from [BCP-I]. ©2021 IEEE.

Focusing this section on the object detector used for the framework (motion and georeferencing are explained in 7.2), I defined a custom dataset from the video using 75 random frames for training and 20 for validation. Then, the ships bounding boxes were annotated manually. Given the high density of vessels in each frame, this lead to a total of 4922 and 1387 bounding boxes on the training and validation set, respectively.

The object detector selected was YOLOv4-CSP [67], with CSP-Darknet53 backbone [67], to train and validate with the generated custom dataset. After 234 epochs, the model achieved a peak mean Average Precision (mAP) of 75.86%. Figure 5.2 shows the inference of vessels with the resulting model on a frame that was not part of either the training or validation set for the ship detector. Among the configuration parameters for training, there was the need to increase the image resolution to 1536×1536 pixels to obtain meaningful results with small ships or those located far away from the camera (see Figure 5.2), to the detriment of real-time performance. The goal of the publication was the proof of concept for a framework for ship detection and georeferencing for the identification of abnormal behaviour. Therefore, real-time processing was not a concern and all modules were run on high-end servers in an off-line manner.

The vessel detector presented in [BCP-I] plays a key role within the framework. It identifies vessels and ships in video data, enabling further analysis such as motion detection through optical flow and accurate mapping of vessel locations using georeferencing. This process allows for the identification of vessel movements and anomalies on maps using web services, crucial for improving maritime safety and security. The methodology of bounding box georeferencing and optical-flow based course calculation, which form a significant part of the contribution of [BCP-I], are discussed in Chapter 7.

Moreover, while the presented framework demonstrates promising results in a controlled setting, transitioning to a real-time, real-world application remained unexplored in [BCP-I]. This thesis further explores solutions to bridge this gap with regards to the recognition and georeferencing of ships. As motivated in Chapter 1, georeferencing results are evidently superior when using the mask of ships rather than the bounding box, due to the unnecessary background included within bounding boxes and the inaccuracies of using the bounding box center as the georeferencing point. Selecting an incorrect pixel from the bounding box for georeferencing introduces more error, leading to the consideration of instance segmentation over object detection in the following chapters of this thesis, aiming for more precise georeferencing.

5.1.2 Ship Detection for Integrity Assessment of Camera Obstruction [BCP-III]

The study presented in [BCP-III], focuses on evaluating the resilience and reliability of maritime object detection algorithms under conditions of partial camera obstruction. The work is based around the ShipSG dataset, to explore the effects of various simu-

lated obstructions on detection performance. An obstruction is defined as a physical anomaly, mostly static in nature, that obstructs the camera in close proximity to the lens, potentially requiring intervention to remove or clean. The goal was to quantify the detrimental impact on the system's ability to detect maritime objects correctly, treating the obstruction as a type of fault to investigate its effects on object detection performance.

Figure 5.3: Examples of a ship with different synthetic partial obstruction profiles. (a) No obstruction; (b) 30% bright obstruction at the bottom; (c) 30% dark obstruction at the center; (d) 60% gray obstruction at the right. Reprinted from [BCP-III]. ©2023 IEEE.

My contributions to [BCP-III] lie in the use of ShipSG dataset and the detection of ships, that allow an investigation on how the obstructions affect false positive, misclassification, and false negative ratios, along with the detection score distributions. The work employs Faster R-CNN [71] as the object detector for ship detection. While not specifically designed for real-time applications, its robust architecture offers a strong foundation for object detection tasks. Faster R-CNN was trained on the seven different ship classes of the dataset, using 80% of images for training and 20% for validation, with 1333×800 pixel resolution with the ResNeXt-101 backbone [73]. The training was initiated using pre-trained COCO weights [56] and after additional training on ShipSG of 11 epochs, the model achieved a mAP of 82.6%.

The work of [BCP-III] aligns with the expectations, that true positives (correct detections) decrease and false negatives rise linearly with obstruction. False positives peak from 50 to 60% obstruction, and beyond 60%, as the occlusion covers most of the ship, the detector often fails to recognize any object, leading to an increase in false negatives. Incorporating an obstruction detection step can support maritime stakeholders in identifying camera faults, saving operational time and the subsequent costs.

Further advancements will employ instance segmentation instead of object detection to offer significant integrity assessment improvements, as it allows for precise delineation of ship contours, minimizing background noise in detections. Therefore, segmentation could provide deeper insights into how lens obstructions specifically affect the visibility and classification of ships, by isolating the object from obstructive elements more effectively than bounding boxes. Moreover, more realistic obstructions should be used to increase the understanding of their impact in the recognition.

5.1.3 Ship Detection for 3D Reconstruction [BCP-IV]

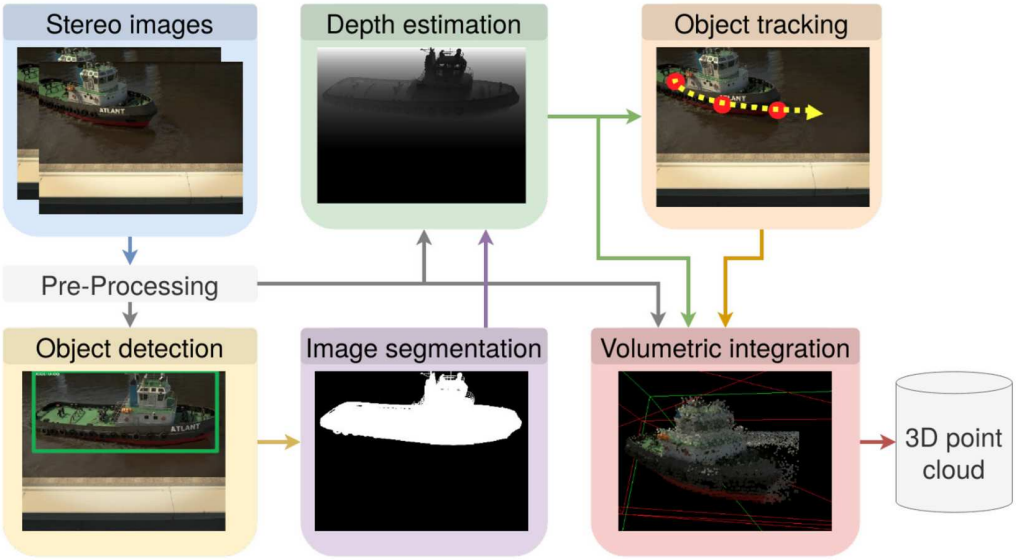

Figure 5.4: Framework proposed in [BCP-IV] for 3D reconstruction of ships using synthetic stereo images. Reprinted from [BCP-IV]. ©2023 Springer.

The work in [BCP-IV] proposes a novel experimental framework (see Figure 5.4) for real-time 3D reconstruction of a detected ship, using synthetic stereo images, and is deployed on an NVIDIA Jetson AGX Xavier. The goal is to enhance maritime situational awareness by processing 2D video data for display into a single consistent 3D display using an embedded system. This transformation from 2D videos into 3D displays provides an intuitive, comprehensive maritime environment understanding with enhanced visualization. The framework is validated using a synthetic and controlled environment created with Blender3D [113], that represents a simulated sequence of a tugboat. It introduces a pipeline prototype for dynamic 3D reconstruction using virtual stereoscopic cameras on a Graphics Processing Unit (GPU)-accelerated em-

bedded device, where object detection plays the role of locating the tugboat on the frames.

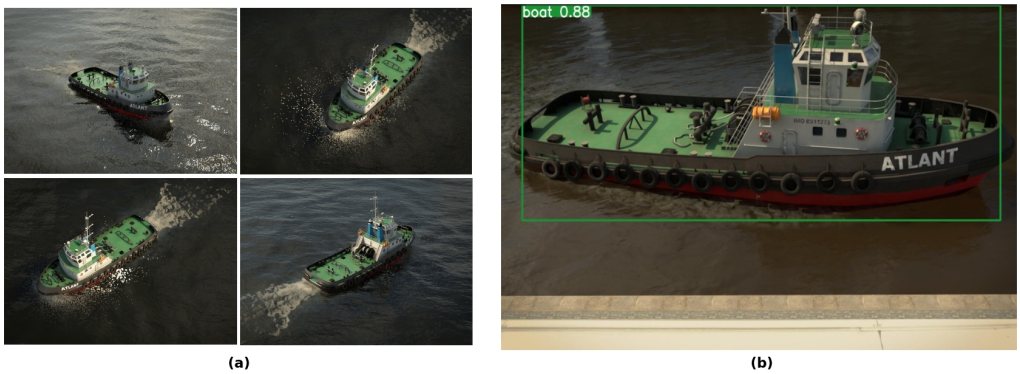

Figure 5.5: Object detection example for 3D reconstruction. (a) Four samples of the rendered dataset for object detection training. (b) An example of tugboat detection using YOLOv5 on one frame of the synthetic sequence for reconstruction. Reprinted from [BCP-IV]. ©2023 Springer.

As part of my contributions to [BCP-IV], YOLOv5 [82] was selected as object detector, in its lightest configuration (nano or n). It was trained on a custom synthetic dataset of a tugboat in various sizes and perspectives (see Figure5.5 (a)), for 50 epochs with an image resolution of 640×640 pixels. For inference, the object detector was deployed, using Pytorch weights, on an NVIDIA Jetson Xavier AGX for real-time processing, using Pytorch [55], achieving a speed of 74 ms per frame and a mAP of 90.7%. The tugboat detector, the fast and accurate detection, enables the framework to focus the rest of the pipeline on the content of the bounding box, therefore supporting the subsequent 3D reconstruction process.

Testing this framework on a synthetic dataset presents challenges in extrapolating results to real-world scenarios. Real-world deployment faces issues such as varied lighting conditions, diverse ship designs, and environmental factors like sea state and weather, which can significantly impact detection and reconstruction accuracy. We observe that the high mAP (90.7%) is a result of the dataset just being constituted by the same boat throughout the whole sequence, with no other boats or classes being present. Addressing these challenges requires robust algorithmic improvements and real-world datasets to ensure the framework's effectiveness in practical maritime monitoring and safety applications.

Moreover, exploring instance segmentation instead of object detection for future work could yield benefits, particularly for accurate 3D reconstruction, where ship segmentation is essential. After the tugboat detector, the framework relies on traditional

segmentation techniques, due to the absence of a real-time capable segmentation solution using deep learning deployable on the NVIDIA Jetson AGX Xavier at the time of the implementation of [BCP-IV]. This underscores the potential of real-time instance segmentation developments deployed on embedded systems shown in Chapter 6. For example, the framework presented in Chapter 6, a deep-learning-based instance segmentation method could provide a more unified approach, enhancing 3D reconstruction efficiency and potentially accuracy.

5.2 Standard Ship Segmentation Using ShipSG [BCP-II]

Building on the foundational work of this thesis in ship detection and its implications for maritime applications, as outlined in the preceding sections, we look now into the experimental evaluation of standard instance segmentation methods on the ShipSG dataset, as presented in [BCP-II]. The use of instance segmentation is motivated by the significant enhancement that it would provide in applications such as abnormal vessel behavior detection with more accurate georeferencing, integrity assessment of camera obstruction with more accurate analysis, and 3D reconstruction of detected ships with the unification of parts in the pipeline.

The creation of the ShipSG dataset (see Chapter 4), provided a comprehensive basis to perform an evaluation of robust instance segmentation methods like Mask R-CNN [72] and DetectoRS [74], as well as real-time methods including YOLACT [77] and Centermask-Lite [79]. The latter evaluations seeking real-time performance involved configurations that sought to balance inference speed with mAP. Therefore, two configurations for each were selected, one deeper and another one lighter, as can be seen in Tables 5.1 and 5.2. All methods initiated training on ShipSG with COCO pre-trained weights. It is notable that in [BCP-II], inference speed was measured using the NVIDIA GV100, a high-end GPU, boasts Tensor Cores for AI acceleration, 32 GB of memory, and over 5000 Compute Unified Device Architecture (CUDA) cores for unparalleled computational performance. Embedded system based deployment will be discussed in Chapter 6.

As shown in Table 5.2, the robust methods, Mask R-CNN and DetectoRS, demonstrated superior mask mAP across all categories when compared to their real-time counterparts. DetectoRS, in particular, achieved the highest overall mAP, underscoring its effectiveness in accurate ship segmentation, however at the highest computational cost, even when using a high-end server.

Table 5.1: Configurations during training for each instance segmentation method evaluated in [BCP-II]. (CC BY 4.0)

Method	**Input Size (Pixel)**	**Backbone**	**Number of Epochs**
Mask R-CNN	1333 × 800	ResNeXt-101	11
DetectoRS	1333 × 800	ResNet-50	11
YOLACT_{550}	550 × 550	ResNet-50	18
YOLACT_{700}	700 × 700	ResNet-101	16
$\text{Centermask-Lite}_{\text{V19}}$	800 × 600	Vovnet-19	17
$\text{Centermask-Lite}_{\text{V39}}$	800 × 600	Vovnet-39	17

Table 5.2: Resulting instance segmentation APs and inference speed per method evaluated. Inference times are measured on a high-end NVIDIA GV100 GPU. Adapted from [BCP-II] (CC BY 4.0).

Method	**mAP (%)**	**mAP_s (%)**	**mAP_m (%)**	**mAP_l (%)**	**Inference (ms)**
Mask R-CNN	73.3	50.3	75.2	77.2	117
DetectoRS	74.7	55.6	75.7	79.2	151
YOLACT_{550}	52.7	8.6	51.5	70.9	28
YOLACT_{700}	58.2	14.0	58.2	75.1	36
$\text{Centermask-Lite}_{\text{V19}}$	63.5	45.5	64.0	65.7	24
$\text{Centermask-Lite}_{\text{V39}}$	64.4	46.1	64.8	66.1	28

For real-time applications, Centermask-Lite showcased better mAP performance, especially in handling small and medium-sized objects, while YOLACT was more adept at segmenting larger objects. However, we observe that small-sized objects are segmented with a significant lower performance than the rest of object sizes for all methods. As explained in Chapter 1, small ship segmentation a critical problem in maritime monitoring and it will be tackled by this thesis in Chapter 6.

While Centermask-Lite in its deeper form exhibits the best trade-off in mAP and inference speed, deployment of instance segmentation on an embedded system, remained open. This fact highlights the ongoing challenge of optimizing for both speed and accuracy in maritime object detection and segmentation. The challenge stemmed from framework incompatibilities with GPU-powered embedded systems, such as the NVIDIA Jetson AGX Xavier. Notably, this system utilizes an architecture based on ARM (Advanced Reduced Instruction Set Computing Machine) [114]. Memory constraints and the ARM-specific architecture compounded the difficulty of deploying Centermask-Lite. The effective deployment of our custom real-time ship segmentation on the edge using an embedded system to fill this gap is shown in Chapter 6.

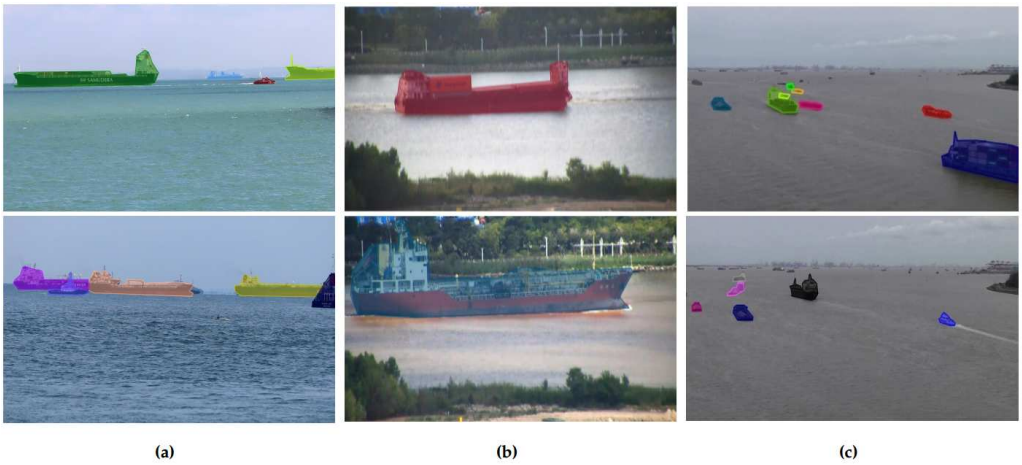

Figure 5.6: Annotated masks on existing datasets to study the generalization of our models. (a) Annotated examples of the Singapore Maritime Dataset (SMD). (b) Annotated examples of Seaships7000. (c) Annotated examples of the dataset by Chen et al. Reprinted from [BCP-II] (CC BY 4.0)

The generalization capability of models trained on ShipSG is crucial for deploying these models in diverse real-world maritime environments, where conditions and scenarios can vary significantly. To assess the generalizability of the instance segmentation models presented in this section, I tested their performance on a mini-dataset of 100 images derived from other maritime datasets, namely the SMD [17], Seaships7000 [60], and the dataset by Chen et al [61]. These datasets, which only used for testing in this work, provide a diverse range of maritime scenarios and vessel types to challenge the ability of the models to accurately segment ships in different conditions. Since these datasets did not contain mask annotations, ship masks were annotated on the 100 images manually (see Figure 5.6). With DetectoRS leading the mAP with 48.6%, the test revealed that models trained on the ShipSG dataset could predict ships from other datasets with reasonable accuracy, given the complex diversity of the mini-dataset.

The work presented in this section underpins the importance of instance segmentation in enhancing maritime safety and security applications. By providing a detailed evaluation of different segmentation methods, it paves the way for the next research focus on optimizing instance segmentation models for real-time applications while deployed on an embedded system, which is tackled in Chapter 6. Moreover, georeferencing from the resulting masks of the evaluated methods of this section is quantitatively analyzed in Chapter 7.

5.3 Summary and Discussion

This chapter showcased the initial exploration in applying deep learning techniques for ship detection and instance segmentation within maritime applications for the improvement of situational awareness, presented in [BCP-I], [BCP-II], [BCP-III] and [BCP-IV]. Along the chapter, the pivotal role of ship detection in facilitating a range of applications has been highlighted, from abnormal vessel behavior detection or camera integrity assessment to 3D ship reconstruction. Moreover, it underscored the enhanced capabilities that instance segmentation offers over bounding box detection, particularly in extracting detailed ship features crucial for applications like georeferencing, which is specially interesting for this thesis to improve maritime situational awareness.

Despite the demonstrated potential of bounding box ship detection and success in controlled [BCP-I] or synthetic [BCP-IV] settings, several challenges and open tasks remain. For example, the improvement of georeferencing and integrity assessment using segmented masks instead of bounding boxes or to unify detection and segmentation in the case of our 3D reconstruction framework. Therefore, this opens the way to use instance segmentation instead of bounding box detection.

In the initial exploration of standard instance segmentation techniques on the ShipSG dataset, we studied the precision of ship feature extraction, crucial for the various maritime applications. This highlights the challenges associated with real-time processing. The evaluation of real-time methods on the NVIDIA GV100 GPU revealed that the best trade-off between computational efficiency and segmentation accuracy, was given by Centermask-Lite.

Though deployment of YOLOv5 (bounding box) using Pytorch weights is reported in [BCP-IV] (see Sec. 5.1.3), the deployment of instance segmentation on GPU-powered embedded systems was not reported in [BCP-II] (see Sec.5.2). Deployment challenges arose from the incompatibility between deep learning and the ARM architectures of GPU-powered embedded systems like the NVIDIA Jetson AGX Xavier. This highlights the need for adaptable methods to enable advanced on-board instance segmentation processing. The move towards embedded systems is essential for practical deployment in dynamic maritime environments, where processing speed and accuracy are paramount. This transition to real-time instance segmentation on embedded systems is addressed in Chapter 6.

Another critical aspect discussed is the importance of instance segmentation for accurate georeferencing of ships. We motivated in Chapter 1 that while ship detection provides valuable insights for several maritime applications, instance segmentation of-

fers a more detailed analysis crucial for precise georeferencing. The ability to extract exact ship contours rather than relying on bounding boxes allows for more accurate positioning of vessels. This capability is explored further in Chapter 7, which delves into the application of the methods proposed for improving maritime situational awareness through enhanced georeferencing.

Furthermore, the initial instance segmentation study shown in this chapter, while promising, highlighted a precision decrease in segmenting small or distant ships. This issue is particularly pertinent for maritime situational awareness, where the ability to accurately identify all vessels, independent their size and within the proximity of the port area is crucial. Chapter 6, Section 6.4, addresses this by introducing a solution that enhances the segmentation of small ships, thereby filling this gap in the initial methodology.

In essence, while the initial exploration into ship detection and instance segmentation revealed significant potential for enhancing maritime situational awareness, it also uncovered several challenges and areas for further development. Specifically, the need for real-time processing on embedded systems, improved detection of small or distant ships, and the utilization of instance segmentation for accurate georeferencing. The subsequent chapters of the thesis aim to address these gaps, presenting custom-tailored solutions that bring these advanced computer vision techniques closer to practical deployment in the maritime domain.

Chapter 6

Advanced Ship Recognition for Real-time Operation

In Chapter 5, we explored the impact of ship detection and segmentation in the improvement of maritime situational awareness and how the development of advanced methodologies can further improve results. Moreover, the deployment of such algorithms on embedded systems has been proven important for practical deployment in dynamic maritime environments, where processing speed and accuracy are paramount. We discussed the applicability of ship detection when deployed on an embedded system, as reported in [BCP-IV]. However, in the case of instance segmentation, task required for more accurate ship georeferencing, deployment on GPU-powered embedded systems remained open. The deployment of instance segmentation methods highlighted the need for adaptable methods to enable advanced on-board processing.

We investigate in this chapter the proposed improvements for real-time ship segmentation proposed in this thesis, as introduced in [BCP-V], [BCP-VI]. First, I present the ScatBlock, a 2D Scattering-transform-based block to be used in the proposed tailored deep-learning architecture. Second, I delve into the design of the custom architecture, ScatYOLOv8+CBAM, that integrates the ScatBlock and attention mechanisms, and demonstrate its superior performance using ShipSG. Thirdly, I propose an optimization to the architecture, followed by the deployment with TensoRT on the embedded system to measure inference times for real-time applicability. Lastly, I address and propose a solution to the precision decrease in segmenting small and distant ships discussed in Chapter 5 and essential for improved maritime situational awareness.

6.1 The ScatBlock [BCP-V]

The ScatBlock, introduced in [BCP-V], is a custom designed 2D scattering-transform-based block that is key for the novel and tailored architecture for ship segmentation developed in this thesis. The 2D scattering transform is a specialized operator that extracts invariant feature representations by decomposing the input image data into a set of scattering coefficients. Each coefficient is a translation-invariant feature map representation that captures spatial and angular variations in an image.

Mathematically, the scattering transform is computed using a set of dilated and rotated versions of a mother wavelet ψ and a low-pass filter ϕ_J, with J being the spatial scale of the transform. The process involves convolving the input image with a predefined filter bank, followed by an element-wise complex modulus operation:

$$U_\lambda = \left|(x * \psi_\lambda)\right| \tag{6.1}$$

where x represents the original input image, and ψ_λ denotes the mother wavelet filter at a specific scale and orientation determined by λ. The output is obtained with a smoothing operation using the low-pass filter ϕ:

$$S_\lambda = U_\lambda * \phi_J \tag{6.2}$$

where S_λ are the scattering coefficients after the smoothing operation, which capture the invariant and descriptive features of the original image. The total number of scattering coefficients (feature maps), $J \times L + 1$, is determined by L, the number of orientations or rotations of the mother wavelet ψ, and J the scale, or also known as order.

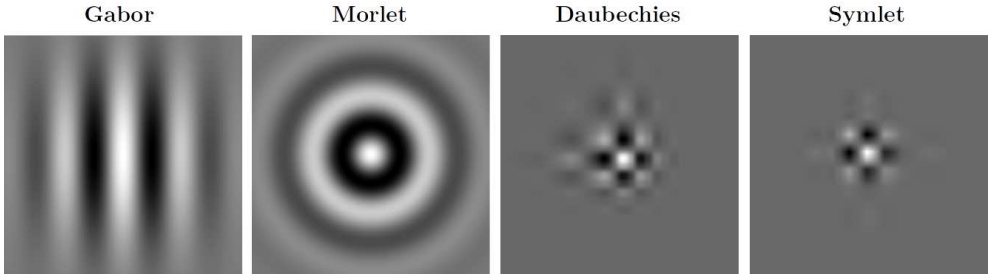

Figure 6.1: Examples of commonly used 2D wavelets. From left to right: Gabor, Morlet, Daubechies and Symlet wavelets.

Commonly used 2D wavelets [115] are represented in Figure 6.1. When used in the 2D scattering transform, Gabor wavelets, though computationally expensive, are

very sensitive to spatial frequencies and variations in textures. Morlet wavelets are characterized by their sinusoidal shape and can perform better with periodical patterns. Daubechies wavelets, can be useful for images with specific geometric patterns. Lastly, Symlet wavelets offer symmetry to preserve features and minimize distortion, which prevents artifacts in the scattering coefficients.

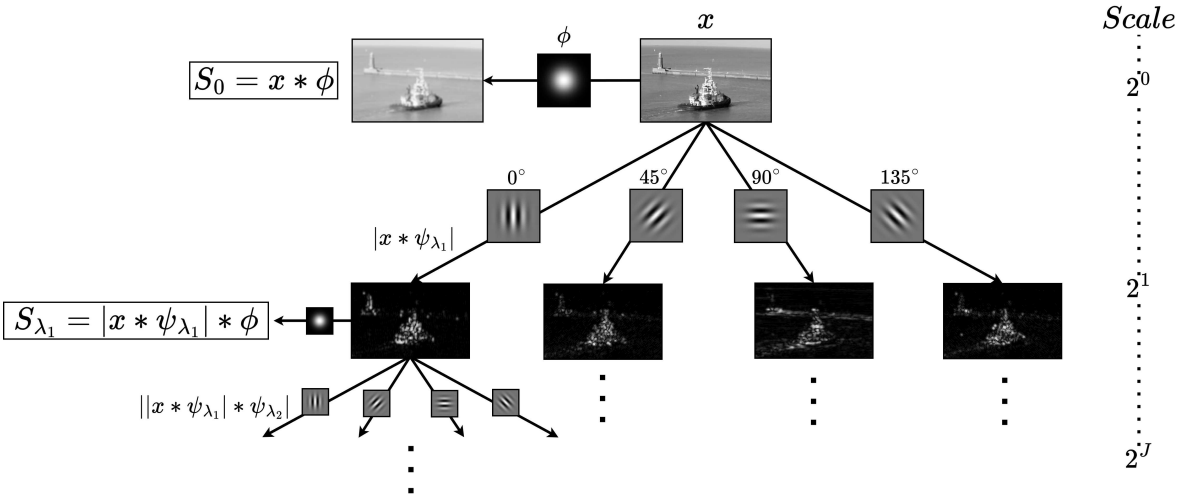

Figure 6.2: Scattering coefficient decomposition of an image x, showing low-pass filtering to obtain S_0 and wavelet modulus operations at orientations 0°, 45°, 90°, and 135° for scale 2^1 to produce first-order coefficients S_{λ_1}. Higher-order coefficients $S_{\lambda_1,\lambda_2,\ldots}$ are obtained at scale 2^J.

An illustrative example of the multi-scale decomposition in a scattering network is shown in Fig. 6.2. The input image x is subjected to a low-pass filtering to produce S_0, and to *Gabor* wavelet convolutions with four orientations at the first scale 2^1. The modulus of each convolved image is taken, followed by another low-pass filtering to yield the first-order scattering coefficients S_{λ_1}. This process is repeated iteratively to produce higher-order coefficients S_{λ_1,λ_2}, with wavelet convolutions at increasing scales 2^J, capturing progressively coarser image features. The network cascades through multiple scales to extract robust, invariant features for image analysis. We observe that each scattering coefficient is a translation-invariant feature map representation that captures spatial and angular variations in the input image.

To incorporate the scattering transform, the ScatBlock architecture (see Fig. 6.3) begins by upsampling the input image to counteract the resolution decrease inherent to the 2D scattering transform, which typically downsamples the input image by a factor of 2^J to reduce computational complexity across scales. Specifically, the input image x is upscaled to $(2 \times H) \times (2 \times W)$, ensuring the output dimensionality

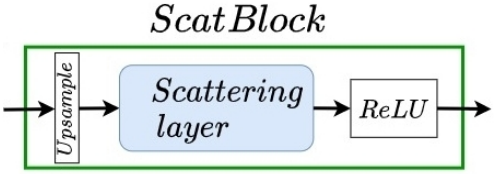

Figure 6.3: ScatBlock, as conceived in [BCP-V], contains an upsample operation, followed by the scattering layer and a Rectified Linear Unit (ReLU) activation. Adapted from [BCP-V]. ©2023 IEEE.

aligns with the input image size, a requirement for subsequent processing layers in the backbone. These sparse feature maps resulting from the scattering transform are then forwarded to a ReLU activation function. The ScatBlock conceived in [BCP-V] uses the first-order coefficients ($J = 1$). Although the computation of subsequent orders is achievable by incorporating more layers, the first-order coefficients hold significant information for mainstream tasks [87]. In this case, the first order was found to be sufficient to enrich the capability of the model to discern and segment ships while maintaining fast computation time.

Regarding implementation, the ScatBlock developed in [BCP-V] uses the approach developed in reference [116] to achieve the scattering transform, using the open-source Python module *pytorch_wavelets*[1]. The scattering transform implementation in *pytorch_wavelets* is based on the Dual-Tree Complex Wavelet Transform (DTCWT). The DTCWT, initially introduced by [117], computes the scattering transform with enhanced efficiency while ensuring theoretical consistency with the traditional approach. The DTCWT, employs wavelet trees for signal decomposition in frequencies, facilitating information capture and directional selectivity. This decomposition allows for a detailed analysis of the isolated frequency content across both horizontal and vertical dimensions of the image, revealing textures and patterns. The approach proposed by [116] in *pytorch_wavelets* to achieve the scattering transform not only enhances efficiency on GPUs due to the DTCWT fast convolution capabilities and suitability for parallel processing, but also ensures the robustness and invariance of the extracted scattering coefficients. A detailed technical description of how the DTCWT is used for a faster scattering transform can be found in [116].

Another option to implement the 2D scattering transform is the package Kymatio [118], which closely mirrors the traditional scattering transform in its approach but falls short in computational speed compared to *pytorch_wavelets*. The use of DTCWT by the latter for GPU-optimized computations significantly accelerates per-

[1]https://github.com/fbcotter/pytorch_wavelets/

formance, making *pytorch_wavelets* the preferred choice for the scattering transform implementations of the ScatBlock due to its efficiency.

As introduced in Chapter 3, the work proposed in [87] demonstrated that the features extracted by the scattering transform are quite meaningful for Convolutional Neural Networks (CNNs). The motivation drawn in [87] lies in the ability of the transform to provide a deep systematic understanding of how invariant features can be captured and utilized for improved deep-learning-based image classification. This approach is particularly relevant for maritime awareness, where the recognition of ships across diverse sizes and types, under varying lighting and weather conditions, demands a robust feature extraction mechanism that can handle the complexities of real-world scenarios. The scattering transform using wavelets are particularly suited for ship recognition because the output coefficients excel at capturing multi-scale geometric and structural features, and the relatively uniform water background provides a good contrast for highlighting ships against the static background. Having this in mind, the ScatBlock has been designed to capture the shape of ships by extracting their inherent geometric and structural properties, and will be added to the custom architecture for ship recognition explained in the following section.

6.2 ScatYOLOv8+CBAM [BCP-V]

To conform the customized ship segmentation architecture presented in [BCP-V], named ScatYOLOv8+CBAM, two additions were implemented.

Firstly, the ScatBlock was blended (Section 6.1) at the beginning of the backbone of YOLOv8 to enhance the input image for instance segmentation, replacing the first convolutional block of YOLOv8. This was motivated by [88]. In their work, the authors explore the efficiency of using the scattering transform to preprocess images before feeding them into CNNs, showcasing how this technique can significantly enhance the quality of feature representations. The relevance of the findings of [88] to the ScatBlock lie in the practical application of the scattering transform to streamline the processing pipeline for real-time instance segmentation and object recognition tasks. The advantage of YOLOv8-like architectures is their deployability on GPU-powered embedded systems, which enables deployment of the custom architecture as well.

The second addition to the network is the CBAM. Introduced by [45], this module advances CNNs by embedding attention mechanisms with minimal computational overhead. It is structured around two key components (see Fig. 6.4). The channel attention module, being the initial component, discerns the importance of each fea-

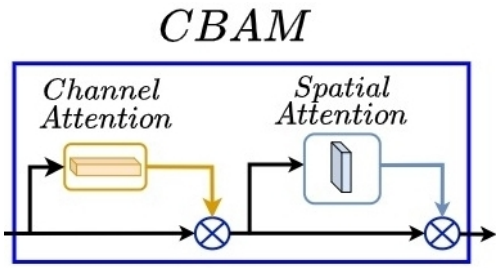

Figure 6.4: Convolutional Block Attention Module (CBAM) module introduced by [45], depicting the channel and spatial attention mechanisms. Adapted from [BCP-V]. ©2023 IEEE.

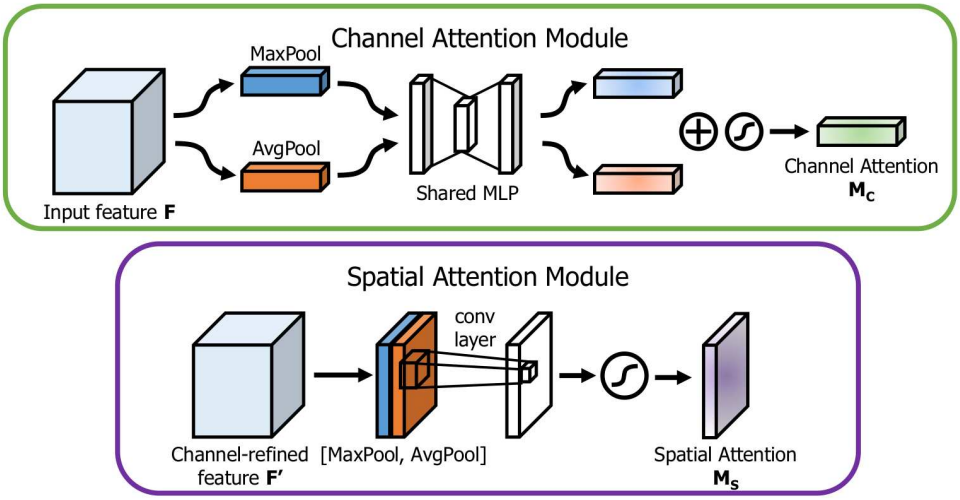

Figure 6.5: Diagram of each attention sub-module in CBAM, taken from [45]. The channel attention module (top) uses both max-pooling and average-pooling operations, followed by a shared Multi-layer Perceptron (MLP), to generate channel attention. The spatial attention module (bottom) applies max-pooling and average-pooling across the channels, then uses a convolutional layer to generate spatial attention.

ture channel, effectively determining "what" is significant in the given feature map and enhancing these channels while suppressing less relevant ones. This is achieved using global average pooling and max pooling, which aggregate information efficiently without heavy computation. The channel attention map M_c is computed as follows:

$$M_c(F) = \sigma(W_1 \cdot (\text{ReLU}(W_0 \cdot \text{AvgPool}(F))) + W_1 \cdot (\text{ReLU}(W_0 \cdot \text{MaxPool}(F)))) \quad (6.3)$$

where F is the input feature map, $\text{AvgPool}(F)$ and $\text{MaxPool}(F)$ are the global average pooled and global max pooled feature maps, $W_0 \in \mathbb{R}^{C/r \times C}$ and $W_1 \in \mathbb{R}^{C \times C/r}$ are the

shared weight matrices of the fully connected layers, ReLU is the Rectified Linear Unit activation function, and σ is the sigmoid activation function.

The subsequent component, the spatial attention module, focuses on crucial spatial regions within the feature map, identifying "where" the important information is located. It applies average pooling and max pooling across the channels, followed by a convolutional layer, which is computationally light. The spatial attention map M_s is computed as follows:

$$M_s(F) = \sigma(f^{7\times7}([\text{AvgPool}(F); \text{MaxPool}(F)])) \tag{6.4}$$

where F is the input feature map, AvgPool(F) and MaxPool(F) are the average pooled and max pooled feature maps across the channel dimension, [;] denotes the concatenation operation along the channel axis, $f^{7\times7}$ represents a convolution operation with a filter size of 7×7, and σ is the sigmoid activation function.

By sequentially applying these attention mechanisms, CBAM refines the feature representation. The efficiency of CBAM comes from its simple yet effective design that avoids complex operations, ensuring fast processing and minimal computational overhead. Since the attention maps of CBAM are broadcast and applied element-wise, they do not alter the height, width, or number of channels of the input feature map.

Through the fusion of channel and spatial attention mechanisms with the YOLOv8 backbone and the 2D scattering transform, the CBAM enables the network to concentrate on pertinent spatial areas while highlighting critical channels, enhancing feature depiction and localization accuracy. According to [45], integrating CBAM into various deep learning frameworks has yielded notable enhancements across a wide array of classification and detection tasks. Specifically, in tasks related to instance segmentation, the CBAM has been instrumental in refining object perimeters and improving the precision of segmented individual objects in images [45]. The integration of CBAM into ScatYOLOv8+CBAM at the head enhances feature extraction through both channel and spatial attention mechanisms. These mechanisms leverage scattering transforms to concentrate on significant regions and features, thereby improving the capability to precisely segment ships within intricate maritime settings.

The proposed ScatYOLOv8+CBAM, as illustrated in Figure 6.6 (a), substitutes the initial convolutional block of the YOLOv8 backbone with a ScatBlock. Unlike the original first *Conv* block of CSPDarknet53 [69] (YOLOv8 backbone), the ScatBlock employs the first-order 2D scattering transform. The ScatBlock is operated only in forward mode, since the rotated wavelets and low-pass filter are fixed, and it does not allow for the backpropagation and filter parameter updates during the training

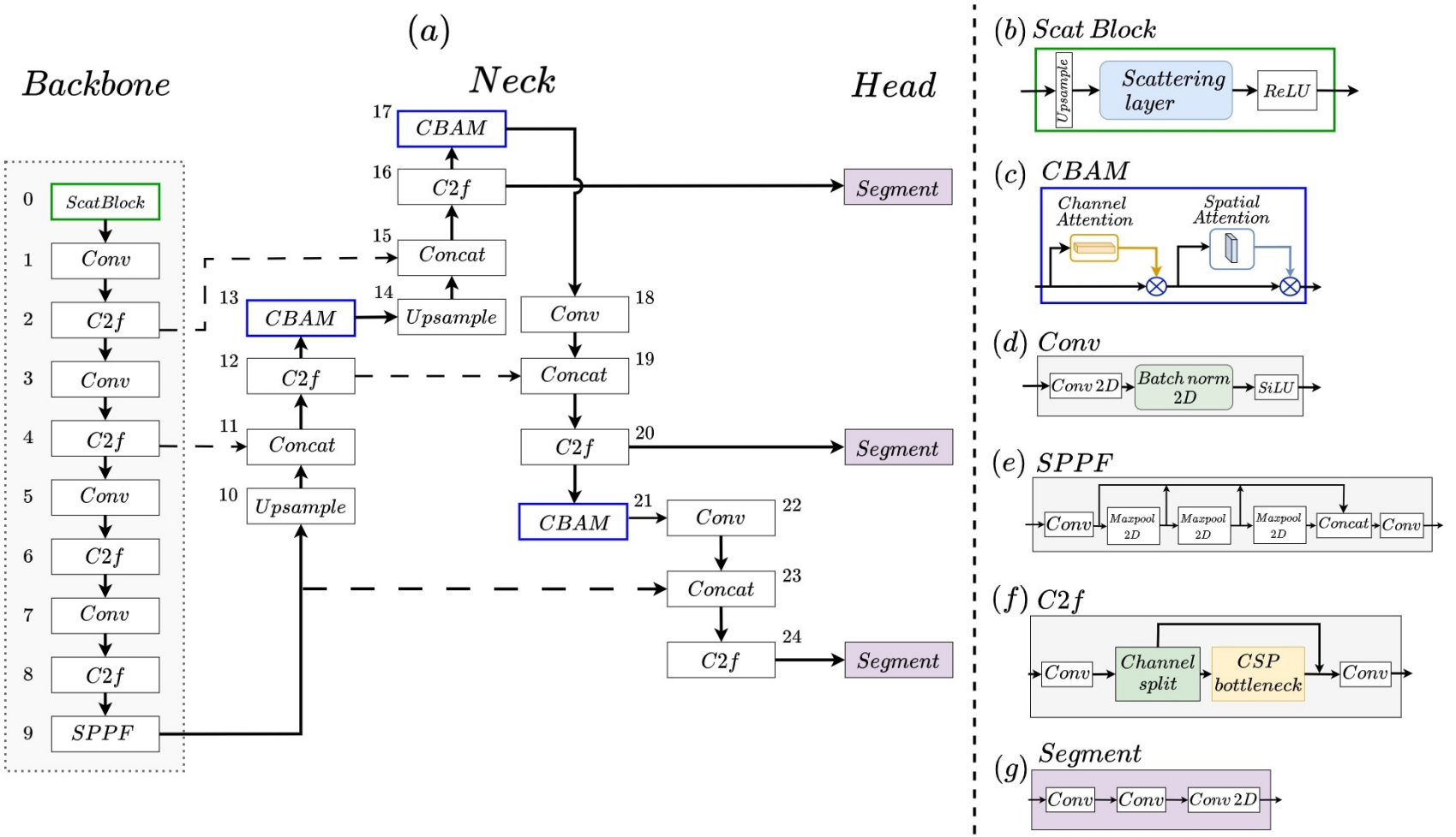

Figure 6.6: The proposed architecture in [BCP-V], with the ScatBlock in the backbone of YOLOv8 and CBAM modules in the head. (a) We place ScatBlock at the beginning of the YOLOv8 backbone to render a set of sparse feature maps, replacing the first Conv block of the original YOLOv8 backbone. The CBAM module is placed at the output of the head blocks of YOLOv8 to distill valuable object shape information for the segmentation task. The numbers next to every block represent the sequential order followed by the implementation, from input to output (b) The scattering block contains an upsample operation, followed by the scattering layer and a ReLU activation (c) CBAM module depicting the channel and spatial attention mechanisms[45] (d) Standard YOLOv8 convolutional block (e) Spatial Pyramid Pooling Fast module of YOLOv8 (f) C2f with split channel operation and a CSP Bottleneck. (g) Segment block of YOLOv8 that performs segmentation. Reprinted from [BCP-V]. ©2023 IEEE.

phase. Following the insights from [119], which applied CBAM to enhance the head of YOLOv5 for aerial object detection, the CBAM block is integrated after the C2f blocks (see C2f block explanation in Section 3.2) within the YOLOv8 neck. This integration serves a dual purpose: assisting the network in identifying areas of interest, specifically ships, and utilizing these identified regions as inputs for subsequent blocks in the head.

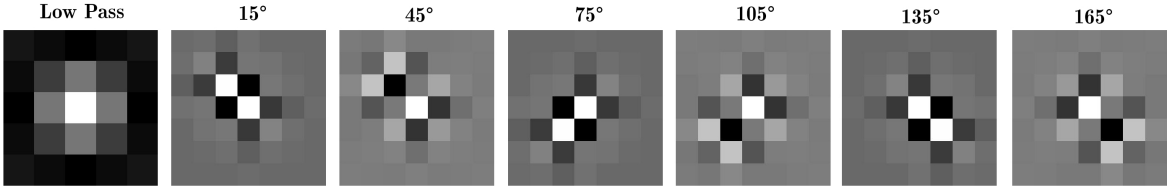

Figure 6.7: Visualization of 2D Wavelet Filters used in the ScatBlock. The first on the left shows the low-pass filter. The rest display the oriented filters, generated by combining and rotating to represent different directional sensitivities of the Symlet wavelet.

As explained in Section 6.1, the implementation of the ScatBlock (see block number 0 of Figure 6.6 (a)) uses the 2D Wavelet transformations by [116], with the open-source Python module *pytorch_wavelets*, which has CUDA support for GPU operations.

With regards to feature map resolution and channel number changes in the proposed architecture, changes in resolution occur primarily due to the Conv blocks, which reduce the height and width, and the Upsample blocks, which increase them. The other blocks (ScatBlock, C2f, SPPF, CBAM, and Segment) do not alter the height and width resolution. Regarding the number of channels, Conv blocks increase the number of channels, while CBAM blocks decrease them. The ScatBlock increases the number of channels (output scattering coefficients). The SPPF and CBAM block maintain the number of channels. The Concat blocks increase the number of channels by merging feature maps, and the Segment block adjusts the number of channels to match the number of output classes.

To evaluate ScatYOLOv8+CBAM, in [BCP-V], the ShipSG dataset was used. Following the common practice in the field and for comparison with the results obtained in Section 5.2, mean Average Precision (mAP) was reported as performance metric. For a fair comparison with the state-of-the-art, YOLOv8 and ScatYOLOv8+CBAM models were trained using an NVIDIA A100 GPU with random weight initialization for all models. The number of training epochs was 300, with the default settings provided by YOLOv8 [66]. The input size used for all models is 640×640 pixels.

In the example of ship segmentation inference on ShipSG, as illustrated in Fig. 6.8, it is observable that the inclusion of the ScatBlock enhances the ships within the

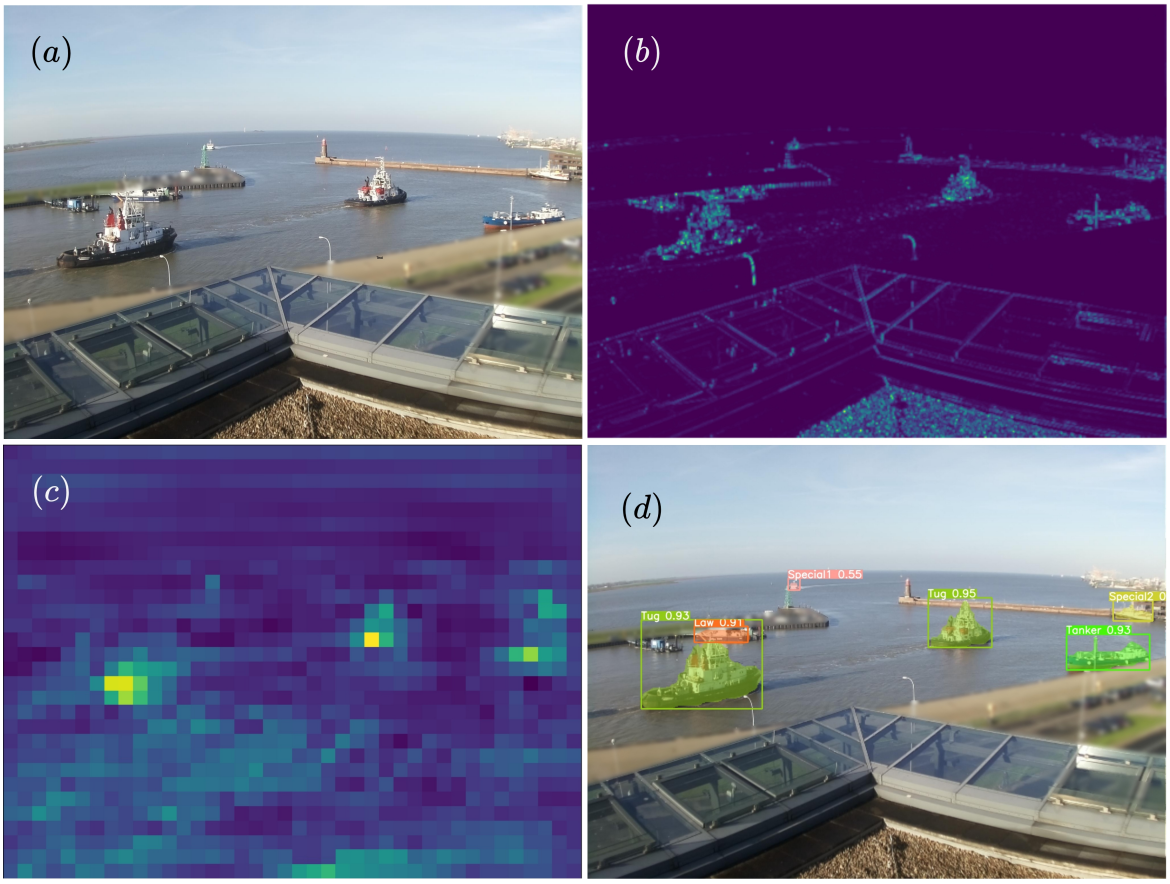

Figure 6.8: Instance segmentation process of ScatYOLOv8+CBAM on ShipSG. (a) ShipSG input sample image. (b) Output of the ScatBlock (here, for visualization, the mean of output channels without S_0). (c) Output of CBAM (module number 21 in Fig. 6.6). (d) ShipSG image with segmented and classified ships using the proposed architecture ScatYOLOv8n+CBAM. Reprinted from [BCP-V]. ©2023 IEEE.

image. This enhancement significantly improves the clarity and visibility of the ship edges, leading to a more defined and distinct outline of the ships present. Regarding the CBAM output, it is noted that the implemented attention mechanisms (both channel and spatial) effectively complement the scattering transform. The attention maps enhance the location of ships within the image while minimizing background influence.

In [BCP-V], I focused on the lightest version of YOLOv8 for the implementation ScatYOLOv8+CBAM, version n, due to its potential for real-time operation. As seen in Table 6.1, the baseline YOLOv8n shows improvement compared to the mAP of previous real-time approaches for ship segmentation on ShipSG studied in Chapter 5, YOLACT and Centermask-Lite. Yet, it does not show advantage against more robust

Table 6.1: Comparison of state-of-the-art segmentation performances on ShipSG with YOLOv8n and ScatYOLOv8n+CBAM. Adapted from [BCP-V]. ©2023 IEEE.

Segmentation model	**mAP (%)**
Mask R-CNN	73.3
DetectoRS	74.7
YOLACT_{700}	58.20
$\text{Centermask-Lite}_{V39}$	64.40
YOLOv8n	70.15
ScatYOLOv8n + CBAM	**75.46**

methods like Mask R-CNN and DetectoRS. ScatYOLOv8n+CBAM achieves 5.31% improvement with respect to standard YOLOv8n, 11.06% improvement compared to Centermask-Lite and 0.76% with respect to the most robust of the previous study, DetectoRS. While the mAP increase is modest when compared to DetectoRS, DetectoRS provided an inference time of 151 ms using a high-end GPU. As discussed in Chapter 5, the deployment of instance segmentation on GPU-powered embedded systems was not reported for the methods presented in the initial study due to the incompatibility between deep learning and the Advanced Reduced instruction set computer Machine (ARM) architectures of GPU-powered embedded systems. The advantage of YOLOv8-like architectures, such as ScatYOLOv8+CBAM, is the deployability on embedded systems. In [BCP-V], the NVIDIA Jetson AGX Xavier was selected as the target embedded system for deployment. The ScatYOLOv8+CBAM model was deployed using native Pytorch weights. This allow us to measure inference times of the new architecture on the system, which will show a great advantage against the previously studied instance segmentation methods.

Table 6.2: Ablation study of YOLOv8 segmentation models and ScatYOLOv8+CBAM additions after training on ShipSG and inference times on the NVIDIA Jetson AGX Xavier. Reprinted from [BCP-V]. ©2023 IEEE.

Segmentation model	**mAP (%)**		**Inference (ms)**	
YOLOv8n	70.35	-	28.7	-
YOLOv8s	71.99	(↑1.64)	32.2	(↑3.5)
YOLOv8m	74.84	(↑4.49)	72.4	(↑43.7)
YOLOv8l	75.89	(↑5.54)	127.1	(↑98.4)
YOLOv8x	76.45	(↑6.10)	196.6	(↑167.9)
ScatYOLOv8n	74.42	(↑4.07)	58.2	(↑29.5)
YOLOv8n + CBAM	70.75	(↑0.40)	29.9	(↑1.2)
ScatYOLOv8n + CBAM	**75.46**	**(↑5.11)**	**59.3**	**(↑30.6)**

In assessing the inference times on the embedded system for the proposed enhancements within ScatYOLOv8+CBAM, specifically the ScatBlock and CBAM, their individual impacts were also examined in the ablation study presented in Table 6.2. The first part of the table outlines the performance metrics of each YOLOv8 model variant. The second part provides the individual and combined contributions of the enhancements introduced in this work. The increments are noted in comparison to the baseline performance of the YOLOv8n model. It can be seen in Table 6.2 that the addition of CBAM produces an increased mAP at a very minimal computational cost. The proposed architecture ScatYOLOv8+CBAM, in the lightest version n, provides a mAP comparable to the deeper and heavier YOLOv8l (75.46% vs 75.89%). However, the proposed model demonstrates a substantially faster inference speed (59.3 ms versus 127.1 ms) on the NVIDIA Jetson AGX Xavier. This marks a significant improvement over the preliminary findings discussed in Chapter 5, Section 5.2.

It has been shown that ScatYOLOv8+CBAM enables the efficient handling of images in maritime environments deployed on embedded systems, facilitating faster and more accurate real-time ship segmentation by leveraging the capabilities of CNNs with the added robustness provided by the scattering transform and attention mechanisms. This indicates its potential viability to enhance real-world maritime situational awareness applications.

It is important to note that one of the goals of this thesis is the improvement of maritime situational awareness leveraging ship segmentation for accurate georeferencing. Therefore, to further validate the ScatYOLOv8+CBAM architecture, the output masks of this architecture are evaluated for ship georeferencing Chapter 7, Section 7.3, as presented in [BCP-V]. This evaluation shows, with higher segmentation mAP, consistent results for georeferencing compared to the georeferencing evaluation of the standard segmentation methods studied in Chapter 5, Section 5.2.

6.3 Optimized ScatYOLOv8+CBAM [BCP-VI]

As it was motivated in Chapter 1, to enhance maritime situational awareness, optimizing for real-time processing capabilities is key. Therefore, ship recognition should operate with the highest possible accuracy and the shortest inference times on embedded systems. This section studies optimizations to ScatYOLOv8+CBAM for a more time-efficient ship segmentation, by circumventing redundancies in the original ScatBlock.

As shown in Section 6.1, the 2D scattering transform typically downsamples the input image by a factor of 2^J to reduce computational complexity across scales. For the ScatBlock, which uses only first-order coefficients, this translates to an output resolution that is half the input resolution. To address this, the ScatBlock upsamples the input image to $(2 \times H) \times (2 \times W)$ (see fig. 6.3). This ensures that the output dimensions match the size of the input image, which is crucial for the compatibility with subsequent YOLOv8 backbone blocks. However, the sequential upsampling and downsampling in image resolution increases computational burden. Furthermore, while the deployment on the NVIDIA Jetson AGX Xavier was documented in [BCP-V] (Section 6.2) using Pytorch weights, the potential for model optimization with TensorRT to achieve more efficient real-time inference was not investigated. The contributions of [BCP-VI] address these areas of improvement, by optimizing the custom architecture ScatYOLOv8+CBAM and performing a comprehensive evaluation for ship segmentation with all model sizes, focusing on real-time processing on the embedded system.

The main optimization focuses on the ScatBlock. As explained in Section 6.1, the ScatBlock was implemented using the open-source Python module *pytorch_wavelets* [116] to achieve the 2D scattering transform. In the optimization, the initial step involves removing the downsampling associated with the scattering transform. This refinement is achieved by bypassing, within the *pytorch_wavelets* package, the division of the image into distinct frequencies and considering all frequency components simultaneously. Following this, the downsampling step of the scattering transform is omitted. The absence of quadrant division means that downsampling would inappropriately cause a mismatch in resolution by presupposing a quadrant-based reduction. As a result, the enhanced ScatBlock does not need the upsample to retain the original resolution in its output feature map. This ensures the preservation of vital image details crucial for accurate segmentation, while simultaneously boosting inference speed.

The optimization of the ScatYOLOv8+CBAM architecture introduced [BCP-V] was presented in [BCP-VI], which removes the upsample from the Scatblock (see upsample in Fig. 6.3) and the downsampling operation eliminated from the *pytorch_wavelets* package [116]. To examine the optimization, the original ScatYOLOv8+CBAM serves as a benchmark for comparison, as well as standard YOLOv8 (see Fig. 6.9). The newly optimized ScatYOLOv8+CBAM was trained across all model sizes (n, s, m, l, x), employing the original parameters specified in [BCP-V]. This includes using an input size of 640×640 pixels, initializing weights randomly, and setting the training period to 300 epochs. To measure inference times, the duration from when an image is

inputted to when predictions are obtained is recorded, now using TensorRT-exported weights on the Jetson AGX Xavier. The approach in [BCP-VI] of using TensorRT contrasts with the prior utilization of Pytorch weights in [BCP-V] (Section 6.2).

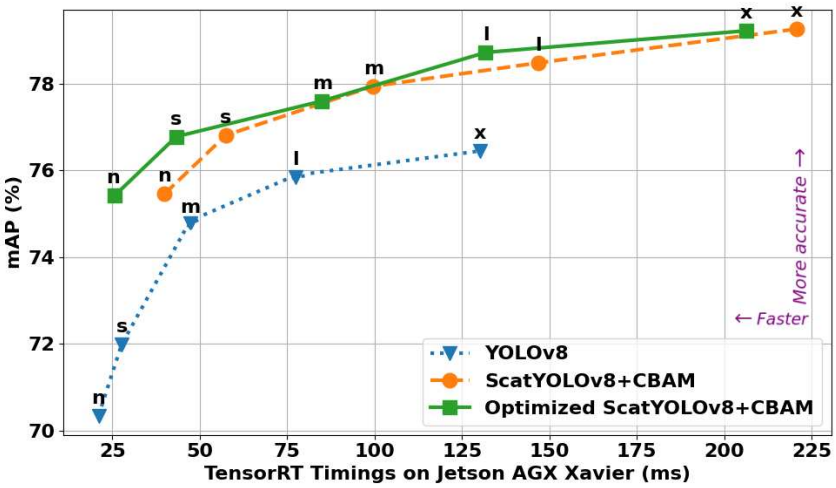

Figure 6.9: mAP vs TensorRT inference times on NVIDIA Jetson AGX Xavier, showing the improved performance speed of the optimized ScatYOLOv8+CBAM relative to the predecessor. Reprinted from [BCP-VI]. ©2024 IEEE.

A comparison of mAP for ship segmentation and TensorRT inference times on the Jetson AGX Xavier can be seen in Figure 6.9. The comparison covers the optimized ScatYOLOv8+CBAM, its predecessor, and the standard YOLOv8. The optimized ScatYOLOv8+CBAM n model provides comparable precision compared with the earlier version and the larger l model of YOLOv8, with mAPs of 75.39%, 75.46%, and 75.89%, respectively. Yet, the optimized model is much faster, with an inference time of 25.3 ms versus 39.9 ms and 77.5 ms for the other models. This shows that the n model of the optimized ScatYOLOv8+CBAM is 36.5% faster than its predecessor with a minor drop in mAP compared to the original ScatBlock-equipped architecture (0.06%). Additionally, the s and m models exceed in mAP over the largest YOLOv8x but offering quicker inference.

The optimized ScatYOLOv8+CBAM speeds up inference against its predecessor, making it more suitable for real-time use. Although the l and x sizes of the optimized model lead in mAP, their slower inference speeds limit their real-time deployment.

In summary, the optimized ScatYOLOv8+CBAM has achieved faster inference speeds on the NVIDIA Jetson AGX Xavier with TensorRT, outpacing the previous version. Moreover, it surpasses standard YOLOv8 models in mAP. However, while

shallower models demonstrate significant improvements in accuracy and speed compared to state-of-the-art ship segmentation on ShipSG, deeper and larger models exhibit a noticeable slowdown. This slowdown suggests an increase in computational complexity within the CNN when it processes scattering coefficients in larger models, opening room for further improvements that will be discussed in Section 6.5.

6.4 Enhanced Small Ship Segmentation Using Higher Resolution Images [BCP-VI]

In Chapter 5, we observed a mAP decrease in segmenting small and distant ships, especially for the initially considered as real-time methods (see Table 5.2). As motivated in Chapter 1, the recognition of small and distant ships in maritime footage holds significant implications for navigation, safety, and security. Typically, object detectors and instance segmentation methods reduce image size for quicker inference, losing critical details in the image. On the other hand, high-resolution deep learning approaches strain memory and computational resources, particularly on embedded systems with limited capacity. Innovating segmentation architectures for such systems is crucial to overcome these challenges. In [BCP-VI], the proposed batch-processed Slicing Aided Hyper Inference (SAHI), combined with the optimized version of ScatYOLOv8+CBAM, advance the state-of-the-art in small ship segmentation using embedded platforms.

Existing deep-learning approaches use image super-resolution or incorporate additional network blocks [21, 91], which is not ideal for embedded systems due to memory constrains. The SAHI framework [93] improves the recognition of small objects in high-resolution images by dividing images into overlapping patches that retain their original size. Then, object detection or segmentation is performed in sequentially on the patches using a compatible method, such as YOLOv8. The resulting detections are then merged with Non-Maximum Suppression (NMS). In the case of the segmented masks, they are merged using NMS and then combined appropriately with a logical OR operator.

The sequential inference of SAHI limits speed, suggesting batch inference integration could boost efficiency on the embedded system. The work in [BCP-VI] modifies the SAHI framework[2] by adding batch processing for the inference stage, addressing the original sequential slice processing constraint as documented in [93]. By enabling ScatYOLOv8+CBAM (and YOLOv8) to infer masks on multiple slices simultaneously

[2]https://github.com/obss/sahi/

($batch = N_{slices}$), the inference phase is made more resource-efficient. Preprocessing (slicing) and postprocessing (merging) within SAHI remain unchanged.

Splitting an image into slices, to form a batch, optimizes both memory usage and computational efficiency, especially on embedded systems with limited GPU memory. Performing object recognition on the high-resolution images may exceed GPU memory capacity if processed as a whole, leading to out-of-memory errors. By splitting the image, each slice can be processed independently within memory constraints. Batch processing these slices allows the GPU to handle multiple slices concurrently, leveraging its parallel processing capabilities. This approach balances the need to manage memory effectively while maximizing computational throughput, making it ideal for resource-constrained environments like the NVIDIA Jetson AGX Xavier.

Additionally, the slicing mechanism was used for model fine-tuning, with slices of the full-resolution ShipSG images as new training set. This allows the model to focus on improving the ability to segment small ships from full-resolution image slices of the dataset. By incorporating batch inference for inference and targeted fine-tuning during training, the optimized ScatYOLOv8+CBAM with SAHI not only maintains quality for small ship segmentation but also boosts real-time performance on embedded systems.

For the fine-tuning process, a new training dataset comprising 33648 slices from ShipSG images was created, each of 640×640 pixels and incorporating a 20% overlap between slices. This overlap guarantees that sufficient contextual information is retained for objects at the edges, yielding the most favorable experimental outcomes. Per image of full-resolution in ShipSG (2028×1520 pixels), the total number of slices of 640×640 pixels is 12. This augmented dataset facilitated the training of the optimized ScatYOLOv8+CBAM model, initiating as pre-trained weights the models of Section 6.3, and extending with an extra 30 epochs of training. Analogously, this fine-tuning approach was applied as well to standard YOLOv8 models for comparative analysis.

At the inference stage, the SAHI preprocessing includes the real-time slicing of the image from which ships are being segmented. This means that inference times when using SAHI encompass the total time from inputting an image to obtaining predictions, with all processing steps, using TensorRT-exported weights on the Jetson AGX Xavier. For inference, the framework uses the same slicing parameters as during the fine-tune training stage, that is, 12 slices of 640×640 pixels per full-resolution image, with 20% overlap.

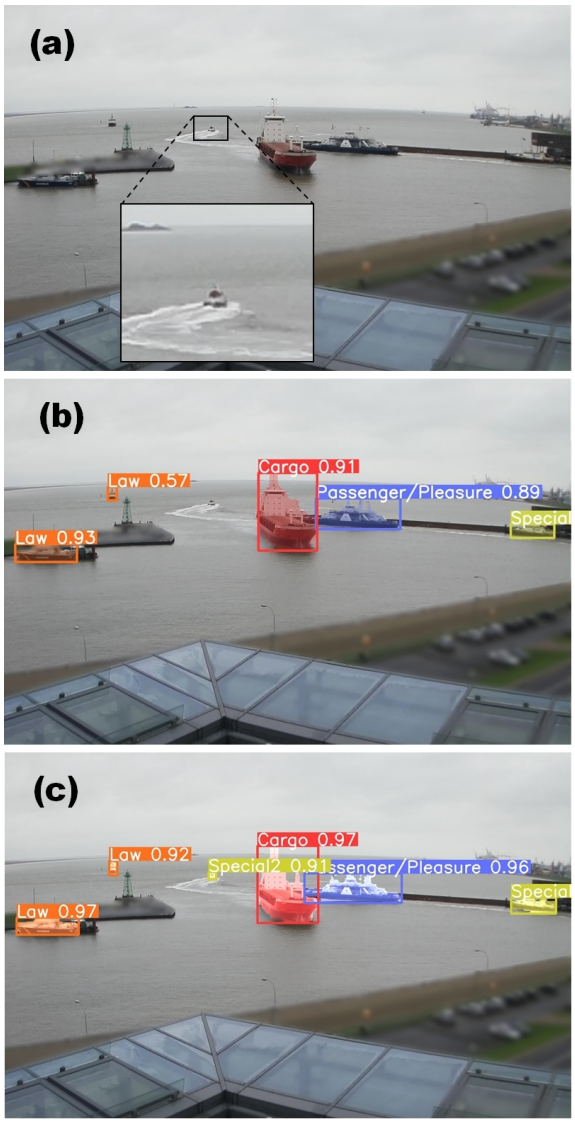

Figure 6.10: Small ship segmentation on ShipSG. (a) Original ShipSG image, with a small ship zoomed in for visualization. (b) Inference using the optimized ScatYOLOv8x+CBAM, where the small ship has been undetected. (c) Inference using the proposed optimized ScatYOLOv8x+CBAM with SAHI, where the small ship appears segmented, at a high confidence score. Reprinted from [BCP-VI]. ©2024 IEEE.

We can see in Figure 6.10 the effectiveness of the SAHI approach, illustrating how a small ship, undetected without SAHI, is accurately segmented, highlighting its significance in boosting maritime situational awareness. Furthermore, the enhanced confidence scores for larger ships in the same figure underscore the robustness of SAHI.

Table 6.3: Comparison of mAP scores for small objects with all model sizes using standard YOLOv8, our proposed optimized ScatYOLOv8+CBAM, and the addition of SAHI. Reprinted from [BCP-VI]. ©2024 IEEE.

Model	mAP small objects (%)				
	n	s	m	l	x
YOLOv8	39.9	40.8	41.8	42.9	43.4
Opt. ScatYOLOv8+CBAM	45.8	47.1	47.2	47.9	48.0
YOLOv8 & SAHI	53.1	54.2	55.0	55.4	55.7
Opt. ScatYOLOv8+CBAM & SAHI	**54.7**	**55.6**	**56.7**	**57.9**	**58.9**

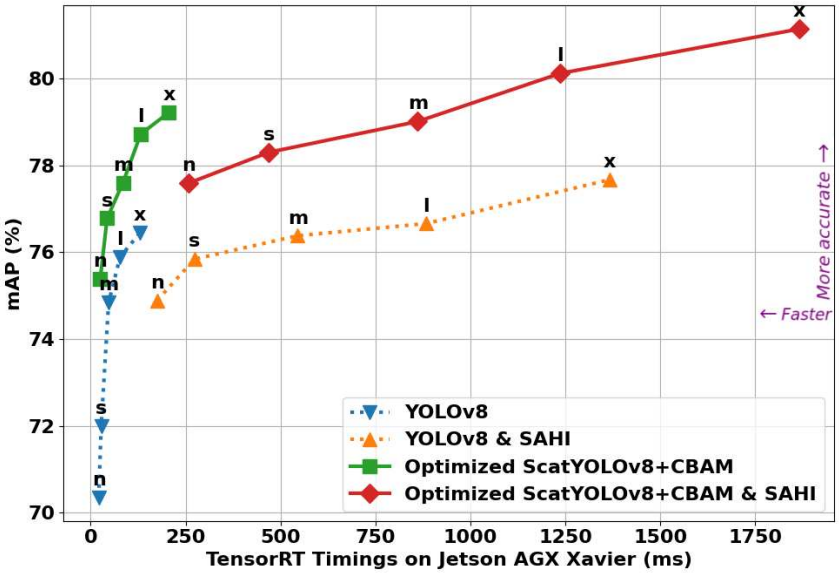

Figure 6.11: mAP vs TensorRT inference times on NVIDIA Jetson AGX Xavier. ScatYOLOv8+CBAM with SAHI provides the best accuracy for ship segmentation. Reprinted from [BCP-VI]. ©2024 IEEE.

Table 6.3 presents the results of the analysis on small ship segmentation across different model sizes. The optimized ScatYOLOv8+CBAM outperforms the standard YOLOv8 for small objects across all sizes. SAHI significantly boosts performance for both standard YOLOv8 and the optimized ScatYOLOv8+CBAM, showing gains between 8% and 11% over configurations without SAHI. Specifically, the optimized ScatYOLOv8+CBAM with SAHI achieves the highest mAP improvements for small objects, ranging from 1.4% to 3.2% over standard YOLOv8 equipped with SAHI. Moreover, the advantage of integrating our optimized model with SAHI becomes more pronounced with model depth, highlighting a scalable improvement in small ship segmentation with increased network complexity. When comparing to the

small ship segmentation performance provided in Chapter 5, Table 5.2, the optimized ScatYOLOv8+CBAM with SAHI in model size s onwards performs as good or better than the best one from the initial study, DetectoRS. It is important to recall that DetectoRS was not compatible with embedded system deployment, which underlines another superiority of the customized architecture.

Figure 6.11 compares the overall mAP (all mask sizes) of models using batch-processed SAHI against those without it on the Jetson AGX Xavier. The ScatYOLOv8+CBAM model with SAHI outperforms all standard YOLOv8 models in ship segmentation accuracy. The ScatYOLOv8+CBAM n with SAHI not only achieves higher mAP than the largest YOLOv8x with SAHI but also performs substantially faster, sparing resources of the embedded system even on full-resolution images.

Compared to its own non-SAHI version, the optimized ScatYOLOv8+CBAM with SAHI in n size shows equivalent mAP to a the m model without SAHI, albeit with increased computation time of $\sim$170 ms, significantly enhancing small ship segmentation by 7.5%, as shown in Table 6.3. This shows that the benefits of SAHI in densely populated maritime areas where monitoring small, distant ships in real-time is crucial for safety and security, carry considerable associated computational costs.

During the tests, it was measured that 25% of the processing time during SAHI batch-inference goes to slicing and merging the masks form the slices, suggesting room for optimization.

6.5 Summary and Discussion

This chapter presented an advancement in the field of real-time ship segmentation with the design of ScatYOLOv8+CBAM. In lightest version n, the architecture demonstrated its capability on ShipSG with a mAP of 75.46%, 5% higher than baseline, comparable to the performance of larger baseline models but at half inference speed per frame (59.3 ms). We analyzed an optimization of the architecture, that removes the upsampling and downsampling from the ScatBlock to save computing time, and deployed it with TensoRT on the Jetson AGX Xavier to measure inference times for real-time applicability. The optimized ScatYOLOv8+CBAM in model size n performs 36.5% faster than its predecessor, achieving 25.3 ms per frame. Finally, I proposed a batch-processed SAHI to increase the segmentation of small and distant ships that is able to run within embedded system resources. Though bringing along increased computation, the mAP for small ships increased in ranges from 8% to 11% in comparison with the baseline without SAHI. The work presented bridges the transition

from standard methods to real-time instance segmentation on embedded systems, and addresses the ability to accurately identify all ships, independent from their size, and within the vicinity of the port area.

Additionally, the chapter has uncovered various areas for enhancement. Further research could focus on improving ScatYOLOv8+CBAM and advancing the performance of SAHI. As seen in Section 6.3, shallower models show notable improvements in both accuracy and speed over the standard ship segmentation models explored with ShipSG. However, larger models become slower. This indicates that larger models face increased computational demands when handling scattering coefficients, hinting at potential strategies for refinement to be explored.

One possible strategy is making the scattering transform learnable. This would involve introducing adjustable parameters within the wavelet filters (parametrization) by merging deep learning adaptability with theoretical wavelet analysis. However, this is a challenging task due to the complicated balance between maintaining the translation and deformation invariant properties of the transform while allowing for sufficient flexibility to learn from data.

The second possible strategy is enhancing the transform with learnable downsampling and attention mechanisms for a more practical approach for real-time use. For example, learnable downsampling, such as strided convolutions with batch normalization, can compress scattering coefficients while learning to preserve key information. On the other hand, attention mechanisms focus computational resources on significant features, improving processing efficiency. An example is the integration of transformers, that are based on attention and could enable models to combine the invariant feature extraction of the scattering transform with the transformer contextual learning, offering a focus on the most relevant features for the segmentation.

Moreover, it was measured during the experiments that 25% of the processing time during SAHI batch-inference is dedicated to the slicing (preprocessing) and merging of the masks after segmentation (postprocessing). In the pursuit of enhancing SAHI, by using parallel processing, the slicing and merging tasks could be improved. An example is the division of the workload with multi-threading or a pool of processes to enable concurrent execution of slicing and merging, optimizing computational resources and improving efficiency.

Given the slowdown in larger models, is pertinent to address how potential practitioners and users of ScatYOLOv8+CBAM should select the size of the architecture (from n to x), and the use of SAHI, in their specific application. The decision should consider the critical balance between real-time processing demands and compu-

tational limitations. The optimized ScatYOLOv8+CBAM, with its SAHI adaptation for small and distant ships, serves as a guide for such selections. Users should weigh the operational complexity, the necessity for rapid data processing, and accuracy requirements against the computational resources available. By aligning architecture choices with these considerations, practitioners and users can ensure the an effective deployment tailored to their unique application needs. For instance, the optimized ScatYOLOv8+CBAM in smaller configurations (n and s) is ideal for scenarios demanding rapid response with considerable accuracy, such as real-time port surveillance or navigation aid systems. Additionally, though optional, the use of SAHI, could enhance access to small and distant ships approach the maritime infrastructure, when the application requires higher levels of responsiveness. Conversely, in scenarios where computational resources are less constrained, larger configurations (m, l and x) could be leveraged for enhanced precision. This choice underscores the necessity for a strategic approach in deploying these technologies, where understanding the specific maritime application context and corresponding computational trade-offs guides the optimal use of the architecture.

Integrating the methodologies presented in this chapter with other processing chains and sensors can also further enhance maritime situational awareness. This could include displaying real-time georeferenced ships on maps through web services, merging various data sources such as ship tracking, 3D reconstruction and anomaly detection or information from other sensors as thermal imaging or radar.

In conclusion, the results presented in this Chapter establish a new standard for maritime monitoring on embedded systems and create a foundation for future work aimed at enhancing real-time, high-resolution processing within the limits of resource-restricted settings.

Chapter 7

Ship Georeferencing for Maritime Situational Awareness

In Chapters 4, 5 and 6, we explored the creation of a custom maritime dataset, the impact of ship recognition on maritime applications, and the advancements and optimizations for real-time processing on embedded systems with a customized architecture, even with higher resolutions to enhance detail in small ship recognition. Building on these foundations, this chapter studies the georeferencing of recognized ships using monocular images. The aim is the implementation of a method that provides meaningful information from the recognized ships to the situational awareness picture for a better semantic understanding of the maritime situation. This process involves the development of methods for the representation of the recognized ship on a global scale using single images and without prior knowledge of the camera. We call this ship georeferencing.

First, we delve into understanding the concept of homographies for georeferencing, which serves as fundamental for the ship georeferencing techniques proposed.

Then, this chapter presents my proposed ship bounding box georeferencing method and calculation of ship heading direction from optical flow, which form a significant part of my contributions to [BCP-I], in addition to what has been presented in Sec. 5.1.1.

Moreover, I expand upon the georeferencing methodology and show an in depth quantitative studies of the use of homographies for ship georeferencing using ShipSG and the results from Sec. 5.2 and Sec. 6.2, based on [BCP-II] and [BCP-V], respectively.

7.1 Homographies for Image Georeferencing [BCP-II]

Homography [120] is a mathematical concept widely used in the fields of computer vision and image processing to describe the transformation of points between two planes (see Fig. 7.1). This transformation, encapsulated by the homography matrix, allows for the conversion of coordinates between these two planes, highlighting the essence of homography in linking different spatial perspectives.

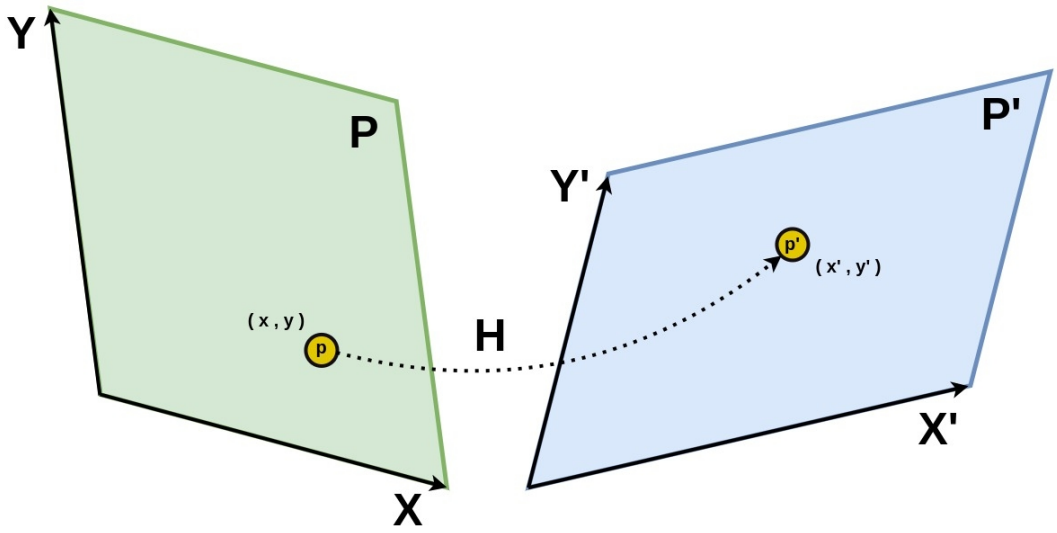

Figure 7.1: Homography between two planes. Plane P and P' represent two different surfaces or image planes. The point p on P and p' on P' illustrate a pair of points that are related by a homography ($Homography(H)$), showcasing how a point in one plane can be mapped to another plane through a projective transformation (dashed arrow).

Essentially, a homography is represented by a 3×3 matrix:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \tag{7.1}$$

When the H matrix is multiplied by a point in homogeneous coordinates, it maps it from one plane to another. It offers a powerful tool for projective transformations, allowing for the mapping of the geometric correspondence between points p and p' in different planes, such that

$$p' = H \cdot p \tag{7.2}$$

where $p = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ and $p' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$, as seen in Fig. 7.1.

The creation of a homography matrix requires the identification of correspondences between points in the two planes, typically two images. Once a sufficient number of point pairs (usually at least four non-collinear points) is established, the homography matrix can be solved using linear algebra techniques. The process involves setting up a system of equations based on the point correspondences and solving for the eight unknowns of the homography matrix (the ninth element is conventionally set to 1 for normalization).

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y'_2x_2 & -y'_2y_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -y'_3x_3 & -y'_3y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -y'_4x_4 & -y'_4y_4 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_nx_n & -x'_ny_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_nx_n & -y'_ny_n \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \\ x'_4 \\ y'_4 \\ \vdots \\ x'_n \\ y'_n \end{bmatrix} \tag{7.3}$$

To solve the homography, these equations are stacked for all correspondences to form a system $Ah = b$. Typically, Direct Linear Transformation (DLT), Random Sample Consensus (RANSAC), or Least Squares (LS) are employed to find the optimal homography matrix that minimizes the re-projection error between the observed and predicted points. LS provided the most accurate results for this application and so was exclusively considered for the remainder of this work. The LS solution to $Ah = b$ is given by minimizing $\|Ah - b\|^2$, which leads to the solution:

$$h = (A^T A)^{-1} A^T b \tag{7.4}$$

where h is the vector containing the eight unknown elements of H. Finally, h is reshaped back into the 3×3 homography matrix H.

Once the homography matrix H is created, it can be used as shown in Eq. 7.2. In computer vision, homography has been extensively utilized across a range of appli-

cations. These include image stitching in panoramic photography, feature matching, camera calibration, 3D reconstruction, augmented reality for overlaying virtual objects onto real-world scenes, motion estimation in video sequences, and perspective correction for rectifying images of planar surfaces. In the field of Geographic Information System (GIS), it has been used to map ground surfaces in surveillance video frames to topographies that are small enough (few hundred meters range) to be approximated as a plane [121, 122]. However they do not provide quantitative assessments of their accuracy.

The ship georeferencing methodology proposed in this thesis also treats the Earth's surface observed by the camera as a planar area for simplification. For example, in the case of ShipSG, given the 23 meter altitude of the cameras and considering the georeferencing range of up to 1200 meters on the Weser river, the curvature of the Earth introduces a minimal height difference. This difference can be approximated by $h \approx \frac{d^2}{2r}$, where d is the horizontal distance between two points on the Earth's surface, and r is the radius of the Earth. This formula derives from the geometric properties of a circle, where the Earth's curvature over a small distance can be represented as the segment height (h) of a circular segment with radius r. For the observation range of 1200 meters, this formula gives the height difference due to curvature to be approximately 0.113 meters. This value is significantly small, especially when compared to the elevation of the camera and and length of the ships. Consequently, we simplify the water surface visible from the cameras as the tangent plane to the Earth's curvature.

By establishing correspondences between known geographic points on the water surface (e.g., buoys, landmarks [BCP-I], or Automatic Identification System (AIS) signals from ships [BCP-II]) and their representations in the image frame, a homography matrix can be computed. This matrix then allows for any point captured on the water surface to be mapped to geographic coordinates. This method not only facilitates the accurate mapping of static water surfaces but also paves the way for dynamic georeferencing applications, such as real-time ship recognition. By leveraging the homography matrix, ships on the water can be precisely georeferenced, offering valuable insights for maritime situational awareness and bridging the digital and physical worlds.

The static view of a monitoring camera allows the water surface captured in the image, with pixel coordinates, to be transformed to the water surface with geographic latitude and longitude coordinates. Homography offers therefore a potent solution that enables ship georeferencing without the need for detailed camera calibration (i.e.,

intrinsic or extrinsic parameters). The work in [BCP-I] explores the homography-based method, and an experimental analysis of the method is presented in [BCP-II] and [BCP-V]. A significant advantage of the proposed georeferencing method, lies in its applicability to any existing camera setup, provided there are identifiable reference points on the surface to create the homography.

In the following sections of this chapter, we explore my proposed method to use homography for the mapping of ships on water surfaces captured by monitoring camera images. We will delve into the detailed methodologies for recognizing and georeferencing ships on the water, further illustrating the practical implications and benefits of homography in real-world scenarios.

7.2 Ship Detection and Georeferencing Using Homographies [BCP-I]

As discussed in 5.1.1, the vessel detector plays a key role within the anomaly detection framework presented in [BCP-I] as it identifies vessels and ships in video data, enabling accurate mapping of vessel locations using georeferencing. The motion detector leverages the ship's bounding box and uses optical flow for motion detection. Optical flow analyzes pixel intensity changes between sequential frames to quantify displacement vectors, indicating motion. I use this displacement to indicate the course of the ship (heading).

Once the vessels are detected per video frame using YOLOv4-CSP [67] (see Sec. 5.1.1), the pixel-based locations and course estimations (heading) are translated to a geographic coordinate system using a homography. This georeferencing process allows for further analytics pursuing vessel abnormal behavior interpretation and for visualization on the situational awareness tool in [BCP-I].

Given that the video used in [BCP-I] has a static perspective, the pixels representing the water surface within its field of view can be treated as a planar area (as discussed in Section 7.1), where all detections of vessels occur. The generation of H from these two planes is done with selected visual landmarks as reference points (see red pins in Fig.7.2), that are used to solve the linear system specified by Eq. 7.3. These references are pixel coordinates from the camera view and latitudes and longitudes, in decimal degrees, of the geographic coordinate system. Once the homography is solved, by taking the center of the bounding boxes provided by the YOLOv4-CSP detector, the georeferencing of vessel positions is possible. The georeferencing, together with

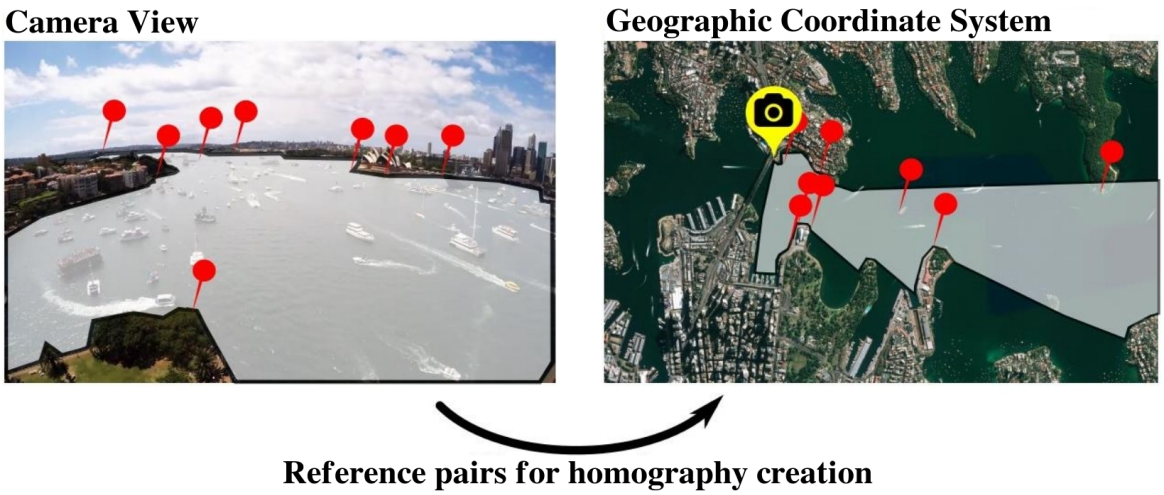

Figure 7.2: Representation of the two planes used to create the homography of publication [BCP-I]. Left: camera view. Right: geographic coordinate system. The white surfaces show the same water plane on both views. The red pins correspond with the selected points on both planes to calculate our homography matrix H. The yellow pin shows the location of the camera. Modified from [BCP-I]. ©2021 IEEE.

the YOLOv4-CSP object detector, allows the geo-location of observed anomalies in the scene and the calculation of vessel heading for their display on a map.

As discussed in Chapter 5, an additional contribution to [BCP-I], along with vessel detection and georeferencing, is the calculation of the heading direction of the vessels. The course of a vessel is defined by its steering direction with respect to the geographic north pole, also called the heading angle.

In the framework presented in [BCP-I], Brox's optical flow [123] is calculated to train the Generative Adversarial Network (GAN) that performs unsupervised anomaly detection. The optical flow was used to determine the direction of the displacement vectors of the detected bounding boxes, represented by blue arrows in Fig. 7.3 (left). This allows the estimation of the angle of the course of the vessels (heading) using georeferencing. From the displacement vectors, the main direction of motion is determined from the median of all displacement directions within the bounding box, represented in Fig. 7.3 by the green arrow. The homography created is then multiplied by the bounding box center and the tip of the displacement arrow to obtain their corresponding geographic coordinates. The tip of the arrow is defined by the cutting point of the median displacement direction with the bounding box edge. Once the two points (center and tip) are georeferenced, the heading angle (θ) is calculated to obtain the course of the vessel using:

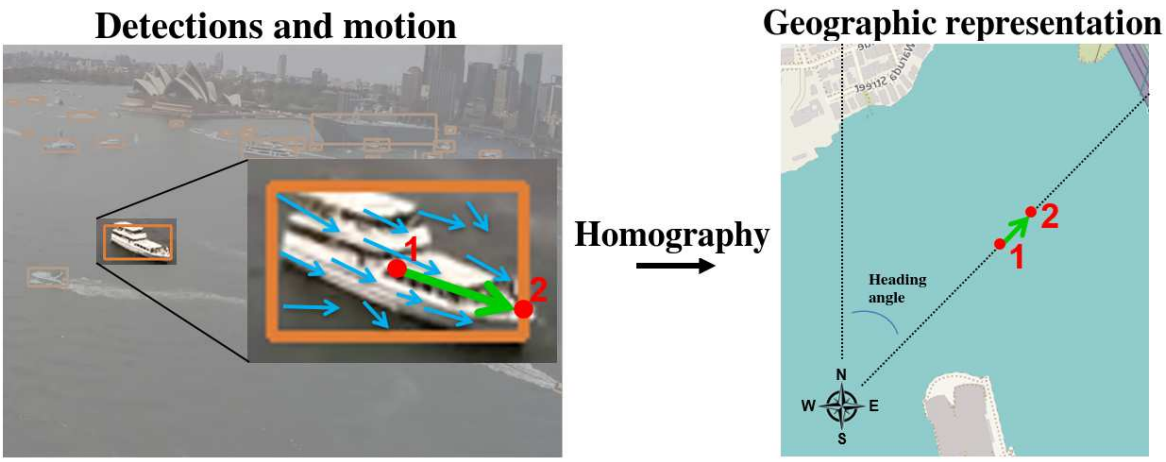

Figure 7.3: Illustrative representation of detected and georeferenced vessel with heading performed in [BCP-I]. Left: example of detection using YOLOv4-CSP. The blue arrows represent displacement vectors (optical flow) with respect to the previous frame, and the green arrow the median direction within the bounding box. The red dots represent the georeferenced points and are defined by the median direction, from the center to the cutting point with the bounding box edge. Right: geographic representation (OpenStreetMap [107]) of the vessel heading angle with respect to north using the created homography.

$$\theta = \text{atan2}\left(\sin\Delta\lambda \cdot \cos\phi_2, \cos\phi_1 \cdot \sin\phi_2 - \sin\phi_1 \cdot \cos\phi_2 \cdot \cos\Delta\lambda\right) \tag{7.5}$$

where $(\phi_1,\ \lambda_1)$ and $(\phi_2,\ \lambda_2)$ represent the coordinates of the georeferenced points (latitude and longitude, respectively), and Δ represents difference.

For display purposes, each vessel detected and georeferenced, is converted to an isosceles triangle, centered around the detection coordinate and whose vertex is pointing in the direction determined by the heading angle. Lastly, the three vertices are provided to a web service for map visualization (see Fig. 7.4).

This section completes my contributions to [BCP-I]. These include vessel detection using YOLOv4-CSP (see Sec.5.1.1), vessel heading calculation using optical flow and georeference for map visualization using a homography. These contributions are paramount to contextualize the identification of the abnormal vessel behaviour, providing useful geographic and spatial information regarding the anomaly.

Since ground truth latitudes and longitudes of the vessels present in the video were not available, the quantitative georeferencing error of the methodology was not reported in [BCP-I]. The results, therefore, serve as a qualitative proof of concept of how ship recognition and georeferencing can improve maritime situational awareness.

Figure 7.4: Visualization of detected and georeferenced vessels with heading. The vessels detected are provided to a web service map (WorldWind [124]) in the form of triangles pointing towards their motion direction. Vessels for which the motion displacement was near zero are represented by circles. The rest of the figure represents the anomaly detection and visualization pipeline presented in [BCP-I]. The red cell represents the interpreted area in which the anomalous ship is navigating. Reprinted from [BCP-I]. ©2021 IEEE.

The lack of availability of ground truth for vessel geographic positions motivates the creation of ShipSG (see Chapter 4), where vessel geographic positions (using AIS) were collected together with the images. The quantitative analysis of homographies for ship georeferencing is shown in the following section.

7.3 Analysis of Ship Segmentation and Georeferencing Using Homographies [BCP-II] [BCP-V]

As motivated in Chapter 1, georeferencing results are superior from the mask of ships, due to the unnecessary background of bounding boxes as well as the inaccurate georeferencing result when using the bounding box center.

I expand upon the georeferencing methodology of the Section 7.2 and show in depth quantitative studies of the use of homographies for ship georeferencing using ShipSG. These studies use the resulting masks from Sec. 5.2 and Sec. 6.2, based on [BCP-II] and [BCP-V], respectively.

In Section 5.2, we have seen the results of an initial evaluation of different segmentation methods for the recognition of ships on the ShipSG dataset. The annotation,

using AIS, of latitudes and longitudes of ships present in the images allow to discuss now the evaluation of georeferencing from the resulting masks provided in [BCP-II].

Following the same principle presented in Section 7.2, the ships, after being segmented, are georeferenced to provide their location to the situational awareness system in the form of latitude and longitude. Since the views of the cameras on ShipSG are static, we can perform a transformation between the camera pixel coordinates (C_x, C_y) and Earth's geographic latitude and longitude (ϕ, λ) in decimal degrees using a homography.

I took 200 samples of the training set of ShipSG to create the homographies for the two camera views, and solved Equation 7.3 to obtain H. The validation set is later used to quantitatively analyse how well the georeferencing performs. The separation of homographies by high or low tides was not found to provide a significant improvement in results. Likewise, the correction of lens distortion prior to the homography calculation did not show an experimental improvement of the method. Therefore, due to their negligible impact, both tidal conditions and lens distortion were excluded from consideration in [BCP-II] and [BCP-V].

Upon the creation of the homographies, I proposed a method to automatically determine the pixel (C_x, C_y) from the masks that most accurately represents the geographic position of the ship.

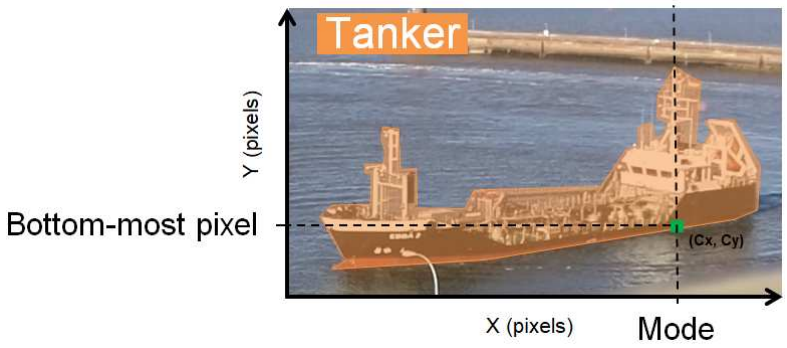

Figure 7.5: Example of segmented ship mask with calculated pixel to be georeferenced (in red, enlarged for visualization). Reprinted from [BCP-II] (CC BY 4.0).

The pixel to be georeferenced represents the intersection between the ship's hull and the waterline, located beneath the bridge or wheelhouse where the navigation antenna is positioned. This is achieved by identifying the bottom-most pixel within the ship mask in the vertical direction (Y) that corresponds to the statistical mode along the horizontal axis (X) (see Fig. 7.5). Then, this pixel is georeferenced using

the homography transformation defined in Equation 7.2, facilitating the conversion of image pixel coordinates into real-world latitudes and longitudes.

For the evaluation, [BCP-II] uses the resulting masks of DetectoRS, which was the method that provided the best mAP result during the initial study (74.7%).

The true latitudes and longitudes obtained via AIS (ϕ_{AIS}, λ_{AIS}) on the validation set of ShipSG, were quantitatively compared with those georeferenced via homography (ϕ_H, λ_H). To facilitate this comparison, both sets of latitudes and longitudes were converted from decimal degrees to Universal Transverse Mercator (UTM) coordinates, allowing for all results to be expressed in meters. Among the metrics employed for the quantitative assessment in [BCP-II], the Georeferencing Distance Error (GDE) is the one that best represents the accuracy of the method as it directly measures the distance in meters between the actual and estimated positions. Therefore, the GDE measures the distance between true (AIS) and georeferenced (H) positions. The haversine equation (Eq. 7.6) is used instead of euclidean distance to take into account the radius (R) of the Earth

$$GDE = 2 \ \cdot R \cdot \arcsin \sqrt{\sin^2 \frac{|\phi_{AIS} - \phi_H|}{2} + \cos \phi_{AIS} \cdot \cos \phi_H \cdot \sin^2 \frac{|\lambda_{AIS} - \lambda_H|}{2}} \quad (7.6)$$

where ϕ_{AIS} and ϕ_H represent the longitudes from the ground truth and georeferencing method using homography, respectively, λ_{AIS} and λ_H correspond to the latitudes, and R is the Earth's radius at Bremerhaven.

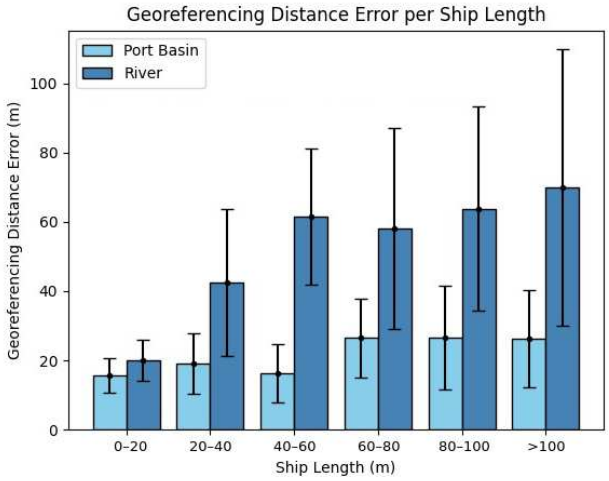

Figure 7.6: Georeferencing distance error per ship length. GDEs and their uncertainties fall within the bounds of the ship length. Reprinted from [BCP-II] (CC BY 4.0).

The GDE, given as *mean* ± *standard deviation* in meters, reaches 22 m ± 10 m for ranges inside the port basin (up to 400 m to the camera) and 53 m ± 24 m on the river (from 400 m to 1200 m). In Chapter 4 it was described that ship lengths along with the ship positions from AIS messages were collected and are used to observe how the GDE changes with ship length and range. As seen in Fig. 7.6, for the smallest ship lengths (0 to 20 m), the GDE within the port basin and river are similar. This demonstrates that the accuracy of pinpointing the georeferenced pixel of the mask increases with the decrease in ship size, regardless of the distance between the ship and the camera. Distance to the ship from the camera is the primary factor influencing the GDE. For ships longer than 20 meters, a marked rise in GDE occurs at distances beyond 400 meters (on the river). In contrast, at distances shorter than 400 meters (within the port basin), the length of the ship has a smaller effect on the GDE compared to distances on the river beyond 400 meters. Despite the challenge of identifying the precise pixel of the mask for georeferencing increasing with the distance to the ship, where each pixel spans a broader geographical area, the GDE remains consistent within uncertainties for each ship length. This suggests that the method provides estimations with a level of accuracy that can be considered contextually appropriate, when the specific operational contexts allow a a deviation of this magnitude within acceptable safety or operational thresholds.

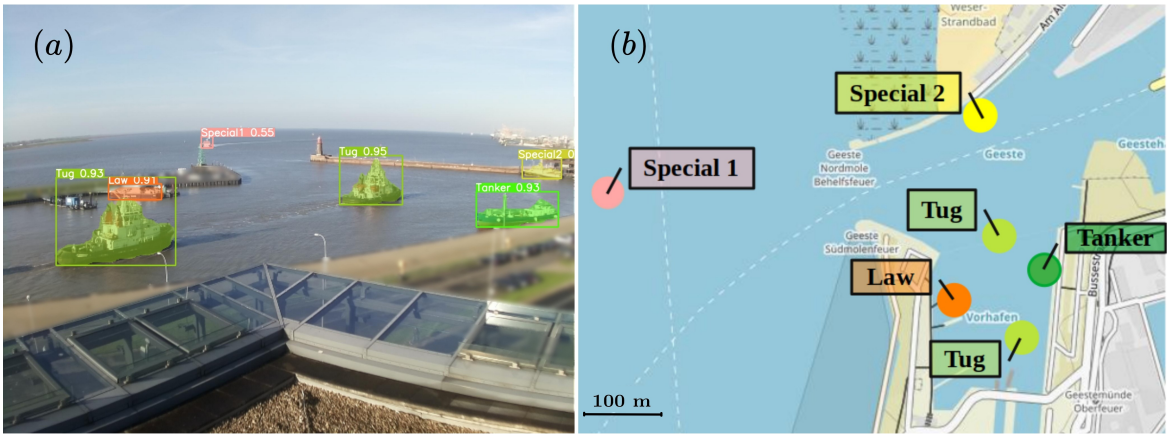

Figure 7.7: Segmented and georeferenced ships using ScatYOLOv8+CBAM and homographies to improve maritime awareness. (a) ShipSG image with segmented and classified ships using the ScatYOLOv8n+CBAM architecture presented in [BCP-V]. (b) Georeferenced ships displayed on OpenStreetMap [107] using the homography-based method of [BCP-II] on segmented masks. Reprinted from [BCP-V]. ©2023 IEEE.

An additional quantitative georeferencing exploration, was done in [BCP-V] with the resulting masks of the proposed ScatYOLOv8n+CBAM (see Sec. 6.2). The georeferencing of the masks resulting from ScatYOLOv8+CBAM, studied in [BCP-V], follows the same procedure to automatically identify the pixel to be georeferenced, and with the same homographies calculated in the previous evaluation with standard methods. The GDE yielded by the masks predicted with ScatYOLOv8+CBAM was of 18 m $\pm$ 13 m within the port basin (up to 400 m range). On the river (range from 400 m to 1200 m), the measured GDE using ScatYOLOv8+CBAM was 44 m $\pm$ 27 m. These results confirm the applicability of the optimized model allowing the benefits described in 6 without any degradation in georeferencing performance. As seen in Figure 7.7, when the georeferenced masks are accessed using web services, this work can support real-time decision making, thus improving maritime awareness.

It is important to note the lack of complex operations in the georeferencing method, which allows an easy deployment on an embedded system. Specifically, these georeferencing steps were measured 0.5 ms on average on the NVIDIA Jetson AGX Xavier, a stark contrast to the inference times of the instance segmentation. This shows that the bulk of the computation time is devoted to mask segmentation. The subsequent pixel searching and coordinate transformation using a homography adds a minimal amount of time to the overall procedure.

Table 7.1: Comparison of the proposed method for ship georeferencing accuracy with existing works.

Source	**System**	**Range to Object**	**Error (m)**
[97]	Radar Antenna + GPS	1000 m	6.5
[98]	Synthetic Aperture Radar	800 km	13 $\pm$ 23
[99]	Opt. Remote Sensing	36000 km	165 $\pm$ 109
[24]	Opt. Camera + GPS + IMU	400 m	20
[BCP-II]	Opt. Camera	400 m	22 $\pm$ 10
[BCP-II]	Opt. Camera	1200 m	53 $\pm$ 24
[BCP-V]	Opt. Camera	400 m	18 $\pm$ 10
[BCP-V]*	Opt. Camera	1200 m	44 $\pm$ 27

*Value not reported in [BCP-V] but calculated for this table.

Given the absence of directly comparable methodologies that simultaneously address the use of monocular cameras without prior camera pose knowledge for fast ship georeferencing, this approach establishes a benchmark in the literature. A comparison of the obtained results with existing ship georeferencing accuracies that use other technologies can be seen in Table 7.1, which is an updated version of Table 3.1 for ship

georeferencing. The most comparable set-up and result is given by [24], that utilizes prior knowledge of camera calibration, and its application was limited to controlled conditions with a single video sequence of a two small ships. Our method obtains similar positioning error, however providing a much more comprehensive study: results using two views, longer ranges, uncertainties, a large variety of ships of different categories and sizes, and does not need prior camera pose knowledge.

In this section, I expanded upon the georeferencing methodology of Section 7.2 to show quantitative results of the use of homographies for ship georeferencing on ShipSG, based on [BCP-II] and [BCP-V]. The results prove that the approach provides useful information from the segmented ships to the situational awareness picture. This information involves the representation of the ship on a global scale using single images and without prior knowledge of the camera, standing out in the literature.

7.4 Summary and Discussion

Ship georeferencing using maritime footage, as discussed in this chapter, stands as a pivotal element in improving situational awareness. We have seen an in-depth exploration of ship georeferencing techniques using single images for enhancing maritime situational awareness, as presented in [BCP-I], [BCP-II] and [BCP-V].

This chapter started with a foundational understanding of homographies, essential for the proposed method and quantitative studies of ship georeferencing. Through the utilization of homographies, we could qualitatively and quantitatively explore the mapping of recognized ships from monocular images to geographic coordinates without previous knowledge about the camera. This is a significant benefit for its versatility; it can be applied any existing camera setup, as long as there are identifiable reference points on the surface to create the homography. This fundamental understanding provided the basis for subsequent discussions on ship detection, heading calculation using optical flow, and the quantitative analysis of ship segmentation and georeferencing using the ShipSG dataset.

We have seen the practical application of ship detection, georeferencing and heading to support abnormal behaviour detection in the maritime domain. The process allows, using a created homography from static camera view, to subsequently map ship locations to geographic coordinates and obtain the heading direction for visualization.

Bounding boxes, while useful for detection, introduce unnecessary background noise and inaccuracies, particularly when the center of the bounding box is used for georeferencing. Masks, on the other hand, delineate the precise contours of ships, pro-

viding a more precise representation for geospatial mapping. This precision is crucial for applications requiring exact geolocated data for improved situational awareness, such as assisted navigation, maritime monitoring or maritime safety and security operations. Therefore, the complete pipeline first segments ships from images, yielding precise ship masks. The segmentation is followed by the automatic search of the pixel within the mask that intersects the ship hull and the water below, at the point where the navigation antenna is located on the ship. The pixel coordinates are then transformed into real-world geographic coordinates through a homography transformation.

Quantitative analyses of ship segmentation and georeferencing using ShipSG uncovered promising accuracy, with a positioning error of 18 m $\pm$ 13 m m in range of up to 400 m and 44 m $\pm$ 27 m from 400 m to 1200 m. The computational time for the pixel search and homography multiplication, averaging 0.5 milliseconds, is remarkably low in comparison to the instance segmentation inference times. This efficiency contrasts with the time-intensive nature of instance segmentation inference on an embedded system, highlighting that the major computational demand lies in mask segmentation. The need for faster and more accurate ship segmentation was addressed by the proposed advanced methodologies for ship segmentation in Chapter 6.

The analyses underscored the reliability and efficiency of the proposed georeferencing methods, laying the ground for their integration into a real-time maritime situational awareness system. The georeferenced latitudes and longitudes of ships derived from this pipeline can be integrated into web services for real-time decision-making, significantly bolstering maritime situational awareness. This integration facilitates the dynamic monitoring of maritime traffic, enabling prompt responses to navigational hazards, environmental risks, and security threats.

In light of the state-of-the-art methodologies in ship georeferencing prior to this research, the proposed method stands out by diverging from traditional reliance on complex camera calibration, high-resolution orthophotos, or Digital Elevation Models (DEMs). The proposed approach provides a scalable, camera-agnostic solution that significantly advances applicability of maritime situational awareness technologies.

Looking ahead, future work in ship georeferencing could delve into further refining accuracy by incorporating additional ship annotations from the ShipSG dataset, such as keypoints and cuboids. These annotations could facilitate a more nuanced consideration of ship dimensions in the georeferencing process, potentially reducing error margins by accounting for the ship's length. Moreover, the exploration of cuboids in conjunction with ShipSG annotations opens new avenues for calculating ship orientations without relying on optical flow calculations. This approach could eliminate

the need for sequential image analysis in anomaly detection, which demands higher frame rates, thereby streamlining the process of determining ship headings and orientations. Future research could also benefit from analyzing sequences of images or videos and different camera perspectives of the same maritime scene, enhancing the depth and accuracy of situational awareness beyond the capabilities demonstrated with the ShipSG dataset. Moreover, further fusion of the proposed image-based real-time ship georeferences with data and processes from multiple sensors, e.g. infrared imaging, radar and AIS, will provide a more comprehensive understanding of maritime situations. Overall, ship georeferencing represents a critical component in enhancing maritime situational awareness, with significant potential for further development and integration into maritime surveillance systems.

Chapter 8

Summary and Conclusion

This compilation thesis addresses the enhancement of maritime situational awareness through advanced ship recognition and georeferencing methodologies. With the creation and utilization of the ShipSG dataset, we have established a benchmark in the field, which facilitates the development of recognition and georeferencing methodologies for maritime applications. This work is driven by the intuition that the integration of deep-learning-based object recognition methods, ship georeferencing and embedded systems can significantly advance the state of real-time maritime monitoring. The chapters within this thesis offer a progression from the creation of a foundational dataset to the implementation and deployment of real-time recognition (detection and segmentation) and georeferencing methods, following strategies to overcome the limitations of the previous existing literature. The key findings from each chapter are summarized below:

Chapter 4 shows the creation of a novel dataset, ShipSG, for ship segmentation and georeferencing with two views of a maritime infrastructure. The dataset contains 3505 images and 11625 ship masks with their corresponding class, position and length. ShipSG marks a significant advancement in the field by setting a new standard for ship segmentation and georeferencing research. Moreover, it plays a crucial role in validating innovative methodologies for maritime situational awareness presented in this thesis. The dataset has facilitated the verification of recognition methods discussed in the chapters dedicated to ship recognition and advanced ship recognition, along with the quantitative validation of the proposed georeferencing methods. The contribution of ShipSG is underscored with its use in publications [BCP-II][BCP-III][BCP-V][BCP-II], evidencing its importance and utility in pushing the boundaries of maritime situational awareness research.

Chapter 5 showcased the initial investigation in applying deep learning techniques for ship detection and segmentation within maritime applications to enhance situational awareness, underscoring the pivotal role of these methods in facilitating a range of applications, such as abnormal vessel behavior detection [BCP-I], camera integrity assessment [BCP-III] and 3D ship reconstruction [BCP-IV]. Despite the demonstrated potential and success for ship detection in controlled or synthetic settings, the exploration of standard ship segmentation methods [BCP-II] was highlighted for the superiority of instance segmentation over traditional bounding box detection in extracting detailed ship features crucial for applications like georeferencing. Challenges remained, notably in the precision, real-time processing, and the deployment of these technologies on GPU-powered embedded systems. The initial studies also pointed out a decrease in precision when segmenting small and distant ships, emphasizing the need for improved detection methods. Overall, while significant potential for enhancing maritime situational awareness has been revealed in this chapter, this exploration also uncovers the need for further development in real-time ship segmentation, the recognition of small and distant ships, and deployment on embedded systems, with subsequent chapters aiming to address these gaps through custom-tailored solutions for practical deployment in the maritime domain.

Chapter 6 builds advances in the field real-time ship segmentation with the design of a customized architecture, ScatYOLOv8+CBAM [BCP-V], and demonstrated its enhanced performance on ShipSG (mAP 75.46%). The chapter analyzed an optimization of the architecture [BCP-VI], that removes the upsampling and downsampling from the ScatBlock to save computing time, and deployed it with TensoRT on the Jetson AGX Xavier to measure inference times for real-time applicability, which brought an acceleration of 36.5% of the inference speed. Moreover, the chapter proposed a batched-processing SAHI to increase the segmentation performance of small and distant ships that is able to run on embedded systems, emphasizing on the use of high-resolution images for a better understanding of the maritime situation. The mAP in small ship segmentation compared to the baseline achieved an improvement of 8% to 11%, however resulted also in a slowdown in inference speed. Choosing the optimal ScatYOLOv8+CBAM model size and incorporating SAHI depends on balancing real-time processing needs with computational capabilities. For critical applications like port surveillance, smaller configurations offer quick, accurate responses. However, larger models improve precision both in general and specifically in small and distant ships. Understanding specific needs and computational trade-offs is key to effective deployment. The presented advances bridge the transition from standard methods to

real-time instance segmentation on embedded systems, and addresses the ability to accurately identify all ships, independent from their size, and within the proximity of the port area.

Chapter 7 identifies ship georeferencing using maritime footage as a pivotal element for enhancing maritime situational awareness. This is done through the application of the ship georeferencing method developed in [BCP-II] and further validated in [BCP-V]. The chapter begins with a foundational understanding of homographies, crucial for the method, followed by qualitative and quantitative analysis of ship latitude and longitude positions from monocular images to geographic coordinates without prior camera knowledge. The qualitative analysis was used in a practical application; abnormal behavior detection in the maritime domain, where georeferences where used for ship positioning and heading direction, leveraging optical flow. In the quantitative analysis, the georeferencing, using ScatYOLOv8+CBAM masks, achieved a positioning error of 18 $m \pm 10$ m for ranges inside the port basin (up to 400 m) and 44 $m \pm 27$ m on the river (from 400 m to 1200 m), which improves upon existing results in the literature. Moreover, the measured average time for the georeferencing, 0.5 ms per frame on the Jetson AGX Xavier, is not significant when compared to the instance segmentation timings. The versatility of this approach and its applicability to any camera setup with identifiable surface reference points, sets a solid groundwork for further work on ship recognition and georeferencing, where the ShipSG dataset can be used as a benchmark. The real-world applicability of georeferenced ship coordinates to support real-time decision-making processes, alone or in combination with other sensors and processes, shows the significant potential of this research to enhance maritime situational awareness.

The outcome of this thesis reflects a significant advancement in maritime situational awareness through the deployment of novel methods both for real-time ship segmentation and georeferencing that are able to run on an embedded system. The introduction of the ScatYoloV8+CBAM architecture, optimized for the ShipSG dataset, demonstrates improved performance metrics over existing methodologies. Focusing on precise ship recognition and georeferencing regardless of class and size, the research has yielded a framework that, when applied, enhances the situational awareness of maritime stakeholders by displaying accurately located ships on digital maps, thereby consolidating knowledge into a user-friendly format which is easy to interpret and act upon. The methodological choices have been centered on maximum accuracy and minimal processing times on embedded systems. This balance of speed and precision

facilitates the integration with other sensor data and services, ultimately advancing maritime operations to be safer, more secure, and efficient. This rationale ensures that the presented work has tangible impacts. This thesis has thus contributed to the body of knowledge with validated approaches that aim to facilitate monitoring in the vicinity of maritime infrastructures. Moreover, the methods and findings presented here offer a solid foundation for future research directed at refining high-resolution, real-time processing in resource-limited maritime contexts.

Chapter 9

Future Work

The outcomes and challenges presented in this thesis suggest several areas that are worth further research.

With regards to the ShipSG dataset, future iterations of the dataset will focus on expanding its diversity by incorporating images from various cameras, and including higher resolution images, for more detailed analysis. This initiative aims to address the current limitations related to the variability and detail of maritime scenes, including more adverse weather conditions. Moreover, further improvements will involve leveraging AIS data for annotating ship heading, and enriching the dataset with additional annotations like ship cuboids or keypoints. These advancements will support the development of more sophisticated algorithms for automatically recognizing ship heading and dimensions, significantly benefiting maritime situational awareness research.

Focusing on improvements of ScatYOLOv8+CBAM and the overall performance of ship segmentation, key areas for advancement include refining the architecture to manage computational demands of larger models more effectively. Potential strategies for improvement involve making the ScatBlock learnable by introducing adjustable parameters within the wavelet filters, or enhancing the ScatBlock by adding learnable modules, such as strided convolutions and integration of transformers, to focus on significant features more efficiently. Additionally, optimizing the slicing and merging tasks in batch-inference processes through parallel processing techniques like multi-threading could significantly enhance efficiency when processing high-resolution images for small ship recognition.

Future work in ship georeferencing should focus on enhancing accuracy by using additional annotations on the ShipSG dataset, specifically keypoints and cuboids, to refine the georeferencing process by considering ship dimensions. Exploring cuboids

alongside ShipSG annotations could also streamline ship orientation calculations, potentially eliminating the need for high frame-rate sequential image analysis in anomaly detection pipelines.

Finally, by integrating the methodologies presented in this thesis with additional processing chains and sensor data, we could contribute to the production of a broad and further enhanced situational awareness picture. This involves creating real-time visualizations of georeferenced ships using web services. For example, a dynamic map where ship positions update constantly, providing a clear picture of maritime situation. Furthermore, by merging diverse data sources such as ship tracking, 3D reconstruction, anomaly detection, and information from other sensors and data sources like thermal imaging, radar or AIS, we can gain a more comprehensive understanding of the maritime environment. This fusion of data will help authorities identify potential hazards, improve navigation safety, and optimize resource allocation for search and rescue operations.

Chapter 10

Publications by the Author for this Thesis

This chapter presents a chronological list of the publications used for this compilation thesis, together with my shares on the works and a short summary of my contributions to each publication.

[BCP-I]

E. Solano, **B. Carrillo-Perez**, T. Flenker, Y. Steiniger, and J. Stoppe, **"Detection and Geovisualization of Abnormal Vessel Behavior from Video,"** in 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), 2021, pp. 2193–2199.

My share on this publication is 35%. Summary of contributions as 2^{nd} author:

- Training and validation of YOLOv4 using custom dataset.
- Implementation of pipeline for ship detection and georeferencing from Sydney harbour video sequence.
- Development of a homography for ship georeferencing and adapted it for the pipeline.
- Calculation of ship course estimation of detected ships using optical flow and homography.
- I advised and participated in the writing of the manuscript.

- Outcome: Proof of concept for ship detection and georeferencing to improve maritime situational awareness.
- Next step: Quantitative analysis of the georeferencing method was not reported. Achieved in [BCP-II] and [BCP-V].

[BCP-II]

B. Carrillo-Perez, S. Barnes, and M. Stephan, **"Ship Segmentation and Georeferencing from Static Oblique View Images,"** Sensors, vol. 22, no. 7, p. 2713, Apr. 2022.

My share on this publication is 90%. Summary of contributions as 1^{st} author:

- Creation and publication of novel dataset for ship recognition and georeferencing (ShipSG).
- In-depth study of four state-of-the-art real-time ship segmentation methods, using ShipSG.
- Quantitative analysis of the homography-based ship georeferencing method, using ShipSG.
- Preparation of the manuscript.
- Outcome: Definition of a pipeline for ship segmentation and georeferencing for ship display on situational awareness maps.
- Next step: Deployment on embedded system was not reported. Achieved in [BCP-IV], [BCP-V] and [BCP-VI].

[BCP-III]

F. A. Costa de Oliveira, **B. Carrillo-Perez**, A. García-Ortiz, and F. Sill-Torres, **"Integrity Assessment of Maritime Object Detection Impacted by Partial Camera Obstruction,"** in 2023 IEEE International Conference on System Reliability and Safety (ICSRS), Nov. 2023, pp. 474–480.

My share on this publication is 20%. Summary of contributions as 2^{nd} author:

- Training and validation of Faster R-CNN for ship detection using ShipSG. The study of obstructions impact was also validated using ShipSG.
- I advised and participated in the writing of the manuscript.
- Outcome: ShipSG dataset impacts other applications for the improvement of maritime situational awareness such as camera integrity assessment.

[BCP-IV]

F. Sattler, **B. Carrillo-Perez**, S. Barnes, K. Stebner, M. Stephan, and G. Lux, **"Embedded 3D reconstruction of Dynamic Objects in Real Time for Maritime Situational Awareness Pictures,"** The Visual Computer, Springer, pp. 1–14, 2023.

My share on this publication is 15%. Summary of contributions as 2^{nd} author:

- Training and validation of YOLOv5 for ship detection on synthetic dataset.
- Deployed detector on NVIDIA Jetson AGX Xavier, using Pytorch weights.
- I advised and participated in the writing of the manuscript.
- Outcome: Deployment of ship detector (bounding box) on embedded system for maritime situational awareness applications.
- Next step: Deployment of instance segmentation (mask) on embedded system with fast inference speed and high accuracy on real dataset (ShipSG). Achieved in [BCP-V] and [BCP-VI].

[BCP-V]

B. Carrillo-Perez, A. Bueno Rodriguez, S. Barnes, and M. Stephan, **"Improving YOLOv8 with Scattering Transform and Attention for Maritime Awareness,"** in 2023 IEEE International Symposium on Image and Signal Processing and Analysis (ISPA), 2023, pp. 1–6.

My share on this publication is 90%. Summary of contributions as 1^{st} author:

- Improved YOLOv8 for ship segmentation with the novel addition of 2D scattering transform and attention mechanism to conform ScatYOLOv8+CBAM. The model was validated on ShipSG.

- Analysis of georeferencing results on ShipSG with the novel proposed architecture.
- Deployment of instance segmentation on embedded system (NVIDIA Jetson AGX Xavier).
- Preparation of the manuscript.
- Outcome: Faster and more accurate ship segmentation and georeferencing, deployed on embedded system.
- Next step: Optimize architecture further and use higher image resolutions, specially for small or distant ship segmentation. Achieved in [BCP-VI].

[BCP-VI]

B. Carrillo-Perez, A. Bueno Rodriguez, S. Barnes, and M. Stephan, **"Enhanced Small Ship Segmentation with Optimized ScatYOLOv8+CBAM on Embedded Systems,"** 2024. IEEE International Conference on Real-time Computing and Robotics (RCAR), 2024, pp. 1–6. (Accepted)

My share on this publication is 90%. Summary of contributions as 1^{st} author:

- Optimization of ScatBlock for faster inference with ScatYOLOv8+CBAM.
- Comprehensive analysis of the use of ScatYOLOv8+CBAM for all model sizes.
- Improvement of the slice prediction method (SAHI) to perform inference in batches, focusing on small or distant ship segmentation using ShipSG.
- Weight serialization with TensorRT for Deployment on NVIDIA Jetson AGX Xavier, allowing faster inference.
- Preparation of the manuscript.
- Outcome: More efficient deployment of ScatYOLOv8+CBAM on embedded system and improved the small and distant ship segmentation accuracy.

Bibliography

[1] E. Engler, D. Göge, and S. Brusch, "Resiliencen–a multi-dimensional challenge for maritime infrastructures," *NAŠE MORE: znanstveni časopis za more i pomorstvo*, vol. 65, no. 2, pp. 123–129, 2018.

[2] F. S. Torres, N. Kulev, B. Skobiej, M. Meyer, O. Eichhorn, and J. Schäfer-Frey, "Indicator-based safety and security assessment of offshore wind farms," in *2020 Resilience Week (RWS)*. IEEE, 2020, pp. 26–33.

[3] F. T. Cetin, B. Yilmaz, Y. Kabak, J.-H. Lee, C. Erbas, E. Akagunduz, S.-J. Lee, and A. I. A. (TURKEY), "Increasing maritime situational awareness with interoperating distributed information sources," in *18th Interantional Command and Control Research and Technology Symposium*, 2013, pp. 9–22.

[4] N. P. Ventikos and K. Louzis, "Risk dynamics for marine systems: towards a bio-inspired framework for dynamic risk assessment," *Transportation Safety and Environment*, vol. 4, no. 3, p. tdac018, 2022.

[5] K. Wang, M. Liang, Y. Li, J. Liu, and R. W. Liu, "Maritime traffic data visualization: A brief review," in *2019 IEEE 4th International Conference on Big Data Analytics (ICBDA)*. IEEE, 2019, pp. 67–72.

[6] B. Belmoukari, J.-F. Audy, and P. Forget, "Smart port: a systematic literature review," *European Transport Research Review*, vol. 15, no. 1, p. 4, 2023.

[7] International Maritime Organization (IMO), "Revised guidelines for the onboard operational use of shipborne automatic identification systems (ais)," Resolution A.1106(29), December 2015, adopted on December 2, 2015, under agenda item 10 during the 29th session of the Assembly.

[Online]. Available: https://wwwcdn.imo.org/localresources/en/OurWork/Safety/Documents/AIS/Resolution%20A.1106(29).pdf

[8] K.-i. Kim and K. M. Lee, "Adaptive information visualization for maritime traffic stream sensor data with parallel context acquisition and machine learning," *Sensors*, vol. 19, no. 5273, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/23/5273

[9] S. Jakovlev, A. Daranda, M. Voznak, A. Lektauers, T. Eglynas, and M. Jusis, "Analysis of the possibility to detect fake vessels in the automatic identification system," in *2020 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*. IEEE, 2020, pp. 1–5.

[10] M. C. Struck and J. Stoppe, "A backwards compatible approach to authenticate automatic identification system messages," in *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*. IEEE, 2021, pp. 524–529.

[11] G. Wimpenny, J. Safar, A. Grant, M. Bransby, and N. Ward, "Public key authentication for ais and the vhf data exchange system (vdes)," in *Proceedings of the 31st International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2018)*, 2018, pp. 1841–1851.

[12] E. Alincourt, C. Ray, P.-M. Ricordel, D. Dare-Emzivat, and A. Boudraa, "Methodology for ais signature identification through magnitude and temporal characterization," in *OCEANS 2016-Shanghai*. IEEE, 2016, pp. 1–6.

[13] M. Balduzzi, A. Pasta, and K. Wilhoit, "A security evaluation of ais automated identification system," in *Proceedings of the 30th annual computer security applications conference*, 2014, pp. 436–445.

[14] Z. Yan, Y. Xiao, L. Cheng, R. He, X. Ruan, X. Zhou, M. Li, and R. Bin, "Exploring ais data for intelligent maritime routes extraction," *Applied Ocean Research*, vol. 101, p. 102271, 2020.

[15] M. Reggiannini, E. Salerno, C. Bacciu, A. D'Errico, A. Lo Duca, A. Marchetti, M. Martinelli, C. Mercurio, A. Mistretta, M. Righi *et al.*, "Remote sensing for maritime traffic understanding," *Remote Sensing*, vol. 16, no. 3, p. 557, 2024.

[16] E. Schwarz, D. Krause, M. Berg, H. Daedelow, and H. Maass, "Near real time applications for maritime situational awareness," *The International Archives of*

the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 40, pp. 999–1003, 2015.

[17] D. K. Prasad, D. Rajan, L. Rachmawati, E. Rajabally, and C. Quek, "Video processing from electro-optical sensors for object detection and tracking in a maritime environment: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 8, pp. 1993–2016, 2017.

[18] F. Li, C.-H. Chen, G. Xu, D. Chang, and L. P. Khoo, "Causal factors and symptoms of task-related human fatigue in vessel traffic service: A task-driven approach," *The Journal of Navigation*, vol. 73, no. 6, pp. 1340–1357, 2020.

[19] T. Flenker and J. Stoppe, "Marlin: An iot sensor network for improving maritime situational awareness," *MARESEC 2021*, 2021.

[20] Z. Chen, D. Chen, Y. Zhang, X. Cheng, M. Zhang, and C. Wu, "Deep learning for autonomous ship-oriented small ship detection," *Safety Science*, vol. 130, p. 104812, 2020.

[21] A. M. Rekavandi, L. Xu, F. Boussaid, A.-K. Seghouane, S. Hoefs, and M. Bennamoun, "A guide to image and video based small object detection using deep learning: Case study of maritime surveillance," *arXiv preprint arXiv:2207.12926*, 2022.

[22] J. T. Hastings and L. L. Hill, *Georeferencing*. Boston, MA: Springer US, 2009, pp. 1246–1249. [Online]. Available: https://doi.org/10.1007/978-0-387-39940-9_181

[23] N. Wawrzyniak, T. Hyla, and A. Popik, "Vessel detection and tracking method based on video surveillance," *Sensors*, vol. 19, no. 23, p. 5230, 2019.

[24] Ø. K. Helgesen, E. F. Brekke, A. Stahl, and Ø. Engelhardtsen, "Low altitude georeferencing for imaging sensors in maritime tracking," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 14 476–14 481, 2020.

[25] S. Mittal, "A survey on optimized implementation of deep learning models on the nvidia jetson platform," *Journal of Systems Architecture*, vol. 97, pp. 428–442, 2019.

[26] H. Zhao, W. Zhang, H. Sun, and B. Xue, "Embedded deep learning for ship detection and recognition," *Future Internet*, vol. 11, no. 2, p. 53, 2019.

[27] H. Ning, Y. Li, F. Shi, and L. T. Yang, "Heterogeneous edge computing open platforms and tools for internet of things," *Future Generation Computer Systems*, vol. 106, pp. 67–76, 2020.

[28] F. Sattler, S. Barnes, and M. Stephan, "A maritime situational awareness framework using dynamic 3d reconstruction in real-time," in *2023 27th International Conference Information Visualisation (IV)*. IEEE, 2023, pp. 334–339.

[29] R. Szeliski, *Computer vision: algorithms and applications*. Springer Nature, 2022.

[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[31] T. Talaei Khoei, H. Ould Slimane, and N. Kaabouch, "Deep learning: Systematic review, models, challenges, and research directions," *Neural Computing and Applications*, vol. 35, no. 31, pp. 23 103–23 124, 2023.

[32] C. Sammut and G. I. Webb, *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[33] J. Chai, H. Zeng, A. Li, and E. W. Ngai, "Deep learning in computer vision: A critical review of emerging techniques and application scenarios," *Machine Learning with Applications*, vol. 6, p. 100134, 2021.

[34] P. Cunningham, M. Cord, and S. J. Delany, *Supervised Learning*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 21–49. [Online]. Available: https://doi.org/10.1007/978-3-540-75171-7_2

[35] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[36] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee, "A survey of modern deep learning based object detection models," *Digital Signal Processing*, vol. 126, p. 103514, 2022.

[37] R. Shanmugamani, *Deep Learning for Computer Vision: Expert techniques to train advanced neural networks using TensorFlow and Keras*. Packt Publishing Ltd, 2018.

[38] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[39] J. Zhang, X. Li, L. Li, P. Sun, X. Su, T. Hu, and F. Chen, "Lightweight u-net for cloud detection of visible and thermal infrared remote sensing images," *Optical and Quantum Electronics*, vol. 52, pp. 1–14, 2020.

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

[41] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[42] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[43] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[44] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[45] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[46] M.-H. Guo, T.-X. Xu, J.-J. Liu, Z.-N. Liu, P.-T. Jiang, T.-J. Mu, S.-H. Zhang, R. R. Martin, M.-M. Cheng, and S.-M. Hu, "Attention mechanisms in computer vision: A survey," *Computational visual media*, vol. 8, no. 3, pp. 331–368, 2022.

[47] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.

[48] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489.

[49] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4055–4064.

[50] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," *arXiv preprint arXiv:1502.03044*, 2015.

[51] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," *arXiv preprint arXiv:1704.06904*, 2017.

[52] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, and T.-S. Chua, "Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[53] G. Brauwers and F. Frasincar, "A general survey on attention mechanisms in deep learning," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[54] M. Gong, D. Wang, X. Zhao, H. Guo, D. Luo, and M. Song, "A review of non-maximum suppression algorithms for deep learning target detection," in *Seventh Symposium on Novel Photoelectronic Detection Technology and Applications*, vol. 11763. SPIE, 2021, pp. 821–828.

[55] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[56] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[57] I. H. Sarker, "Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions," *SN Computer Science*, vol. 2, no. 6, p. 420, 2021.

[58] D. Qiao, G. Liu, T. Lv, W. Li, and J. Zhang, "Marine vision-based situational awareness using discriminative deep learning: A survey," *Journal of Marine Science and Engineering*, vol. 9, no. 4, p. 397, 2021.

[59] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International journal of computer vision*, vol. 111, no. 1, pp. 98–136, 2015.

[60] Z. Shao, W. Wu, Z. Wang, W. Du, and C. Li, "Seaships: A large-scale precisely annotated dataset for ship detection," *IEEE transactions on multimedia*, vol. 20, no. 10, pp. 2593–2604, 2018.

[61] X. Chen, L. Qi, Y. Yang, Q. Luo, O. Postolache, J. Tang, and H. Wu, "Video-based detection infrastructure enhancement for automated ship recognition and behavior analysis," *Journal of Advanced Transportation*, vol. 2020, 2020.

[62] A. Ghahremani, Y. Kong, E. Bondarev *et al.*, "Multi-class detection and orientation recognition of vessels in maritime surveillance," *Electronic Imaging*, vol. 2019, no. 11, pp. 266–1, 2019.

[63] C. Nita and M. Vandewal, "Cnn-based object detection and segmentation for maritime domain awareness," in *Artificial Intelligence and Machine Learning in Defense Applications II*, vol. 11543. International Society for Optics and Photonics, 2020, p. 1154306.

[64] M. Ribeiro, B. Damas, and A. Bernardino, "Real-time ship segmentation in maritime surveillance videos using automatically annotated synthetic datasets," *Sensors*, vol. 22, no. 21, p. 8090, 2022.

[65] E. Teixeira, B. Araujo, V. Costa, S. Mafra, and F. Figueiredo, "Literature review on ship localization, classification, and detection methods based on optical sensors and neural networks," *Sensors*, vol. 22, no. 18, p. 6879, 2022.

[66] G. Jocher, A. Chaurasia, and J. Qiu, *YOLOv8 by Ultralytics*, Jan. 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[67] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-yolov4: Scaling cross stage partial network," in *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, 2021, pp. 13 029–13 038.

[68] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[69] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[70] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "Cspnet: A new backbone that can enhance learning capability of cnn," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 390–391.

[71] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[72] 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.

[73] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[74] S. Qiao, L.-C. Chen, and A. Yuille, "Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 213–10 224.

[75] M. F. Stollenga, J. Masci, F. Gomez, and J. Schmidhuber, "Deep networks with internal selective attention through feedback connections," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2014/file/19de10adbaa1b2ee13f77f679fa1483a-Paper.pdf

[76] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[77] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact: Real-time instance segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9157–9166.

[78] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[79] Y. Lee and J. Park, "Centermask: Real-time anchor-free instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 13 906–13 915.

[80] Y. Guo, F. Chen, Q. Cheng, J. Wu, B. Wang, Y. Wu, and W. Zhao, "Fully convolutional one-stage circular object detector on medical images," in *2020 4th International Conference on Advances in Image Processing*, 2020, pp. 21–26.

[81] Y. Lee, J.-w. Hwang, S. Lee, Y. Bae, and J. Park, "An energy and gpu-computation efficient backbone network for real-time object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[82] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Z. Yifu, C. Wong, A. V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, and M. Jain, *ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation*, Nov. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.7347926

[83] P. Singh, G. Saha, and M. Sahidullah, "Deep scattering network for speech emotion recognition," in *2021 29th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 131–135.

[84] C. Pan, S. Chen, and A. Ortega, "Spatio-temporal graph scattering transform," *arXiv preprint arXiv:2012.03363*, 2020.

[85] S. Cheng, Y.-S. Ting, B. Ménard, and J. Bruna, "A new approach to observational cosmology using the scattering transform," *Monthly Notices of the Royal Astronomical Society*, vol. 499, no. 4, pp. 5902–5914, 2020.

[86] Á. B. Rodríguez, R. Balestriero, S. De Angelis, M. C. Benítez, L. Zuccarello, R. Baraniuk, J. M. Ibanez, and M. V. de Hoop, "Recurrent scattering network detects metastable behavior in polyphonic seismo-volcanic signals for volcano eruption forecasting," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–23, 2021.

[87] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1872–1886, 2013.

[88] E. Oyallon, E. Belilovsky, S. Zagoruyko, and M. Valko, "Compressing the input for cnns with the first-order scattering transform," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 301–316.

[89] S. Saponara and A. Elhanashi, "Impact of image resizing on deep learning detectors for training time and model performance," in *International Conference on Applications in Electronics Pervading Industry, Environment and Society*. Springer, 2021, pp. 10–17.

[90] H. Mao, S. Yao, T. Tang, B. Li, J. Yao, and Y. Wang, "Towards real-time object detection on embedded systems," *IEEE Transactions on Emerging Topics in Computing*, vol. 6, no. 3, pp. 417–431, 2016.

[91] G. Wang, Y. Chen, P. An, H. Hong, J. Hu, and T. Huang, "Uav-yolov8: a small-object-detection model based on improved yolov8 for uav aerial photography scenarios," *Sensors*, vol. 23, no. 16, p. 7190, 2023.

[92] L. Zhu, X. Wang, Z. Ke, W. Zhang, and R. W. Lau, "Biformer: Vision transformer with bi-level routing attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 10 323–10 333.

[93] F. C. Akyon, S. Onur Altinuc, and A. Temizel, "Slicing aided hyper inference and fine-tuning for small object detection," in *2022 IEEE International Conference on Image Processing (ICIP)*, 2022, pp. 966–970.

[94] K. M. Han and G. N. DeSouza, "Geolocation of multiple targets from airborne video without terrain data," *Journal of Intelligent & Robotic Systems*, vol. 62, no. 1, pp. 159–183, 2011.

[95] M. B. Shami, G. Kiss, T. A. Haakonsen, and F. Lindseth, "Geo-locating road objects using inverse haversine formula with nvidia driveworks," *arXiv preprint arXiv:2401.07582*, 2024.

[96] A. Milosavljević, D. Rančić, A. Dimitrijević, B. Predić, and V. Mihajlović, "A method for estimating surveillance video georeferences," *ISPRS international journal of geo-information*, vol. 6, no. 7, p. 211, 2017.

[97] K. Naus, M. W, P. Szymak, L. Gucma, and M. Gucma, "Assessment of ship position estimation accuracy based on radar navigation mark echoes identified in an electronic navigational chart," *Measurement*, vol. 169, p. 108630, 2020.

[98] C. E. Livingstone, M. Dragosevic, S. Chu, and I. Sikaneta, *Ship detection and measurement of ship motion by multi-aperture Synthetic Aperture Radar*. Defence Research and Development Canada, 2014.

[99] Y. Wei, Z. Zhang, B. Mu, Y. Li, Q. Wang, and R. Liu, "Geolocation accuracy evaluation of gf-4 geostationary high-resolution optical images over coastal zones and offshore areas," *Journal of Coastal Research*, vol. 102, no. SI, pp. 326–333, 2020.

[100] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[101] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.

[102] NVIDIA Corporation, "Nvidia tensorrt developer guide," https://docs.nvidia.com/deeplearning/tensorrt/developer-guide/index.html, 2024, accessed: 2024-05-13.

[103] S. Stepanenko and P. Yakimov, "Using high-performance deep learning platform to accelerate object detection," in *Proceedings of the International Conference on Information Technology and Nanotechnology, Samara, Russia*, 2019, pp. 26–29.

[104] O. R. developers, "Onnx runtime," https://onnxruntime.ai/, 2021, version: x.y.z.

[105] D. Heller, M. Rizk, R. Douguet, A. Baghdadi, and J.-P. Diguet, "Marine objects detection using deep learning on embedded edge devices," in *2022 IEEE International Workshop on Rapid System Prototyping (RSP)*. IEEE, 2022, pp. 1–7.

[106] R. Panero Martinez, I. Schiopu, B. Cornelis, and A. Munteanu, "Real-time instance segmentation of traffic videos for embedded devices," *Sensors*, vol. 21, no. 1, p. 275, 2021.

[107] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org, 2017.

[108] "Port information guide bremerhaven," http://www.hbh.bremen.de/sixcms/media.php/13/PORT-INFORMATION-GUIDE-Bremerhaven.pdf, Harbour Master Port of Bremerhaven.

[109] W. Kentaro, "labelme: Image Polygonal Annotation with Python," https://github.com/wkentaro/labelme, 2016.

[110] M. Riveiro, G. Pallotta, and M. Vespe, "Maritime anomaly detection: A review," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 5, p. e1266, 2018.

[111] F. A. C. de Oliveira, A. Niemi, A. García-Ortiz, and F. S. Torres, "Partial camera obstruction detection using single value image metrics and data augmentation," in *2022 6th International Conference on System Reliability and Safety (ICSRS)*. IEEE, 2022, pp. 292–299.

[112] H.-C. Burmeister, P. Grundmann, P. Hohnrath, and A. Ujkani, "Increasing maritime situational awareness by augmented reality solutions," Fraunhofer Center for Maritime Logistics and Services CML, White paper, 2021, available online: https://www.cml.fraunhofer.de/content/dam/cml/en/documents/Studien/Burmeister%20Grundmann%20Hohnrath%20Ujkani%20%282021%29_Increasing-Situational-Awareness-by-Augmented-Reality-Solutions_White%20Paper.pdf.

[113] *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org

[114] A. Sloss, D. Symes, and C. Wright, *ARM system developer's guide: designing and optimizing system software*. Elsevier, 2004.

[115] T. Guo, T. Zhang, E. Lim, M. Lopez-Benitez, F. Ma, and L. Yu, "A review of wavelet analysis and its applications: Challenges and opportunities," *IEEE Access*, vol. 10, pp. 58 869–58 903, 2022.

[116] F. Cotter, "Uses of complex wavelets in deep convolutional neural networks," Ph.D. dissertation, University of Cambridge, 2020.

[117] I. W. Selesnick, R. G. Baraniuk, and N. C. Kingsbury, "The dual-tree complex wavelet transform," *IEEE signal processing magazine*, vol. 22, no. 6, pp. 123–151, 2005.

[118] M. Andreux, T. Angles, G. Exarchakis, R. Leonarduzzi, G. Rochette, L. Thiry, J. Zarka, S. Mallat, J. Andén, E. Belilovsky *et al.*, "Kymatio: Scattering trans-

forms in python," *Journal of Machine Learning Research*, vol. 21, no. 60, pp. 1–6, 2020.

[119] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "Tph-yolov5: Improved yolov5 based on transformer prediction head for object detection on drone-captured scenarios," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 2778–2788.

[120] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[121] Y. Xie, M. Wang, X. Liu, and Y. Wu, "Integration of gis and moving objects in surveillance video," *ISPRS International Journal of Geo-Information*, vol. 6, no. 4, p. 94, 2017.

[122] Z. Shao, C. Li, D. Li, O. Altan, L. Zhang, and L. Ding, "An accurate matching method for projecting vector data into surveillance video to monitor and protect cultivated land," *ISPRS International Journal of Geo-Information*, vol. 9, no. 7, p. 448, 2020.

[123] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Computer Vision-ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part IV 8*. Springer, 2004, pp. 25–36.

[124] D. G. Bell, F. Kuehnel, C. Maxwell, R. Kim, K. Kasraie, T. Gaskins, P. Hogan, and J. Coughlan, "Nasa world wind: Opensource gis for mission operations," in *2007 IEEE Aerospace Conference*. IEEE, 2007, pp. 1–9.