



HAL
open science

Smart River : Towards Efficient Cooperative Autonomous Inland Navigation

Wided Hammedi

► **To cite this version:**

Wided Hammedi. Smart River : Towards Efficient Cooperative Autonomous Inland Navigation. Artificial Intelligence [cs.AI]. Université Bourgogne Franche-Comté, 2022. English. NNT : 2022UBFCK001 . tel-03614827

HAL Id: tel-03614827

<https://theses.hal.science/tel-03614827>

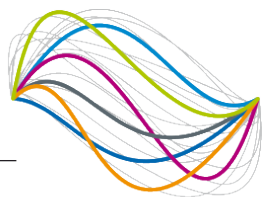
Submitted on 21 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UBFC

UNIVERSITÉ
BOURGOGNE FRANCHE-COMTÉ



DRIVE

Laboratoire DRIVE - EA 1859 -
Département de Recherche en Ingénierie
des Véhicules pour l'Environnement

THESE DE DOCTORAT DE L'ETABLISSEMENT UNIVERSITE BOURGOGNE FRANCHE-COMTE

PREPAREE A L'UNIVERSITE DE BOURGOGNE

Ecole doctorale n° 37

SCIENCES PHYSIQUES POUR L'INGENIEUR ET MICROTECHNIQUES

Doctorat en Informatique

Par

Wided HAMMEDI

**Smart River: Towards Efficient Cooperative Autonomous Inland
Navigation**

Thèse présentée et soutenue à **Nevers**, le **11 Janvier 2022**

Composition du Jury :

| | | |
|-------------------------|---|--------------------|
| Pietro MANZONI | Professeur, Université Polytechnique de Valence, Espagne | Président du jury |
| Toufik AHMED | Professeur, Bordeaux INP | Rapporteur |
| Thierry DELOT | Professeur, Université Polytechnique Hauts-de-France | Rapporteur |
| Franck GECHTER | Maître de conférence (HDR), Université de Technologie de Belfort Montbéliard (UTBM) | Examinateur |
| Charlotte LOGEAIS | Chargée de mission bateau autonome, VNF | Examinatrice |
| Yvon GOURHANT | Ingénieur de recherche, Orange Labs | Examinateur |
| Sidi-Mohammed SENOUCI | Professeur, Université de Bourgogne | Directeur de thèse |
| Philippe BRUNET | Maître de conférence, Université de Bourgogne | Encadrant |
| Metzli RAMIREZ-MARTINEZ | Enseignante chercheure, Université de Bourgogne | Invitée |

Titre : Rivière intelligente : Vers une navigation fluviale efficace, autonome et coopérative

Mots clés : Bateaux Autonomes, Internet des bateaux, Systèmes de transport intelligents coopératifs, écluses automatisées, connectivité, apprentissage machine

Résumé : Ces dernières années, le transport fluvial a fait l'objet d'une attention croissante de la part des pouvoirs publics de France et de nombreux pays européens. Cependant, ce mode transport manque de flexibilité, dispose d'une infrastructure vieillissante et les bateaux actuels ne sont pas adaptés à une augmentation des capacités de transport assurant la sécurité des navires et des marchandises ainsi que des temps de livraisons fiables et constants. Par conséquent, le transport fluvial doit passer par une rénovation organisationnelle et technique propre à son environnement particulier afin d'espérer concurrencer le transport terrestre. Pour répondre à ces exigences, nous proposons dans cette thèse de développer un écosystème fluvial intelligent s'articulant autour de trois axes principaux : (i) automatiser l'infrastructure fluviale existante, (ii) intégrer le bateau autonome à cette infrastructure, et (iii) promouvoir une navigation connectée et coopérative. Le premier axe se concentre sur la modernisation des infrastructures en proposant une méthode efficace de prise de décision pour l'automatisation des écluses existantes (Lock-ADM). L'algorithme Lock-ADM fonctionne en trois étapes. Tout d'abord, il calcule le nombre optimal d'écluses à automatiser pour un coût d'investissement fixé. Il mesure ensuite l'importance des écluses dans le réseau selon plusieurs critères. Enfin, il sélectionne les meilleures écluses à automatiser grâce à un algorithme génétique. Lock-ADM permet ainsi la planification annuelle des écluses à automa-

tiser, en commençant par les plus contraignantes pour le réseau actuel. Le deuxième axe s'intéresse au développement d'un système de perception de l'environnement pour bateaux autonomes. Il permet de délimiter les zones navigables où un bateau peut naviguer en toute sécurité et ainsi éviter tout obstacle sur sa route. Pour ce faire, nous avons construit le premier jeu de données (Images étiquetées) open-source et destiné au domaine fluvial: InlandAutoDetect. Nous avons étiqueté de manière exhaustive les différents objets constitutifs de la navigation fluvial. Ensuite, nous avons comparé les performances de neuf algorithmes d'apprentissage profond en termes de précision de détection et de temps de réponse vis-à-vis de nos étiquettes. Nous avons ainsi retenu l'algorithme Retinanet qui a montré les meilleures performances pour délimiter avec précision et en temps réel une zone de navigation qui soit sûre pour notre bateau autonome. Enfin, le troisième axe introduit C-IAShips, une architecture basée sur les technologies de Blockchain et de MEC (Mobile Edge Computing) à destination des bateaux autonomes coopératifs. L'architecture proposée garantit une faible latence et une communication efficace tout en protégeant la confidentialité des bateaux et la sécurité des données échangées. L'avantage principal des bateaux coopératifs est qu'ils permettent d'assurer un fonctionnement plus puissant et efficace du système global. Nous avons étudié, en particulier, la faisabilité de deux applications coopératives : la première pour l'ordonnancement du passage des bateaux au niveau des écluses et la seconde pour la détection de collisions.

Title: Smart River: Towards Efficient Cooperative Autonomous Inland Navigation

Keywords: Autonomous Ships, Internet of Ships, Cooperative Intelligent Transportation Systems, Automated locks, Connectivity, Machine Learning

Abstract: In recent years, inland waterway transport has witnessed increasing attention from France and many European countries. However, this mode of transport lacks flexibility, has an aging infrastructure and the current ships are not adapted to an increase in transport capacity ensuring the safety of vessels and goods as well as reliable and constant delivery times. Therefore, inland transport must go through an organizational and technical renovation specific to its particular environment in order to hope to compete with land transport. In this thesis, we propose developing a smart river ecosystem that focuses on three principal axes: (i) automatic inland infrastructure, (ii) autonomous inland ship, and (iii) promoting connected and cooperative navigation. The first axis focuses on efficiently automating the existing inland infrastructure using the Lock Automation Decision Making (Lock-ADM). The Lock-ADM algorithm works in three steps. First, it calculates the optimal number of locks to automate for a given investment cost. Then it measures the importance of the locks in the network according to several criteria. Finally, it selects the best locks to automate using a genetic algorithm. Lock-ADM allows the annual planning of the locks to be automated, starting with

the most constraining ones for the current network. The second axis focuses on the development of an environment perception system for autonomous ships. It allows to delimit the navigable zones where a ship can navigate safely and thus avoid any obstacle on its way. To do so, we have built the first open-source dataset (labeled images) for the river domain: InlandAutoDetect. We have exhaustively labeled the different objects that make up the river navigation. Then, we compared the performances of nine deep learning algorithms in terms of detection accuracy and response time. We selected the Retinanet algorithm which showed the best performance to delimit with accuracy and in real time a safe navigation zone for our autonomous ship. Finally, the third axis introduces C-IAShips, an architecture based on Blockchain and MEC (Mobile Edge Computing) technologies for cooperative autonomous ships. The proposed architecture guarantees low latency and efficient communication while protecting the confidentiality of the ships and the security of the exchanged data. The main advantage of cooperative ships is that they allow for a more powerful and efficient operation of the overall system. In particular, we have studied the feasibility of two cooperative applications: the first for scheduling the passage of ships at locks and the second for collision detection.

ACKNOWLEDGEMENT

Thomas Aquinas once said that "if the supreme purpose of a captain is to preserve his ship, he will keep it in port forever." Nevertheless, thanks to several people, my ship was able to navigate and return to port safely. Hence, this part is dedicated to expressing my gratitude to all those who have accompanied me throughout my journey.

First and foremost, I would like to express my sincere gratitude to my thesis supervisor Professor Sidi-Mohammed Senouci, for providing me the opportunity to join his team and accomplish this work. His deep experience and knowledge, continuous support, and patient guidance have greatly encouraged me through my thesis. Thanks to him, I learned the importance of asking the right questions that open paths worth exploring. It has been an honor to be his Ph.D. student. He made me discover how amazing research can be.

I extend my special appreciation and thanks to all the jury members, Toufik Ahmed, Franck Gechter, Pietro Manzoni, Thierry Delot, Yvon Gourhant and Charlotte Logeais, for participating in my Ph.D. defense.

I am also grateful to my co-supervisor, Dr. Philippe Brunet, for his advice, guidance, and valuable suggestions that helped me at various stages of my research. I want to thank Dr. Metzli Ramirez-Martinez and Dr. Bouziane Brick for their invaluable contributions. Even though they were not officially my supervisors, I am delighted and thankful for working with them.

I also would like to thank the University of Burgundy for funding my studies and giving me the chance to be a teaching assistant, a role that I sincerely enjoyed.

I appreciate many colleagues and friends in the DRIVE laboratory for every exciting discussion we have had, inside and outside the workplace. My deepest thanks to all my friends Lasma, Maissa, Marwa, Khouloud, Sarah, and Zeineb, who were by my side and brought me support and inspiration.

The people closest to my heart appear, of course, in the last paragraph. My deepest gratitude goes to my precious family, who tried to support me and encourage me in every possible way. I am entirely grateful to my dad, who always believed in my dreams and let me make my own decisions, including studying abroad. Special thanks to mom, who taught me the value of compassion and resilience and supported me in all my pursuits. I will never forget to thank my sisters Rahma, Imen, Islem, and little brother Yassine for their caring, help, and support throughout my struggles. I am also thankful for my life partner, the light and joy of my life, for his joyful humor, which gave me strength and made this arduous journey easier. Everything I have reached is because of your continuous support and love.

Wided HAMMEDI

CONTENTS

| | |
|--|-------------|
| Contents | v |
| List of Figures | vii |
| List of Tables | viii |
| List of Publications | ix |
| 1 Introduction | 1 |
| 1.1 Research Context and Motivation | 2 |
| 1.2 Research Opportunities and Challenges | 3 |
| 1.3 Research Methodology and Contributions Overview | 5 |
| 1.4 Thesis Structure | 7 |
| 2 Survey on Cooperative Autonomous Inland Ships | 10 |
| 2.1 Introduction | 11 |
| 2.2 Autonomous Inland Ships | 11 |
| 2.2.1 What is an Autonomous System ? | 12 |
| 2.2.2 Key Enabling Competencies of an Autonomous Vehicle | 15 |
| 2.2.3 Opportunities and Challenges Related to Autonomous Inland Ships | 22 |
| 2.3 Cooperative and Connected Inland Ships | 32 |
| 2.3.1 What is a Cooperative and Connected Vehicle? | 32 |
| 2.3.2 Underlying Technologies for Cooperative and Connected Autonomous Inland Ships | 36 |
| 2.3.3 Opportunities and Challenges for Connected and Cooperative In- land Ships | 40 |
| 2.4 Conclusion | 44 |
| I Automation | 45 |
| 3 Automating the Inland Infrastructure | 47 |
| 3.1 Introduction | 49 |
| 3.2 French Waterways Situation Analysis | 50 |
| 3.2.1 River Network | 50 |
| 3.2.2 Traffic Density | 50 |
| 3.2.3 Passage Scheduling | 51 |
| 3.2.4 Opening Time | 52 |
| 3.2.5 Does Automation Have an Impact on Traffic Density? | 52 |
| 3.3 Locks Automation Related Works | 53 |
| 3.4 Problem Formulation and Proposed Solution | 54 |
| 3.5 Lock-ADM: Optimal Number of Locks to Automate and Their Placement . | 55 |
| 3.5.1 Lock-ADM - Stage 1: Calculate the Optimal Number of Locks to Automate | 55 |

| | | |
|-----------|--|------------|
| 3.5.2 | Lock-ADM - Stage 2: Order Locks According to their Importance Score | 57 |
| 3.5.3 | Lock-ADM - Stage3: Select Best Locks to Automate by Studying their Impact on the Network | 60 |
| 3.6 | Implementation and Results | 64 |
| 3.6.1 | Locks Importance Score Calculation (Stage 2) | 64 |
| 3.6.2 | Metaheuristics Comparison (Stage 3) | 67 |
| 3.6.3 | Comparison with Other Algorithms | 67 |
| 3.6.4 | Impact of the Population’s Initialization | 68 |
| 3.7 | Conclusion | 69 |
| 4 | Automating the Inland Ships | 71 |
| 4.1 | Introduction | 73 |
| 4.2 | Related Work | 74 |
| 4.2.1 | Real-time Traffic Object Detection for Autonomous Vehicles | 75 |
| 4.2.2 | Real-time Drivable Area Detection for Autonomous Vehicles | 77 |
| 4.3 | InlandAutoDetect Dataset Construction | 78 |
| 4.3.1 | Data Acquisition | 78 |
| 4.3.2 | Data Model | 80 |
| 4.3.3 | Ground Truth Labeling Process | 81 |
| 4.4 | Object Detection Implementation Details | 82 |
| 4.4.1 | Hardware and Software Configuration Used | 82 |
| 4.4.2 | Data Preprocessing and Preparation | 82 |
| 4.4.3 | Architectural Configuration | 84 |
| 4.5 | Inland Object Detection Performance Evaluation | 90 |
| 4.5.1 | Boundary Conditions | 90 |
| 4.5.2 | Comparison Criteria | 91 |
| 4.5.3 | Experimental Results Analysis | 92 |
| 4.6 | Navigable Area Detection Performance Evaluation | 97 |
| 4.6.1 | Proposed System | 97 |
| 4.6.2 | Experimental Results Analysis | 98 |
| 4.6.3 | Discussions | 99 |
| 4.7 | Conclusion | 100 |
| II | Cooperation | 102 |
| 5 | Cooperative Autonomous Inland Navigation | 104 |
| 5.1 | Introduction | 106 |
| 5.2 | Related Work | 107 |
| 5.2.1 | Mobile Cloud Computing (MCC) | 108 |
| 5.2.2 | Fog Computing (FogC) | 108 |
| 5.2.3 | Mobile Edge Computing (MEC) | 108 |
| 5.3 | Proposed Architecture C-IAShips | 110 |
| 5.3.1 | C-IAShips Architecture Requirements | 110 |
| 5.3.2 | C-IAShips Architecture Layers | 111 |
| 5.3.3 | Blockchain Solution for Security Enhancement | 112 |

| | | |
|----------|--|------------|
| 5.4 | C-IAShips Architecture Advantages | 113 |
| 5.4.1 | C-IAShips Architecture Strengths | 113 |
| 5.4.2 | Cooperative Applications | 114 |
| 5.5 | Security Analysis | 115 |
| 5.5.1 | C-IAShips Architecture Vulnerabilities | 115 |
| 5.5.2 | Ships Privacy Protection | 116 |
| 5.5.3 | Security Attacks Detection | 116 |
| 5.6 | Conclusion | 118 |
| 6 | Cooperative Collision Detection | 120 |
| 6.1 | Introduction | 122 |
| 6.2 | Related Work | 123 |
| 6.3 | Problem Formulation and Proposed Solution | 125 |
| 6.4 | Federated Learning-based Collision Detection System | 126 |
| 6.4.1 | Main Modules Description | 126 |
| 6.4.2 | Cooperative Collision Detection System Training | 129 |
| 6.4.3 | Cooperative Collision Detection System Deployment | 130 |
| 6.5 | Performance Evaluation | 130 |
| 6.5.1 | Simulation Scenarios | 131 |
| 6.5.2 | Mobility Prediction Module Analysis | 132 |
| 6.5.3 | Federated Learning Benefits | 134 |
| 6.5.4 | Cooperative Collision Detection Efficiency | 138 |
| 6.6 | Conclusion | 140 |
| 7 | Cooperative Lock Scheduling | 141 |
| 7.1 | Introduction | 143 |
| 7.2 | Lock Scheduling Related Works | 144 |
| 7.3 | Problem Formulation | 145 |
| 7.3.1 | Proposed Solution | 146 |
| 7.3.2 | Problem Parameters and Assumptions | 147 |
| 7.4 | Dynamic Lock Scheduling (Lock-DS) Description | 148 |
| 7.4.1 | Lock-DS - Stage 1: Collect Ship's Characteristics | 148 |
| 7.4.2 | Lock-DS - Stage 2: Estimate Ship's arrival time and lockage duration | 149 |
| 7.4.3 | Lock-DS - Stage 3: Schedule Ships | 149 |
| 7.4.4 | Lock-DS - Stage 4: Optimize Ships' Speed | 154 |
| 7.5 | Performance Evaluation | 155 |
| 7.5.1 | Performance Evaluation Metrics | 155 |
| 7.5.2 | Numerical Experiments | 155 |
| 7.6 | Conclusion | 156 |
| 8 | Conclusion | 159 |
| 8.1 | Reminder of Goals and Challenges | 160 |
| 8.2 | Summary of the Contributions | 160 |
| 8.3 | Future Research Directions | 162 |
| | References | 163 |

LIST OF FIGURES

| | | |
|------|---|----|
| 1.1 | The envisioned smart river | 4 |
| 1.2 | Dissertation outline diagram | 7 |
| 2.1 | Applications of autonomous systems. | 12 |
| 2.2 | Standard components in an autonomous driving | 15 |
| 2.3 | Perception main steps. | 16 |
| 2.4 | Examples of matched scenes from the Mapillary dataset | 20 |
| 2.5 | Some typical scenes of inland navigation | 24 |
| 2.6 | H2H main components | 27 |
| 2.7 | Standard components in a connected autonomous driving | 33 |
| 2.8 | Journey to Cooperative Automated Vehicles technology. | 34 |
| 2.9 | Some of C-CAVs use case applications | 35 |
| 2.10 | A typical IoS structure (adapted from [117]) | 37 |
| 2.11 | Overall architecture of IoS | 38 |
| 2.12 | Ship-2-Everything (S2X) Communications | 39 |
| 3.1 | The actual French Navigable Waterways map. | 51 |
| 3.2 | Different traffic densities for different locks | 51 |
| 3.3 | Illustration of six different locks before and after the automation | 52 |
| 3.4 | Lock-ADM workflow | 55 |
| 3.5 | General flow chart of the used algorithms. | 61 |
| 3.6 | Solutions representation, where $L_1, L_1', L_2, \dots, L_{10}'$ are the locks. | 63 |
| 3.7 | Crossover explanation | 64 |
| 3.8 | Centrality measures | 65 |
| 3.9 | Traffic density weight | 66 |
| 3.10 | LockADM vs. other test algorithms for different number of nodes | 68 |
| 3.11 | Convergence history | 69 |
| 4.1 | Example image of a occlusion. | 79 |
| 4.2 | A sample from InlandAutoDetect dataset with multiple challenge types | 80 |
| 4.3 | An example of an annotated image. | 82 |
| 4.4 | Exmample of two anchor boxes | 84 |
| 4.5 | Shematic diagram of the FasterRCNN model | 85 |
| 4.6 | Shematic diagram of the SSD model | 86 |
| 4.7 | Shematic diagram of the YOLO model architecture [236] | 86 |
| 4.8 | Shematic diagram of the Retinanet model | 87 |
| 4.9 | VGG architecture. | 88 |
| 4.10 | A regular block (left) and a residual block (right). | 89 |
| 4.11 | Fine tuning steps. | 90 |
| 4.12 | Intersection over Union calculation. | 92 |
| 4.13 | A sample of predicted images generated by YOLO v3 | 92 |
| 4.14 | A sample of predicted images generated by Tiny-YOLO v3 | 93 |
| 4.15 | A sample of predicted images generated by YOLO v2 | 93 |
| 4.16 | A sample of predicted images generated by Tiny-YOLO v2 | 93 |
| 4.17 | A sample of predicted images generated by Faster-RCNN | 93 |

| | | |
|------|---|-----|
| 4.18 | A sample of predicted images generated by SSD | 94 |
| 4.19 | A sample of predicted images generated by Retinanet | 94 |
| 4.20 | Accuracy rates under different quality distortions | 96 |
| 4.21 | Autonomous navigable area detection system | 97 |
| 4.22 | A sample from the testing set with complex conditions | 98 |
| 4.23 | Some false positives and false negatives navigable area detection. | 99 |
| | | |
| 5.1 | The Envisioned C-IAShips architecture | 110 |
| 5.2 | Data stored in blocks of the blockchain. | 113 |
| 5.3 | Architecture malicious actors | 117 |
| | | |
| 6.1 | The collision avoidance process (adapted from [292]). | 123 |
| 6.2 | Cooperative collision detection system model | 127 |
| 6.3 | A sample of predicted images generated by obstacle detection module | 128 |
| 6.4 | Collision detection: training phase steps. | 129 |
| 6.5 | Boxplots of AvMSE value | 133 |
| 6.6 | Training Phase Convergence | 136 |
| 6.7 | MSE comparison | 137 |
| 6.8 | Percentage of collisions avoided. | 139 |
| 6.9 | Danger zones are surrounded by red circles | 140 |
| | | |
| 7.1 | Example of locks | 143 |
| 7.2 | Different types of locks | 146 |
| 7.3 | Lock-DS system model | 146 |
| 7.4 | Lock-DS process | 148 |
| 7.5 | Illustration of the state representation | 152 |
| 7.6 | RL general architecture | 153 |
| 7.7 | The fuel consumption (in tonnes) under five different schedulers | 157 |

LIST OF TABLES

| | | |
|-----|--|-----|
| 1.1 | Research Methodology to develop a cooperative connected autonomous inland navigation | 6 |
| 2.1 | Vehicle automation levels as described by SAE and ISO. | 13 |
| 2.2 | Levels of autonomy for an USV from [63] | 23 |
| 2.3 | Scope of USV types | 23 |
| 2.4 | A summary of path planning algorithms for USV | 30 |
| 2.5 | A summary of control methods for USV | 31 |
| 2.6 | Comparison of terrestrial communication systems for autonomous ships | 36 |
| 3.1 | Problem parameters | 56 |
| 3.2 | List of important symbols | 58 |
| 3.3 | Correlation coefficients for the centrality measures | 66 |
| 3.4 | Comparing metaheuristics | 67 |
| 3.5 | Algorithm parameters for testing | 68 |
| 3.6 | Comparison of the two algorithms Vs. the All-Automated | 69 |
| 4.1 | Popular maritime datasets | 77 |
| 4.2 | Main characteristics of InlandAutoDetect dataset | 80 |
| 4.3 | The detailed information of the InlandAutoDetect dataset before and after data augmentation. | 83 |
| 4.4 | Detection metrics in terms of mAP and AP. | 94 |
| 4.5 | FPS for the different detection models. | 95 |
| 4.6 | Different Retinanet versions performances in terms of WmAP and FPS. | 95 |
| 5.1 | Comparison of features of Edge paradigms [262]. | 109 |
| 6.1 | Collision avoidance methods' categories based on communication and cooperation levels | 124 |
| 6.2 | Simulation parameters | 132 |
| 6.3 | Training time for the 3 different learning techniques | 135 |
| 6.4 | Collision detection results on busy intersections areas | 140 |
| 7.1 | Overview of problem variants discussed in this chapter. | 144 |
| 7.2 | Lock-DS Parameters | 147 |
| 7.3 | Schedulers waiting time comparison | 156 |

LIST OF PUBLICATIONS

Interational journal papers

- Wided Hammedi, Sidi Mohammed Senouci, Metzli Ramirez-Martinez, and Philippe Brunet, "On Reliable Awareness System for Autonomous River Vessels", IEEE Transactions on Intelligent Transportation Systems, minor revision
- Wided Hammedi, Sidi Mohammed Senouci, Philippe Brunet, and Metzli Ramirez-Martinez, "Two-Level Optimization to Reduce Waiting Time at Locks in Inland Waterway Transportation", ACM Transactions on Intelligent Systems and Technology, accept with minor revision
- Wided Hammedi, Bouziane Brik, and Sidi Mohammed Senouci, "Towards Optimal MEC-based Collision Avoidance System for Cooperative Inland Vessels: A Federated Deep Learning Approach", IEEE Transactions on Intelligent Transportation Systems, major revision

Interational conference papers

- Wided Hammedi, Metzli Ramirez-Martinez, Philippe Brunet, Sidi Mohammed Senouci and Mohammed Ayoub Messous, "Deep Learning-Based Real-Time Object Detection in Inland Navigation," 2019 IEEE Global Communications Conference (GLOBECOM), 2019, pp. 1-6, doi: 10.1109/GLOBECOM38437.2019.9013931.
- Wided Hammedi, Bouziane Brik, and Sidi Mohammed Senouci, "Federated Deep Learning Based Framework to Avoid Collision Between Inland Ships", IEEE International Conference on Communications, under review

Poster/Talks

- Wided Hammedi, Sidi Mohammed Senouci, Philippe Brunet, and Metzli Ramirez-Martinez, "Towards Safe Autonomous Inland Navigation", FUTURMOB'21

LIST OF ABBREVIATIONS

Below, the list of abbreviations that has been used throughout this thesis, can be found. This list is made per chapter to show where each abbreviation appears first. No new entry will be made if a certain abbreviation returns in a later chapter.

Chapter 1

| | |
|-----------|--|
| IWT | Inland water transportation |
| tkm | tonne-kilometer |
| UE | European Union |
| ITS | Intelligent Transportation System |
| ICT | Information and Communication Technology |
| AV | Autonomous Vessel |
| GPS | Global Positioning System |
| CLS | Cooperative Lock Scheduling |
| CCD | Cooperative Collision Detection |
| MEC | Mobile Edge Computing |
| Lock-ADM | Lock Automation Decision Making |
| GA | Genetic Algorithm |
| C-IAShops | Cooperative Inland Autonomous Ships |

Chapter 2

| | |
|-------|--|
| IoT | Internet of Things |
| CAVs | Connected and Autonomous Vehicles |
| IoAV | Internet of Autonomous Vehicles |
| IoV | Internet of Vehicles |
| IoS | Internet of Ships |
| IoIS | Internet of Inland Ships |
| C-AIS | Cooperative Autonomous Inland Ships |
| ASs | Autonomous systems |
| ALV | Autonomous Land Vehicle |
| ADAS | Advanced Driving Assistance System |
| ADS | Automated Driving System |
| SAE | Society of Automotive Engineers |
| ISO | International Organization for Standardization |
| GPS | Global Positioning System |
| NHTSA | National Highway Traffic Safety Administration |
| ICP | Iterative Closest Point |
| DNN | Deep Neural Network |
| HOG | Histogram of Oriented Gradients |
| CNN | Convolutional Neural Network |
| HD | High Definition |

| | |
|--------------|---|
| VRP | Vehicle Routing Problem |
| MPC | Model Predictive Control |
| PID | Proportional-Integral-Differential |
| SVR | Support Vector Regression |
| RL | Reinforcement Learning |
| UAV | Unmanned Aerial Vehicles |
| AUV | Autonomous Underwater Vehicles |
| USV | Unmanned Surface Vehicles |
| LoA | Level of Autonomy |
| AI | Artificial Intelligence |
| SCC | Shore Control Centre |
| I-USV | Inland Unmanned Surface Ships |
| SOLAS | Safety of Life at Sea |
| COLREGS | Convention on the International Regulations for Preventing Collisions at Sea |
| IMO | International Maritime Organization |
| CCNR | Central Commission for Navigation on the Rhine |
| CESNI | European Committee for drawing up Standards in the field of Inland Navigation |
| UNECE | United Nations Economic Commission for Europe |
| I-SCC | Inland Shore Control Center |
| H2H | Hull-To-Hull |
| RIS | River Information System |
| AIS | Automatic Identification System |
| SAR | Synthetic Aperture Radar |
| RPN | Region Proposal Network |
| SVM | Support Vector Machine |
| GAN | Generative Adversarial Nets |
| GHz | Gigahertz |
| mmW | millimeter wave |
| LTE | Long Term Evolution |
| GNSS | Global Navigation Satellite System |
| IMU | Inertial Measurement Unit |
| RGB | Red, Green, Blue |
| ROS | Robotics Operating System |
| PLC | Programmable Logic Controller |
| NN | Neural Network |
| R-CNN | Region Convolutional Neural Network |
| Faster R-CNN | Faster Region Convolutional Neural Network |
| SSD | Single Shot MultiBox Detector |
| YOLO | You Only Look Once |
| AMS | Advanced Metropolitan Solution |
| MIT | Massachusetts Institute of Technology |
| SLAM | Simultaneous Localization and Mapping |
| APF | Artificial Potential Field |
| ACO | Ant Colony Optimization |

| | |
|---------|---|
| FPA | Flower Pollination Algorithm |
| RRT | Rapidly-exploring Random Trees algorithm |
| FSM | Finite State Machine |
| LoS | Line-of-Sight |
| MSDF | Multi-Sensor Data Fusion |
| CV | Connected Vehicle |
| V2X | Vehicle-to-Everything |
| V2V | Vehicle-to-Vehicle |
| V2I | Vehicle-to-Infrastructure |
| V2P | Vehicle-to-Pedestrians |
| V2C | Vehicle-to-Cloud |
| C-CAV | Cooperative Connected Autonomous Vehicles |
| ITS | Intelligent Transportation System |
| C-CAISs | Cooperative and Connected Autonomous Inland Ships |
| S2X | Ship-to-Everything |
| 3GPP | 3rd Generation Partnership Project |
| S2Sh | Ship-2-Shore |
| S2S | Ship-2-Ship |
| S2C | Ship-2-Cloud |
| SANET | Sea Ad hoc NETWORK |
| S2I | Ship-2-Infrastructure |
| S2N | Ship-2-Network |
| RFID | Radio Frequency Identification |
| ISMIS | Inland Shipping Management Information System |
| VTS | Vessel Traffic Services |
| ITT | Inter Terminal Transport |
| AGV | Automated Guided Vehicle |
| VRPP | Vessel Rotation Planning Problem |
| VC | Vessel Controller |
| IC | Intersection Controller |
| CRI | Collision Risk Index |
| OS | Own Ship |
| TS | Target Ship |

Chapter 3

| | |
|---------|--|
| VNF | Voies Navigables de France |
| CSV | Comma-Separated Values |
| FCFS | First Come First Served |
| NLP | Non Linear Programming |
| NP-hard | Nondeterministic polynomial-time- hard |
| PSO | Particle Swarm Optimization |
| deg | degree |
| DC | Degree Centrality |

| | |
|-----|------------------------------------|
| BWC | Betweenness Centrality |
| CC | Closeness Centrality |
| WBc | Random Walk Betweenness Centrality |
| Ec | Eigenvector centrality |
| ABC | Artificial Bee Colony algorithm |
| SA | Simulated Annealing |
| ATT | Average Travel Time |

Chapter 4

| | |
|---------|------------------------------------|
| DPM | Deformable Part-based Model |
| SCNN | Spatial CNN |
| GDPR | General Data Protection Regulation |
| GB | Gigabyte |
| GPU | Graphics Processing Unit |
| cuDNN | CUDA Deep Neural Network library |
| IoU | Intersection over Union |
| FPN | Feature Pyramid Networks |
| VGG | Visual Geometry Group |
| ResNets | Residual Networks |
| FPS | Frame per Second |
| AP | Average precision |
| mAP | Mean Average Precision |
| WmAP | Weighted mAP |

Chapter 5

| | |
|-------|---|
| C-ITS | Cooperative Intelligent Transport Systems |
| FL | Federated Learning |
| MCC | Mobile Cloud Computing |
| MEC | Mobile Edge Computing |
| FogC | Fog computing |
| RSUs | Roadside Units |
| PoW | Proof-of-Work |
| CRP | Cooperative Route Planning |

Chapter 6

| | |
|------|------------------------------|
| SUMO | Simulation of Urban MObility |
|------|------------------------------|

| | |
|---------|--|
| AIS | automatic identification system |
| ANI ESS | Automatic Navigational Intention Exchange Support System using AIS |
| RNN | Recurrent Neural Network |
| LSTM | Long Short-Term Memory |
| BiLSTM | Bidirectional LSTM |
| FedAvg | Federated Averaging |
| MSE | Mean squared error |
| HPbi | Position data history |
| CL | Centralized learning |
| DL | Distributed learning |

Chapter 7

| | |
|---------|--|
| MOBF | Multi-order best fit |
| IAD | Inquired Arrival Deadline |
| UPMSP | Unrelated Parallel Machines Scheduling Problem |
| Lock-DS | Dynamic Lock Scheduling |
| RL | Reinforcement learning |
| MDP | Markov's Decision Process |

INTRODUCTION

| | | |
|-----|---|---|
| 1.1 | Research Context and Motivation | 2 |
| 1.2 | Research Opportunities and Challenges | 3 |
| 1.3 | Research Methodology and Contributions Overview | 5 |
| 1.4 | Thesis Structure | 7 |

1.1 Research Context and Motivation

The effects of global climate change cannot be denied any longer. These effects require significant modifications in our behavior, particularly in the field of freight transport. Excessive use of land transport has many drawbacks such as pollution, road congestion, and road deterioration. In contrast, thanks to its advantages, Inland water transportation (IWT) may be a great alternative to other transport modes. The IWT, the oldest and most growing sector, progressing from primitive ships to highly automated ones [1], played a critical role before the development of modern land transport. The Commission of the Transportation Ministries of the European Community Member Countries [2] illustrated the IWT's advantages, as follows:

- **Cost effectiveness:** According to German statistical data, 1 tonne-kilometer (tkm) transportation costs €12.15 by roads, €6.35 by railways, and only €1.95 by inland shipping [1]. In addition, the IWT can carry much larger quantities of heavy goods. For instance, a single Freycinet barge, which can sail on the tiny French canals, with a loading capacity of 250 tonnes, can carry around 15 trucks or almost 4 trains [2].
- **Energy consumption efficiency:** The towed group achieves an average of 127.5 tkm per liter of fuel, whereas railway traffic achieves 76 tkm per liter and road traffic achieves only 23 tkm per liter [2].
- **Navigational safety:** The risks of accidents causing significant damages or injury are rare in inland waterway traffic. This benefit is significant when transporting dangerous cargo. As a result, some industrial sectors, such as construction, mining, forestry, metallurgy, chemical, and oil, may reach the market only via inland waterway transport since other modes are unacceptable.
- **Environmentally friendliest:** According to various analyses, the inland waterway traffic has a lower degree of pollution of air than the other modes. For instance, it has been demonstrated that the CO₂ emissions of a barge carrying 1500 tons of cargo are three times less than those of the fleet of trucks required in order to move the same load [2]. In addition, it reduced the noise, olfactory and visual nuisances.
- **High reliability:** Compared to other modes of transport, which are often confronted with congestion and capacity problems, inland waterway transport is characterized by its reliability in transit and delivery times.

Seeing these benefits, the European Union (UE) has recently decided to progress this sector and strengthen the competitive position of inland waterways in the transport system by increasing its share by 25% by 2030 and by 50% by 2050 [2]. With 8,500 km of waterways, France has the most extended network of waterways in Europe [3]. Thanks to its localization (at the crossroads of Europe) and its multiple ports, the inland network offers potential growth for river freight transportation in the country. However, the French government should also consider the actual waterways situation and the IWT's specific characteristics. We classified these obstacles into three categories, as follows:

- **Seasonal character:** The navigation in some inland waterways areas may be affected by climatic conditions: water may freeze during winter, or its level may drop dramatically during summer. However, even rail and road transport may be disrupted during extreme weather conditions like periods of flooding and snowfall. Therefore, the real challenge is making distribution transport chains organized, including road or rail transport, and building dynamic fleet management to ensure optimal trajectories while avoiding the blocked areas.
- **Longer voyage time:** Inland navigation ships have speeds in the range between 10 and 20 km/h, which is much lower than the speed of trains or road vehicles. In addition, ships have to cross many locks in one voyage generating an enormous waiting time: for example, on the path between Lyon and Paris, they have to cross 220 locks; each crossing takes 20 minutes on average. Therefore, it usually takes longer to carry goods from one place to another through this form of transport in its current format. Moreover, ships can not operate continually for 24 hours to meet the delivery deadlines due to the lack of efficient navigation systems.
- **Isolated and out-dated technology system:** In Europe, where the canal epoch began at the end of the 17th century, France took the lead, further integrating its national waterway system by forging the missing channels connections [4]. However, the waterways have barely been maintained since, and some waterways were abandoned for navigation, mainly in 1925-1955.

To sum up, although picturesque promises, spanned by inland navigation, developing this sector yields new challenges compared to the sea/land sector. Therefore, adequate strategies and solutions must be deployed to benefit from this old transport system's substantial advantages while considering its specific characteristics and challenges. Additionally, with the expected growth of inland shipping and the increased number of inland ships required to convey the freight, the ultimate goal is to design an intelligent shipping system to make inland shipping safer and more efficient. The concept of **Smart River** may be the best way out of this impasse. Smart River means integrating information and communication technologies to manage inland components: infrastructure, ships, sensors, and transport authorities. A smart river aims to improve navigation quality by introducing Intelligent Transportation System (ITS) techniques to the inland sector. This unprecedented growth stimulates the emergence of many applications and services that aim to enhance driving conditions, reduce voyage time, bring efficient traffic management, and increase river safety by collision detection and avoidance. Digital, agile, and flexible rivers manage costs, support the efficient use of all resources, and efficiently adapt to changing demands. Hence, we seek to integrate smart river technology to meet this expected growth in inland shipping during our thesis.

1.2 Research Opportunities and Challenges

The smart river concept integrates Information and Communication Technology (ICT) into the various connected river components to optimize the efficiency of river operations and

services, as illustrated in Figure 1.1. Smart river technology allows inland transport authorities to interact directly with ships and inland infrastructure to monitor what is happening in the river. However, such advancement is not yet possible regarding the actual situation of rivers. In this sense, we formulate our research problem as follows: " Given the current situation of the river marked by restricted maneuverability, lack of efficient navigation systems, and inefficient communication among ships and given the inland specific characteristics, how to determine the best strategies for the ships and the system to reach reliable smart navigation?"

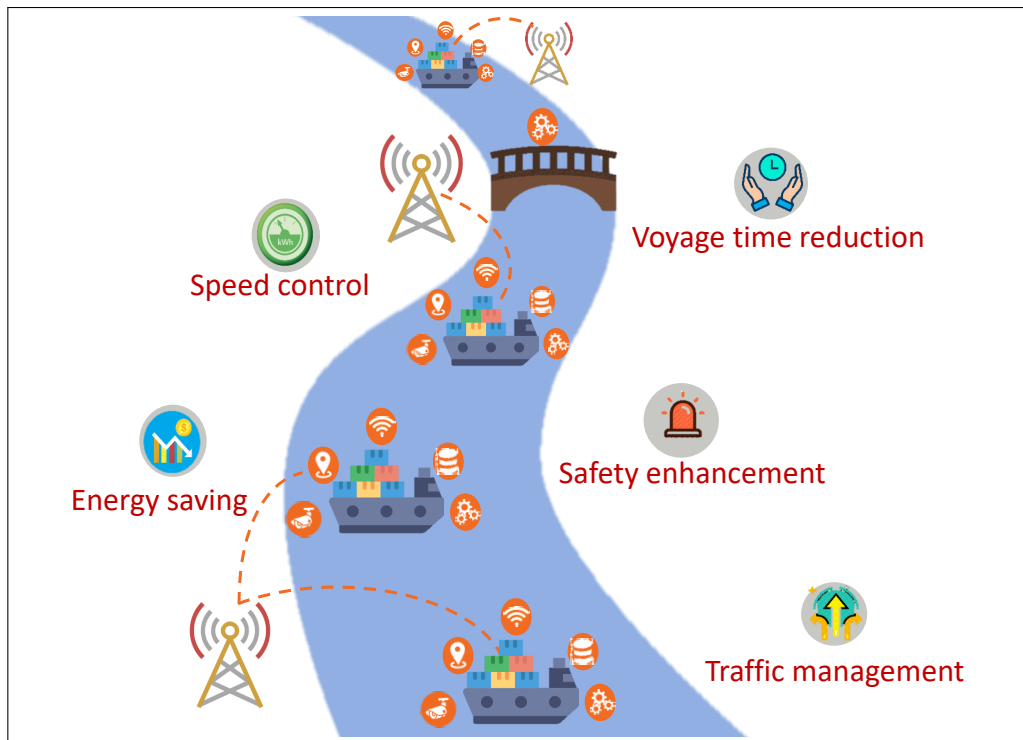


Figure 1.1: The envisioned smart river

One of the most critical steps to ensure such a development is to address the **infrastructure limitations** without **huge investments**. As mentioned earlier, despite France's most extended waterways network in Europe, with no missing channels connections since the 17th century, it abandoned some of them for navigation last years. Hence, one of the critical tasks of the inland transport authorities is to introduce technological solutions to **effectively modernize and automate** the existing river infrastructure, considering the growing traffic size. By eliminating the need for a lock keeper, ships will be able to operate continually for 24 hours and be entirely adapted to observing the delivery deadlines.

A river's smartness also implies considering **autonomous ships** in its pillars to develop more flexible and reliable solutions. An autonomous ship can operate itself and perform necessary functions without any human intervention through the ability to sense its surroundings. Higher levels of autonomy have the potential to increase safety by reducing risky driver behaviors. Implementing such ships is becoming a reality thanks to the increasing development and evolution of embedded technologies, integrated automotive sensors, and **machine visions** for the reliable external **perception** of the surroundings.

However, individual autonomous ships may be inefficient regarding the expected growth of

inland shipping; the increased number of inland ships required to convey the freight would lead to new fleet management challenges. For instance, an autonomous ship can sense its environment but can not comprehend, without fail, the other ships' intentions. Additionally, each ship acting in its way, a fleet of ships may cause conflicting situations when paths cross simultaneously. Therefore, **cooperative autonomous ships** can achieve greater efficiency, operational capability, and robust applications. Consequently, the success of a smart river relies on the cooperation between its different components, i.e., autonomous ships and automated locks, by integrating **intra-communication** capabilities.

To sum up, the main goal of this thesis is concentrated around inland navigation development employing the smart river concept. More specifically, the objective is to provide adequate strategies including, techniques and methods, to allow a reliable cooperative connected autonomous inland navigation. The basic idea of these strategies is to identify the specific requirement to make such a solution become a reality.

1.3 Research Methodology and Contributions Overview

In the context presented in the previous section, we aim to benefit from the considerable development witnessed by the automotive area and available communication technologies to build an efficient smart river ecosystem and address the earlier mentioned issues. For this purpose, the contributions of this thesis make reference to four main requirement classes, as listed in Table 1.1:

- ❖ **Infrastructure-related requirements:** River locks are the oldest waterway infrastructure used to elevate safely and lower ships between water level differences and are considered vital inland navigation nodes. However, they generate an enormous waiting time due to their lack of flexibility and the need for lock-keepers to perform the passage through them. In the entire river network, all commercial ships must announce their entry into the network. They must book the lockage service within fixed timetables by calling the customer-user relations department. Lock automation may be an efficient solution to reduce waiting time, and therefore the shipment costs. Therefore, we consider solving this problem using the Lock-ADM, a three-stage algorithm for determining the number and locks' position to automate, respecting the budgetary and time constraints.
- ❖ **Autonomous ship-related requirement:** The main techniques for autonomous ships include solving tasks like self-localization, parsing the driving road, and understanding objects, enabling the system to reason and act. Therefore, they need reliable and robust perception systems so that the automated driving system can decide upon more cautious maneuvers or even bring the ship to a stop if a near obstacle is perceived. In this work, we develop the perception part with two main methods. On the one hand, using cameras to develop a contextual understanding of the environment, autonomous ships can apply computer vision techniques to localize obstacles, detect traffic signs, and categorize data according to their semantic meaning. To do this, we start with constructing a dataset, the InlandAutoDetect dataset, adapted to

the inland domain. Then, we evaluate the accuracy and performance of nine state-of-the-art object detection models in the inland environment to select the adequate configuration for inland navigation. On the other hand, using GPS data, autonomous ships can provide timely and accurate positioning predictions. We evaluate the accuracy and performance of four state-of-the-art deep learning models architecture to select the most efficient and robust one for mobility prediction in the inland environment to enhance the environmental awareness and, thus, the prediction of potential collisions.

- ❖ **Fleet management-related requirement:** Increasing the number of ships will lead to good fleet management requirements. This study supports two fundamental applications: the Cooperative Lock Scheduling (CLS) application to improve the scheduling method used to cross locks and thus reduce waiting time and fuel consumption, and the Cooperative Collision Detection (CCD) application to ensure safe navigation. To this end, cooperative ships send their characteristics to a central node. Then, this central node can make decisions concerning detected collisions or schedules according to the data received while keeping an eye on the state of the river.

- ❖ **Connectivity-related requirement:** Connectivity is the enabling technology for enhancing the efficiency of cooperative ships. However, in such a cooperative context, some essential connectivity requirements arise. **Low latency** connection is considered to be the most important performance metric as some applications require very stringent requirements. Since the proposed ecosystem needs numerous data collected, analyzed, and used to gain actionable insights, ships' **privacy** and shared data **security** need to be protected. Therefore, in this work, we innovatively introduce the C-IAShips architecture, a three-layer blockchain-based federated learning architecture for cooperative inland autonomous ships. The C-IAShips architecture achieves high communication efficiency and protects ships' privacy from being leaked while realizing the different proposed applications. Delay-sensitive applications are performed on Mobile Edge Computing (MEC) servers, while delay-tolerant computing tasks applications are performed on a Cloud server.

Table 1.1: Research Methodology to develop a cooperative connected autonomous inland navigation

| Requirement | Objective | Proposed Solution | Discussed in |
|---------------------------------------|--|---------------------------------------|--------------|
| Infrastructure-related requirements | Automate the existing locks without huge investments | Lock-ADM algorithm | Chapter 3 |
| Autonomous ship-related requirements | Ensure Reliable environment perception | Vision-based environment perception | Chapter 4 |
| | | GPS-based global localization | Chapter 5 |
| Fleet management-related requirements | Make river waterways more efficient using cooperative applications | Cooperative Collision Detection (CCD) | Chapter 6 |
| | | Cooperative Lock Scheduling (CLS) | Chapter 7 |
| Connectivity-related requirements | Low latency | Mobile Edge Computing (MEC) | Chapter 5 |
| | Ships' privacy protection | Federated Learning & Blockchain | Chapter 5 |
| | Data security protection | Blockchain | Chapter 5 |

1.4 Thesis Structure

We like to remind the reader that all the objectives mentioned above were implemented, not necessarily in the described order. However, for comprehension reasons, we will present them following the organization as given in Fig 1.2.

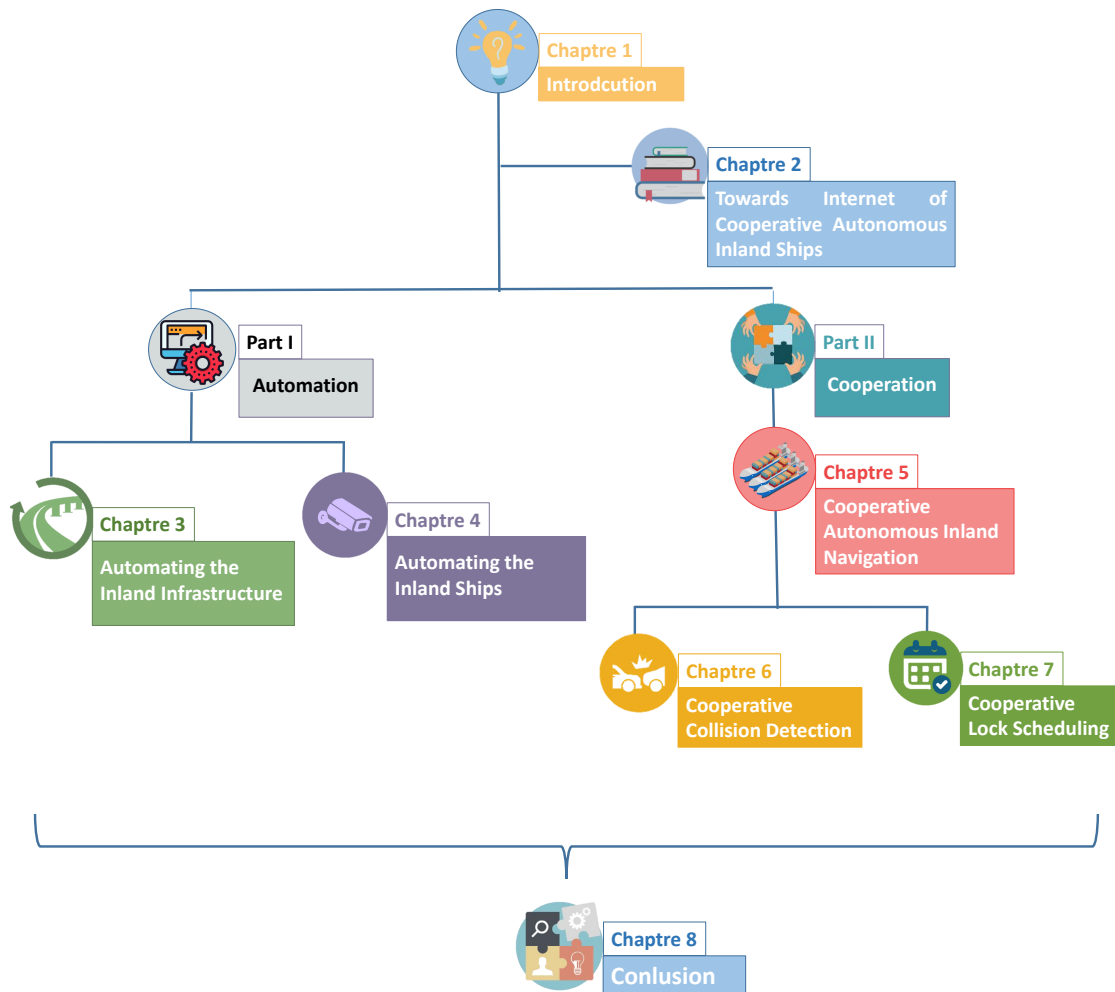


Figure 1.2: Dissertation outline diagram

Apart from the introductory Chapter 1, this thesis consists of seven chapters divided into two parts.

Chapter 2-Survey on Cooperative Autonomous Inland Ships

This chapter explores the different state-of-the-art methods, techniques, and technologies that fall within the scope of our proposed solutions for cooperative, connected autonomous inland navigation deployment. Before the discussion of this specific inland domain, we also

investigated the other types of transportation domains. This review aims to identify the existing solutions and reveal the extent of their applicability to our domain.

Part I-Automation

The first part of this thesis is dedicated to studying the feasibility of automation in the inland environment, i.e., automating the inland infrastructure and the inland ships.

Chapter 3-Automating the Inland Infrastructure

Our first contribution, introduced in this chapter, deals with the infrastructural limitations. We first highlight the specific issues to be addressed by studying the current French river network's states. Then, we focus on making infrastructural modifications by proposing a realistic Lock Automation Decision Making (Lock-ADM) algorithm that integrates all the previously discovered elements. Lock-ADM is a three-stage algorithm. Firstly, we calculate the optimal number of locks while minimizing the investment costs using the exact solver, CPLEX. Secondly, we measure the importance of locks in the network, and finally, we select the best locks to automate using the Genetic Algorithm (GA) metaheuristic.

Chapter 4-Automating the Inland Ships

This chapter develops the vision-based environment perception part, starting with constructing a dataset, the InlandAutoDetect dataset, adapted to the inland domain. The InlandAutoDetect dataset, including a total of 3,377 images, is destined for the fluvial environment with annotations for perception purposes and, more precisely, for object detection through machine/deep learning processes. The dataset is open-source, and it is available online for other researchers working in this area. Then, we evaluate the accuracy and performance of nine state-of-the-art perception models architecture for object detection using the constructed dataset. Among the evaluated models, we select the most efficient and propose an adequate configuration for inland navigation: the Retinanet model with the resnet101 backbone. We design a system to accurately delimit the area where the navigation is secure by simultaneously locating and mapping the area using the Retinanet model findings.

Part II-Cooperation

Once the automation problem is appropriately addressed in the first part, the next logical step is to consider the cooperation between the different river entities. Thus, the second part of this thesis mainly focuses on studying this cooperation's requirements, advantages, and achievable useful applications.

Chapter 5-Cooperative Autonomous Inland Navigation

This chapter introduces the Cooperative Inland Autonomous Ships (C-IAShips) architecture that meets the needs of a smart river, the fleet management-related, and the connectivity-related requirements. The proposed architecture is blockchain-based federated learning architecture. It comprises three interactive layers: a user layer containing the cooperative river components, an edge layer composed of MEC servers, and a cloud layer managed by a

central authority responsible for managing the fluvial navigation. The architecture enables cooperative applications, such as Cooperative Collision Detection and Cooperative Lock Scheduling.

Chapter 6-Cooperative Collision Detection

This chapter investigates the first application enabled by the C-IAShips architecture. It consists of continuously gathering data, such as localization data, from ships and processing them at the MEC level to predict collisions. Then, alerts will be sent to ships to avoid collisions between them.

Chapter 7-Cooperative Lock Scheduling

This chapter addresses another application based on the CIAShips architecture of cooperative fleet management, cooperative lock scheduling. It aims to reduce the fleet's waiting time by scheduling the ships before their arrivals to the lock using a lock-ship real-time and dynamic cooperation. Additionally, it provides recommendations regarding the coming ships' speed to minimize their fuel consumption.

Chapter 8-Conclusion

The closing chapter of the thesis provides a general overview of our work. Furthermore, a detailed review of our contributions and the limitations of the current iteration of our approach are presented. Its final section outlines the potential future research directions related to cooperative autonomous navigation deployment to improve inland transportation.

SURVEY ON COOPERATIVE AUTONOMOUS INLAND SHIPS

Survey and test a prospective action before undertaking it. Before you proceed, step back and look at the big picture, lest you act rashly on raw impulse.

– Epictetus

| | | |
|-------|---|----|
| 2.1 | Introduction | 11 |
| 2.2 | Autonomous Inland Ships | 11 |
| 2.2.1 | What is an Autonomous System ? | 12 |
| 2.2.2 | Key Enabling Competencies of an Autonomous Vehicle | 15 |
| 2.2.3 | Opportunities and Challenges Related to Autonomous Inland Ships | 22 |
| 2.3 | Cooperative and Connected Inland Ships | 32 |
| 2.3.1 | What is a Cooperative and Connected Vehicle? | 32 |
| 2.3.2 | Underlying Technologies for Cooperative and Connected Autonomous Inland Ships | 36 |
| 2.3.3 | Opportunities and Challenges for Connected and Cooperative Inland Ships | 40 |
| 2.4 | Conclusion | 44 |

2.1 Introduction

The Internet of Things (IoT) has witnessed rapid growth—both in scope and number of smart devices and technologies. In particular, connected and autonomous vehicles (CAVs) represent a significant part of this IoT industry. Associated with the different components of intelligent transportation systems, they form the concept of Internet of Vehicles (IoV) [5]. The latter is a distributed network that supports data created by connected vehicles. The IoV's essential goal is to allow vehicles to communicate in real-time with the human drivers, pedestrians, other vehicles, roadside infrastructure, and fleet management systems. In this context, the Internet of Ships (IoS) [6] is defined as a novel vehicular ecosystem that incorporates all IoV-based emerging technological trends adapted for sea shipping. As a particular case of IoS, the Internet of Inland Ships (IoIS) involves river ships. The IoIS combines communication and computing capabilities to provide outstanding inland transport services, such as real-time monitoring and route planning. However, with its distinctive characteristics and specifications, inland navigation has given birth to different deployment requirements and challenges to support such an intelligent transport system. As described in the introductory chapter, in this work, we study the feasibility of employing a smart river where cooperative autonomous ships can communicate with the surrounding automated infrastructures to build a robust IoIS. Therefore, we conducted a survey on cooperative autonomous inland ships solutions over the past years. Furthermore, we did not limit the scope of this review to the inland domain; we also investigated the other types of vehicles.

Based on the literature collected, a survey of the current state-of-the-art algorithms and methodologies has been pursued to identify the most suitable key enabling technologies to address the problem of Cooperative Autonomous Inland Ships (C-AIS) considered in this thesis. We use the terms 'unmanned ships' and 'autonomous ship' interchangeably in this manuscript. For clarity of exposition, we separate the research studies into two main sub-fields. On the one hand, section 2.2 study the autonomous inland ships from an individualistic perspective. On the other hand, in section 2.3, we consider cooperative and connected inland ships. We explain the main requirements, opportunities, and challenges for each sub-field for considering such an advancement. Finally, section 2.4 concludes on specific approaches for building a robust Internet of Cooperative Autonomous Inland Ships.

2.2 Autonomous Inland Ships

The Intelligent Transportation Systems (ITSs) will revolutionize transportation by providing safer and more efficient driving experiences. One of the most significant contributions towards the ITS is developing autonomous vehicles. Over the last decades, the automotive industry has shown interest in integrating new technologies into vehicles' design to reduce the driver's role and exclude him altogether. Seeing the enormous potential to increase driving safety and efficiency [7], autonomous ships will hopefully soon become a reality. Thus, the need to adapt to the specific needs of this advancement in the inland environment becomes apparent.

This section intends to elaborate on the feasibility of deploying autonomous inland navigation. First, we define autonomous systems in general, their level of automation, and their benefits. Then, we explore the core competencies of the vehicle's automation. Next, we investigate the existing projects related to autonomous ships, their opportunities, and challenges.

2.2.1 What is an Autonomous System ?

Autonomous systems (ASs) are machines and systems that can sense and operate without human involvement in an unpredictable and partially unknown environment. They were first invented for the defense and security industries to take jobs too dangerous or unpleasant for humans to do. Nowadays, the deployment of such systems is rapidly growing in a diverse number of applications, as shown in Figure 2.1.

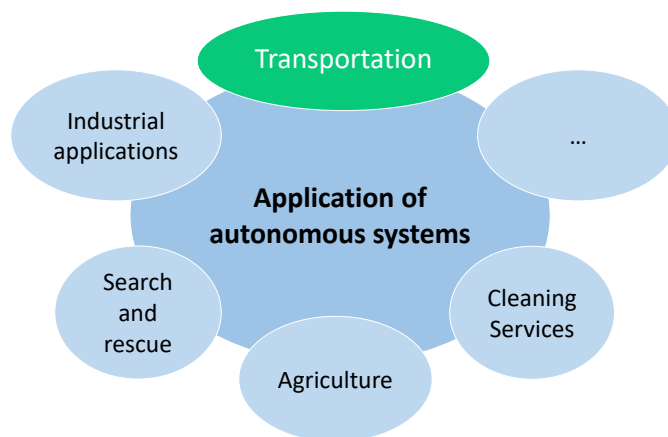



Figure 2.1: Applications of autonomous systems.

In particular, autonomous transportation and especially autonomous cars has gained prominence in the most varied fields of applications of ASs missions. The idea of autonomous cars started with "phantom autos" in the 1920s, where the car was controlled through a remote control device [8]. In the 1980s, we witnessed the emergence of autonomous and self-managed autonomous cars. A significant contributor to the autonomous car field was the NavLab at Carnegie Mellon University, where researchers developed the Autonomous Land Vehicle (ALV) [9]. Although it was not fully autonomous at that time, since the human intervention was necessary for safety reasons, automatically changing lanes was a breakthrough. In the 21st century, the increasing interest in autonomous cars has been fueled primarily by low-cost, high-performance technologies in various areas. Advanced Driving Assistance Systems (ADASs) and Automated Driving Systems (ADSs) are being developed to lead to better road safety and lower congestion where the traditional transport system is becoming increasingly disorganized and inefficient [10].

2.2.1.1 Levels of Automation

In order to standardize a conventional definition of what is considered autonomous, in 2014, the Society of Automotive Engineers (SAE), in collaboration with the International Organization for Standardization (ISO), defined six levels of vehicle automation [11]. Table 2.1 draws clear lines between the conflicting levels of autonomy ranging from Level 0 (fully manual) to Level 5 (fully autonomous).



| Level 0: No automation | Level 1: Driver Assistance | Level 2: Partial Automation | Level 3: Conditional Automation | Level 4: High Automation | Level 5: Full Automation |
|---|---|---|--|--|--|
| The driver is entirely responsible for controlling the vehicle and performing all driving tasks such as steering, braking, accelerating, or slowing down. | “Hands on” The driver controls the vehicle, but the vehicle’s ADAS can assist the human driver with either steering or braking/accelerating . | “Hands off” The vehicle’s ADAS can perform complex functions, such as acceleration and braking, in specific situations. The driver must, however, remain engaged with driving tasks | “Eyes off” Drivers can disengage from the act of driving, but only in specific situations. The vehicle’s ADS can perform all aspects of the driving task, Conditions could be limited to certain vehicle speeds, road types, and weather conditions. | “Mind off” At this level, the vehicle’s ADS can fully handle all driving functions. It may alert the driver when it requires a human in control. If the driver does not respond, it can secure itself automatically. | Level 5-capable vehicles are fully autonomous, and their ADS can ensure all the driving tasks in all circumstances. No driver is required behind the wheel at all. |
| Driver support features | | | Automated driving features | | |

Table 2.1: Vehicle automation levels as described by SAE and ISO.

Some ADASs corresponding to level 2 are already implemented in commercial vehicles, such as Adaptive Cruise Control, Lane-Keeping Assist, and Automatic Emergency Braking. They control the steering or speed based on sensor measurements like distance or visual lane recognition. Moreover, the Audi A8 vehicle with its AI Traffic Jam Pilot 2 has reached level 3. It can perceive the environment and operate under certain conditions. However, they require the driver to stay alert in case of an unexpected event that will require to take over the control. Level 4 vehicles already exist (taxis, low-speed shuttles). However, they are pilot projects restricted to operate under certain conditions (restricted areas, weather, speeds) and analyzed with continuous test evaluations to prove their efficiency. Level 5 vehicles will not have operational restrictions and will be able to navigate anywhere at any time.

Each level of automation requires additional layers of sensors, as the vehicles increasingly assume functions previously controlled by the driver. For example, a Level 1 vehicle might only have one radar and one camera, whereas Level 5 vehicle will require full 360-degree sensing across multiple sensor types. The sensor architecture in a modern autonomous driving system notably includes multiple sets of cameras, radars, Global Positioning System (GPS), and LiDARs for absolute localization of the vehicle in space.

2.2.1.2 Benefits of Automation

The benefits of self-driving vehicles are numerous. Even though many experts have explored them extensively, we list here some of those potential benefits.

Safety.

The benefits of automated vehicles in terms of safety are predominant. Studies have found that over 90 % of road crashes are due to human error. The elimination of human error through the widespread adoption of autonomous vehicles is thus expected to save lives and reduce injuries, with some experts [12] expecting the reduction as high as 80 %.

Efficiency and convenience.

In 2014, Americans lost an estimated 6.9 billion hours due to traffic delays, fuel costs, and vehicle emissions. The time and money spent commuting could be better spent with automated vehicles. According to [13], automated vehicles could free up to 50 minutes per day that were previously devoted to driving.

Lower fuel consumption.

Autonomous vehicle technology can improve fuel economy by 4–10 % by accelerating and decelerating more smoothly than a human driver [12]. Further improvements could be achieved by reducing the distance between vehicles and increasing roadway capacity. Over time, as the frequency of crashes is reduced, cars and trucks could be made much lighter. This would increase fuel economy even more.

Economic and societal benefits.

On the one hand, automated vehicles could deliver additional economic and additional societal benefits. The National Highway Traffic Safety Administration (NHTSA) study showed that motor vehicle crashes in 2010 cost \$242 billion in economic activity, including \$57.6 billion in lost workplace productivity and \$594 billion due to loss of life and decreased quality of life injuries. Eliminating the vast majority of motor vehicle crashes could erase these costs. On the other hand, postings for jobs in autonomous vehicles have been increasing steadily over the last several years [12]. Several experts expect that autonomous vehicles will result in an uptick in engineering-related positions and other positions related to their integration into society.

However, these gains are circumscribed by the capability of an autonomous vehicle to learn and adapt to the driving environment. In reality, driving involves complex interactions with other road users that are near-impossible to be exhaustively described through code or rules. Therefore, autonomous driving systems cannot be pre-programmed with specific rules to cover all possible scenarios on the road. AVs should, thus, potentially discover such complex situations automatically through exploration. Based on the knowledge gathered through interplays with the driving environment, they should evolve their planning and actions to be more successful in driving. More specifically, an autonomous vehicle

needs to develop some key enabling competencies to ensure trustworthy self-driving capability. We detail these competencies in the following section.

2.2.2 Key Enabling Competencies of an Autonomous Vehicle

The autonomy of vehicles heavily relies on their ability to make decisions based on the information provided by their sensors. They have to adapt their decisions to their environments using their sensors and internal memory to analyze and update their representation of both the state of the world and their inner state. These functional systems must cooperate to answer the three critical questions of "Where is the vehicle?", "What is around the vehicle?", and "What does the vehicle need to do next?" to achieve fully autonomous operation. Hence, their basic software proficiencies can be classified into three main groups, namely: *Perception*, *Planning*, and *Control*, with the interactions between these proficiencies and the interactions of the vehicle with the environment. Figure 2.2 explains the standard blocks of an autonomous system, demonstrating the pipeline from sensor stream to control actuation.

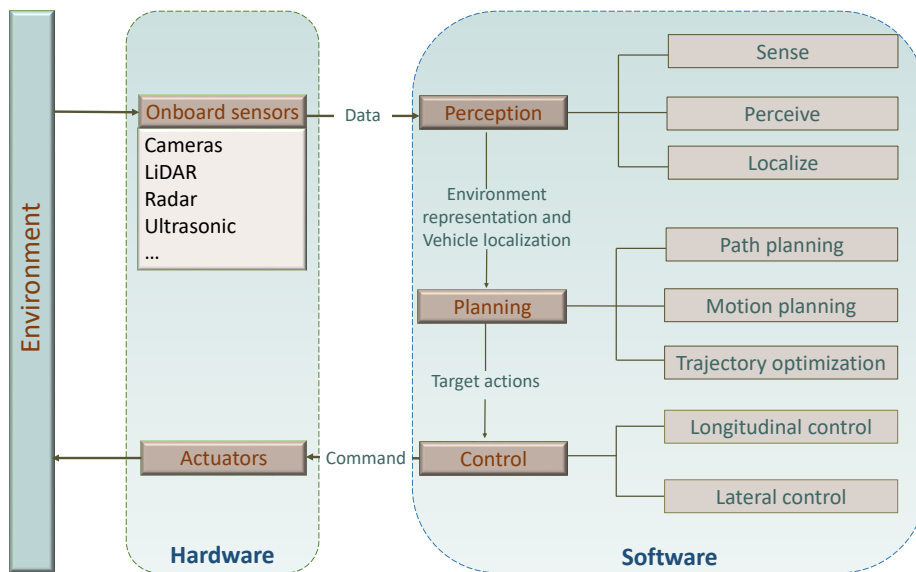


Figure 2.2: Standard components in an autonomous driving system pipeline listing the various tasks.

2.2.2.1 Perception

Environment perception refers to an AV’s capability to collect and interpret sensory information to represent the environment and localize itself. Developing a contextual representation of the environment involves localizing obstacles, detecting traffic signs, and categorizing data according to their semantic meaning. Localization refers to the ability of the vehicle to determine its position with respect to the environment. Perception thus is the first critical part of the computational pipeline for the AV’s safe functioning. Once

the vehicle can select relevant data from the enclosing environment, it can accomplish the other planning and control tasks without human intervention.

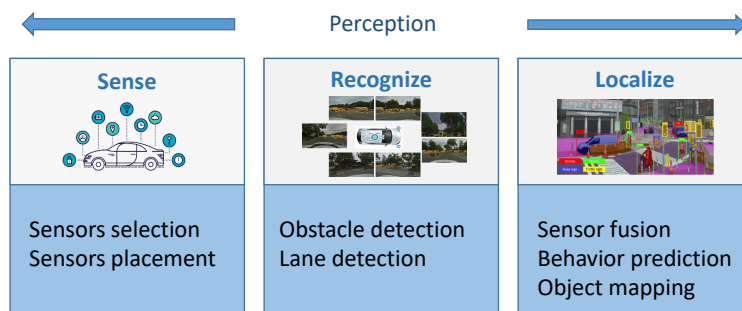


Figure 2.3: Perception main steps.

Fig 2.3 shows the three main steps that should be assured to develop a robust perception capability:

- **Sense** the environment and keep track of the vehicle's current state using onboard sensors and models,
- **Recognize** and understand disparate data sources,
- **Localize** itself and represent the environment and other vehicles/people behavior.

2.2.2.1.1 Sense :

Sensors play a vital role in autonomous vehicles. They help them get their location, identify informational signs, and avoid obstacles and other hazards. The following describes common types of sensors and their purposes:

- **Ultrasonic sensors:** Ultrasonic sensors use high-frequency sound waves to estimate distances between the sensor and an object. However, they require precise modeling for their application since their performance is highly dependent on physical properties [14]. Therefore, they are commonly used in short-distance applications, such as a vehicle's parking assistance system.
- **Radar sensors:** Radar sensors in autonomous vehicles monitor blind spots using short-range (24 GHz) and long-range (77 GHz) radio waves for safer distance control and braking assistance applications. However, a drawback of the radar is their high false-positive rate when detecting objects and an upper bound on the number of objects that can be detected at the same time, e.g., the Bosch midrange radar with a maximum range of 160 meters can only detect up to 32 objects simultaneously [15].
- **LiDAR sensors:** LiDAR sensors use invisible laser light to estimate distance and create 3D images of their surroundings. However, LiDAR relies on rotating systems with a full 360° view around an autonomous vehicle, making it prohibitively expensive and more vulnerable to damage. Additionally, LiDAR data processing is computationally costly.

- **Camera sensors:** Video and image cameras are examples of standard optical sensors. They provide a 2D view of the environment, which is essential for identifying cars, pedestrians, traffic signs and signals, and road markings in a traffic setting. Using a camera as a vision sensor is a widely used approach to classify and detect objects on the road. A 3D view can also be provided using a couple of cameras. However, cameras are not reliable in extreme weather, or lighting conditions [16]. Hence, infrared sensors providing images under low-lighting conditions, such as night-vision systems, may be a suitable alternative.
- **GPS sensors:** GPS sensors are global radio-navigation systems with antennas that use a satellite-based navigation system. They are the most representative and widely used system, suitable for open-sky environments such as outdoor and rural areas to provide position, velocity, and timing data [17].

Emerging autonomous vehicles should handle two main challenges related to sensing capability: onboard sensors selection and placement.

First, each sensor has some faults as well as a different range of applications. Considering aspects of functional safety, a sensor fusion setup would aid in acquiring and developing a more accurate assessment of the environment. It entails using multiple sensors to combine the strengths of different sensor types, e.g., a radar sensor's accurate distance and velocity information and the precise bearing measurement of a camera. In [18], Daimler researchers presented Bertha, an autonomous Mercedes Benz S-Class equipped with close-to-market sensors. They demonstrated their expertise by driving from Mannheim to Pforzheim, Germany, in a fully autonomous manner. The chosen route comprises urban streets and rural roads, including traffic lights, roundabouts, and pedestrian crossings. The perception system was realized using a stereo camera, two mono cameras, and several short and long-range radar sensors. Researchers in [19] designed the BRAiVE autonomous vehicle. They demonstrate advanced safety applications with a low-cost sensor suite, mainly based on vision, instead of many other autonomous vehicle implementations based on expensive and invasive sensors. BRAiVE's sensing suite is based on 10 cameras, 4 laser scanners, 1 radar, 16 fixed laser beams, and a GPS mounted all around the vehicle to obtain a 360° all-round coverage.

The second vital challenge is determining a suitable sensor configuration responsible for environment perception. The faulty placement of sensors may create blind spots causing a vehicle accident when it makes a lane change. Therefore, an optimal sensor configuration should consist of each sensor's carefully selected location and orientation in a heterogeneous suite of sensors to maximize coverage from the combined field of view obtained from the sensors and maintain a high object detection rate. There are no generalized rules for synthesizing sensor configurations, as their location and orientation depend heavily on the target features. [20] proposes VESPA framework to optimize heterogeneous sensor synthesis. The framework uses the genetic algorithm to perform intelligent algorithmic design space exploration to determine the optimal placement and orientation for each sensor on the vehicle to support the required ADAS.

2.2.2.1.2 Recognize :

In this step, the raw heterogeneous data from the sensors is processed to extract related

information about the surrounding environment. Many software, algorithms, and frameworks are developed in the literature to process the data according to each type of sensor. Specifically, existing autonomous driving systems deal essentially with obstacle detection and lane detection. The solutions vary according to the sensor configuration, measurement type, and corresponding performance requirements in each application.

❖ Obstacle detection.

Obstacle detection consists mainly of components that locate obstacle instances from several predefined categories in an image. Although radars are useful for long-range detection, they do not provide visuals of targets, making locating the danger more troublesome and less viable. Moreover, radars have dead zones near their base area, and the powerful radio beams are also destructive to living creatures. These shortcomings make visual surveillance systems an attractive option to assist and supplement radars and other sensors. The point cloud registration algorithms using Lidar data have been an important topic in computer vision and robotics, which finds the best transformation (e.g., rotation and translation) that matches two different point cloud sets. Traditionally, the Iterative Closest Point (ICP) algorithm [21], which allows the distance between two data set points to be minimal through a repetitive performance inspection, has been widely used. Various improvements are still made, such as NICP [22], and voxelized GICP [23]. ICP results generally do not always guarantee optimal global performances and highly depend on its initial guess. Given an incorrect initial pose, the ICP-based method can generate a local optimal or wrong solution. In addition, there is a disadvantage in that the number of computations increases in proportion to the amount of point cloud data. In recent years, works employing cameras have increased outstandingly, owing to the sensor's effective dissemination. However, creating a robust and efficient obstacle detection method is still challenging due to brightening variety, background clutter, and perspective alterations. Nowadays, many impressive Deep Neural Networks (DNNs) have been proposed to help improve performance. Before the rise of deep learning, Histogram of Oriented Gradients (HOG) detectors [24] were the most popular solution.

When deep learning methods began to be integrated into detection problems, the main objective was to replace HOG with a more accurate DNN based detector. Currently, deep learning has achieved an impressive series of results thanks to its success in automatic feature extraction via multi-layer nonlinear transformations, especially in computer vision. In particular, over the last years, Convolutional Neural Network (CNN) architectures have emerged as a successful solution for problems related to visual object recognition [25], [26]. However, training DNN models require a large amount of labeled data. Traditional datasets for autonomous driving, such as KITTI [27], and Cityscapes [28], do not have enough data to deal with complicated scenarios in all navigation environments. Consequently, an imperative need is to construct different datasets adapted to each new navigation domain.

❖ Lane detection.

Lane detection is an essential component within the street scene examination for ADAS to avoid vehicles run in the wrong direction. Since an autonomous vehicle requires information on the road where it circulates, the system and algorithm should be as fast and

straightforward as possible. Many conventional approaches detect the lane using the following information set: edge, color, intensity, and shape. These approaches select a candidate lane solution that maximizes the statistical distances between the two regions' color distributions created by the candidate solution. For instance, in [29], a gradient-enhancing conversion and an illumination-based lane detection algorithm are proposed. Gradient-enhancing conversion produces a color image from an intensity image that has maximized gradients at the lanes. In [30], sensor data from the camera image, velocity meter, and steering wheel encoder are fused, and a lane model was proposed as a series of connected rectangular plates.

2.2.2.1.3 Localize :

Localization is a fundamental capability of autonomous driving. Knowledge of precise vehicle location, coupled with highly detailed maps (often called High Definition (HD) maps), adds the context needed to drive confidently. Thus, mapping approaches have to be robust even in situations where the vehicle's metric position estimation is largely erroneous. Topological maps contain relative information about places in an environment. It consists of an ordered collection of images: a linear database that reflects how places are consistently encountered when an environment is visited. In these cases, localization identifies the most likely location. Metric maps accurately depict the absolute scale of environments, maintaining much information about environment details, such as distances, driving direction, or landmark position. They are usually referenced according to a global coordinate system. This representation is most appropriate for vehicle localization and guidance. However, metric maps are more difficult to build and maintain and are computationally demanding. Recent works enable the vehicle to localize itself based on a previously visited location, tagged with GPS information throughout the place recognition process. It can then achieve its localization by assimilating its position to the retrieved image through the recognized place [31]. Using place recognition-based visual localization, accumulation error that often occurs in the odometry-like approach can be avoided. The system also should be noted that it identifies the most likely places and then gets a rough position. Initially, probabilistic estimation techniques were introduced like Kalman Filters (KF), which were later extended to Extended Kalman Filters (EKF), and Unscented Kalman Filters (UKF) for non-linear systems [32]. Later, researchers developed graph-based Simultaneous Localization And Mapping (SLAM) techniques, such as Oriented fast and Rotated Briefs-SLAM (ORB2-SLAM) [33]. Recently, since the advent of deep learning focused on Convolutional Neural Networks (CNN), quite interesting results were observed, especially with the work on CNN-SLAM [34]. Experiments show that vehicle localization could be achieved from a pair of images. In the same context, Sünderhauf et al. [35] present a novel environment representation and recognition system built on state-of-the-art object detection methods and convolutional visual features. As illustrated in Figure 2.4, the astonishing power of CNN features is used to identify matching landmark proposals between images to perform place recognition over outer appearance and viewpoint variations. The experiment results have also revealed further insights: midlevel CNN features appear to be highly suitable as descriptors for landmarks of various sizes in a place recognition context.

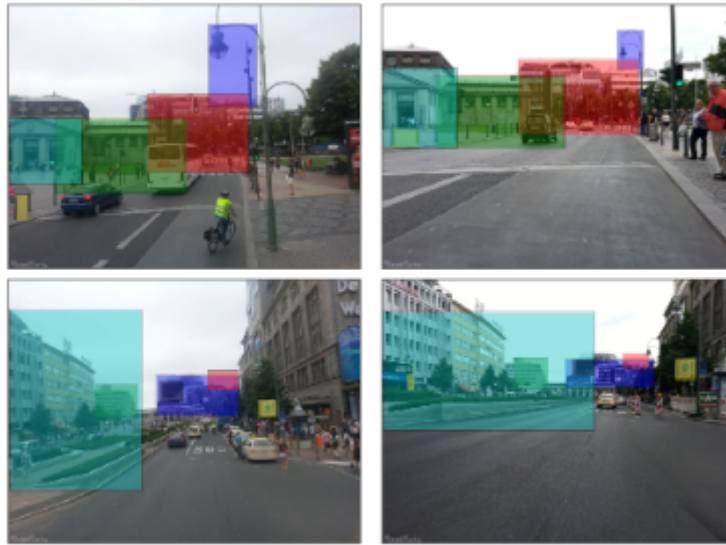


Figure 2.4: Examples of matched scenes from the Mapillary dataset (adapted from [35]). Images in a row belong to the same place but have been taken from different viewpoints, i.e., from the bike lane and the upper deck of a tourist bus. The colored boxes illustrate some of the extracted and correctly matched landmarks.

2.2.2.2 Planning

The planning step is derived from perception information. It refers to the process of making purposeful decisions in order to achieve the vehicle’s higher-order goals, typically to bring it from a start location to a goal location while avoiding obstacles and optimizing its trajectory. Path planning can be divided into global path planning (also known as Vehicle Routing Problem (VRP)) and local path planning (also called motion planning). Global path planning [36] deals with generating the ideal trajectory to reach the final destination. Local path planning [37] is responsible for obstruction avoidance for each segment generated by the global path planner. The vehicle routing problem is one of the most popular combinatorial optimization problems. Its study has given rise to several exact algorithms and heuristics of general applicability [38]. Comparative studies include the following VRP variants: (1) the VRP with trailers where trucks and trailers are routed to customers who may be served by either a truck or a truck-trailer pair but not a trailer itself [39]; (2) the pickup and delivery problem in which each heavy resource (e.g., van) can transport a set of light resources (e.g., scooters or foot couriers) for serving customers [40]; and (3) the two-echelon VRP in which freight is first transported from a central depot to satellite facilities by large vehicles, from where it is then brought to the final customers by small vehicles [40].

Motion planning is based on local environment information to obtain local guidance information, such as obstacle avoidance. One successful motion planning strategy should react to all vehicle encounters, including collision avoidance and generating the lowest crash severity path when avoidance is impossible. Tremendous research efforts have been continuously devoted to developing efficient motion planning approaches for AVs. The early works mainly use graph search-based planners to generate the shortest paths on the graph

constructed by the discretization of the environment. The most usual methods applied to AVs have been the Dijkstra, and A* state lattice algorithms [41]. However, the generated path may not be optimal for dynamic vehicle behaviors. Trajectory optimization-based approaches have become the state-of-the-art AV motion planning approaches in recent years [42]. The core of this technique is formulating the motion planning problem as an optimization problem, taking into account the desired vehicle performance and multi relevant constraints. To this end, [42] proposed a motion planning method based on Model Predictive Control (MPC), which can handle multi-constraints and convex problems. The method solves a sequence of finite-time trajectory optimization problems recursive and can consider updating the environment states during its planning process. Four scenarios were simulated to show that the proposed MPC algorithm could avoid obstacles and, if the collision was inevitable, could mitigate the crash.

2.2.2.3 Control

The control competency refers to the vehicle's ability to execute the higher-level processes' planned actions. An essence of this step is that the vehicle should act only when it is safe to do so, avoiding situations that pose a risk to human safety, property, or the autonomous vehicle itself.

There are two classes of vehicle control [43]. The first class is longitudinal control, which deals with the movement of the forward and backward direction of the vehicle (responsible for regulating the vehicle cruise velocity) [44]. The second one is lateral control which deals with sideways movements perpendicular to the vehicle's heading, in other words, the steering of the vehicle to follow a given trajectory [45]. Usually, two separate controllers are employed to deal with each of them. In previous literature, various studies have been carried out to explore different hypotheses and techniques. These studies incorporate the Proportional-Integral-Differential (PID) control strategies [46], the predictive control paradigms [47], and the model reference adaptive control methods [48]. In addition, some of these studies focused on intelligent approaches like fuzzy control techniques [49], Support Vector Regression (SVR) method [50], and NN-based control strategies [51]. Recent efforts attempt to employ model-free methods and learn from raw sensor data, such as the Reinforcement Learning (RL) [52] technique that improves its control policy by interacting with the environment.

To summarize, the core of the self-driving system involves three parts: perception, planning, and control. Autonomous driving technology means that: self-driving vehicle can recognize its status and its surrounding environment through a variety of on-board sensors (camera, lidar, radar, GPS, etc.), and make analysis and judgment, then autonomously control vehicle movement, and finally achieve self-driving based on the acquired environmental information (including road information, traffic information, vehicle location, and obstacle information, etc.)

2.2.3 Opportunities and Challenges Related to Autonomous Inland Ships

By analogy with other vehicles, autonomous inland ships must likewise acquire the key enabling competencies discussed earlier. However, integrating these competencies in inland driving involves different challenges related to its natural and infrastructure peculiarities. Currently, human-crewed transport ships are equipped with autonomous systems responsible for optimizing some ship equipment and processes (loading and unloading in the port, stability, forecasting the navigation route, and optimizing propulsion system parameters). However, there is still a crew onboard responsible for making the final decisions. An autonomous system installed on the ship collects all parameters related to the ship's operation and environmental parameters in real-time. It also has guidelines based on the developed algorithms that calculate the ship's optimal operating parameters. In the future, as ships become equipped with increasingly advanced automation functions, it is expected that the first fully autonomous cargo ship operating without a crew will be commercially available by 2035 [53].

This subsection will define these peculiarities compared to aerial, land, maritime, and underwater autonomous vehicles. Then, we present existing initiatives related to autonomous inland navigation in particular.

2.2.3.1 Different Types of Autonomous Vehicles

Autonomous vehicles can be classified into four categories, namely, Unmanned Aerial Vehicles (UAVs) [54] [55] [56], Autonomous Land Vehicles (ALVs) [9] [57], Autonomous Underwater Vehicles (AUVs) [58] [59], and Unmanned Surface Vehicles (USVs) [60] [61]. Different driving environment implies different characteristics of vehicles and thus different specific challenges. In particular, the USVs are watercraft of small (<1 tonne) or medium (100 tonnes) size in terms of water displacement. The technology of USVs dates back to World War II. However, significant efforts towards development and understanding the technology started in the 1990s after the successful implementation of USVs in the 1990-1991 Gulf war [62]. Their primary purposes are military surveillance, safety improvement by reducing human-related accidents, environmental monitoring, ocean and scientific research, and hydrocarbons exploration.

Although the term "autonomous ship" appears, at first glance, to imply a concept in which a ship has a system fully responsible for all aspects of its navigation and is independent of humans, autonomous ships may have different Levels of Autonomy (LoAs). It is worth noting that, while the classification given in Table 2.1 is intended to be generic, it was designed primarily with land vehicles in mind. Unlike ALVs, USVs are expected to have various kinds of automation because multiple crew members are responsible for tasks other than ship handlings, such as engine maintenance and cargo monitoring. An unmanned ship may have a certain number of people on board, but it does not need to be performed or supervised by the people. Even if only ship handling is automated, a fully uncrewed autonomous ship still cannot be realized. Therefore, authors in [63] define 8 different LoA taxonomies specific for vessels, as explained in Table 2.2.

| LoA | Description |
|-----|------------------------|
| 1 | Direct control |
| 2 | Decision support |
| 3 | Automatic bridge |
| 4 | Periodically unmanned |
| 5 | Remote control |
| 6 | Automatic |
| 7 | Constrained autonomous |
| 8 | Fully autonomous |

Table 2.2: Levels of autonomy for an USV from [63]

Therefore, USVs may represent the ships essentially belonging to categories 3-8, as given by [63]. Researchers essentially concede that full autonomy, whereby the Artificial Intelligence (AI) is so complete as to obliterate the need for humans, is not a realistic operating model is not likely to be implemented in near future. It emerges that humans remain central to the safe operation of USVs, despite the promise of more machine autonomy. We observe this 'irony of automation' emerging in the wake of unmanned ships, manifested in the Shore Control Centre (SCC) concept. In current operating models of USVs, they may have a dynamic LoAs: the LoA may change in the same voyage depending on certain conditions. Hence, the operators' tasks may change during a voyage. The SCC serves a backup role predominantly: they may have to control the vessel remotely during parts of the voyage when the AI fails to solve a navigation problem and, once resolved, return the ship to the autonomous state. The autonomous ship comprises, thus two basic elements: (i) Operators: the human operators working in the SCC, and (ii) Autonomous systems: the autonomous systems include all systems onboard, such as the hardware, software, and communication channels.

According to their specific applications, we distinguish four types of USVs, as described in Table 2.3 along with some relevant reference projects.

| USV | Description | Reference projects |
|-----------|--|--|
| Urban | Small vessels sailing in urban canals or waterways | Autoferry [64], RoBoat [65] |
| Upcountry | Internal cargo vessels for upcountry waterways | H2H Project [66] |
| Coastal | Short-sea shipping routes in trafficked channels with aids to navigation present along the shore, Vessel Traffic Services (VTS) communication, and demanding navigation. | Land-Based Operation of Autonomous Ships ([67]), SFI AutoShip [68] |
| Ocean | Open-ocean transportation with non-demanding | MUNIN navigation [53] |

Table 2.3: Scope of USV types

This thesis is particularly interested in developing Inland Unmanned Surface Ships (I-USV), including urban and upcountry USVs.

2.2.3.2 Inland Unmanned Surface Ships characteristics compared to other Autonomous Vehicles

The transport in inland waterways is different from other AVs. Although the technological complexity might be assumed to be lower at first glance, the overall challenges for autonomous inland shipping are not trivial. For instance, inland ships suffer from a spatially more restricted complex navigational environment and have no help of tugs to maneuver. We show in Figure 2.5 some typical scenes of inland navigation.



Figure 2.5: Some typical scenes of inland navigation

Unlike road vehicles whose operations are generally governed by well-structured and universal roads, the vessels are expected to follow region-specific unstructured spaces where lanes are not well defined. Even though all the USVs seem to have the same characteristics, in reality, I-USVs are different from other USVs, since the fluvial environment characteristics are different from those of the maritime one. Rivers present challenging conditions for autonomous vessels, where the proximity of the riversides provides a cluttered environment with surface water reflections of trees [69]. Analyzing the differences between the two environments would establish a classification into natural features and infrastructure features. In the maritime environment, water's color and texture provide strongly distinctive leads in some situations. However, a variation in weather can completely block these leads in the inland water environment.

Additionally, the river canal is narrower and shallower than the waterway in the maritime environment. Besides, the origins and destinations of the inland ships are more dispersed. So, they need to be smaller to ensure accessibility and flexibility. Smaller ships imply a higher traffic density. Furthermore, inland areas are characterized by many diversified constructions such as locks and bridges, and the limits of the river canal are not marked by physical infrastructure but generally with trees. Some river portions also may have poor visibility affected by the fixed inland infrastructure. Fluvial environments are strongly affected by external factors. They are, thus, ceaselessly dynamic in both spatial (such as the rustle of trees) and temporal dimensions (such as parked vessels). Inland vessels are sup-

posed to navigate in restricted and congested waterways, shallow waters, and sharp river bends, enter and leave locks, and pass under bridges. All these features make the fluvial environment more complex than other environments.

2.2.3.3 Existing Initiatives Related to Inland Unmanned Surface Ships

Despite the growing number of studies on USV, little research has been done on autonomous ships navigating inland waterways. As a result, I-USVs are still in their early stages. Much of the literature focuses on a critical reading of the feasibility of their deployment, forging strong links with governments and technological companies, using a normative approach. Most projects are still in the conceptual stage, examining legal regulations [70], [71], and technical elements necessary for the realization of autonomous ships [72]. Withal, we can find studies investigating improving the competencies of an autonomous ship, namely the perception, the planning, and the control capabilities. We detail them in the following.

2.2.3.3.1 Regulatory Analysis for the I-USV The authors in [71] identified the regulatory obstacles to the deployment of autonomous inland cargo vessels in Europe. They discussed regulations like the international Convention for Safety of Life at Sea (SOLAS) [73] and the Convention on the International Regulations for Preventing Collisions at Sea (COLREGS) [74]. The results of these regulatory conventions cannot, however, be directly applied to inland waterway shipping. This stems from the fact that the inland waterway ships' operations and context differ significantly from the short sea and ocean-going ship applications. Hence, the authors pointed out that the regulations in their current form limit the deployment of autonomous ships for many reasons. First, the use of autonomous seagoing ships is formalized by International Maritime Organization (IMO), responsible for establishing relations between autonomous seagoing shipping and the international maritime regulatory framework. However, there is no such international organization for inland navigation. As a result, the regulatory aspect is fragmented. In Europe, inland waterway regulations are governed by several organizations on both regional and national levels, such as the Central Commission for Navigation on the Rhine (CCNR), the European Committee for drawing up Standards in the field of Inland Navigation (CESNI), and the United Nations Economic Commission for Europe (UNECE). Nevertheless, these organizations have different criteria and safety assessment methodologies, which may generate ambiguities. Hence, there is an urgent need to harmonize regulations for European inland ships.

Additionally, the technical requirements proposed by these organizations contain several barriers that could hinder the introduction of I-USV on European waterways. They either explicitly require the involvement of human operators in data acquisition, monitoring, and control of systems and devices, or do not make provisions for an automated alternative, or allow an automated process to be substituted by direct human involvement. Furthermore, some regulations preclude the remote control of ships. Therefore, the authors propose some possible improvements of safety standards for inland vessels based on early detection of hazards or potentially hazardous situations. This hazardous information could be supplied to a remote control center. The data then could be processed and analyzed, and from

where the human operator of the vessel would be advised on immediate action necessary for hazard prevention.

2.2.3.3.2 Technical Design of the I-USV A key enabler of the previously envisaged solution may be an Inland Shore Control Center (I-SCC) capable of remotely monitoring and controlling inland vessels. The I-SCC's goal is to enhance the situation awareness and sense-making capabilities of a remote human operator. This method was also adopted by the Hull-To-Hull (H2H) project [75]. Accordingly, in [66], the authors investigated the concept and design requirements to achieve an I-SCC that provides interactive services when supervising an I-USV. Four main design requirements were judged necessary to realize the desired I-SCC concept and to provide a fully operational and industrially relevant experimental set-up:

1. Generate and communicate information about the voyage plan, the sailing status, the safety, and emergencies information to the I-SCC. This information will enhance the situation awareness and sensemaking abilities of the remote operator.
2. Provide interaction possibilities with the I-USV to remotely alter its motion and its system configurations. This interaction will also assist the sensemaking and situation awareness capabilities of the operator.
3. Install industrial, marine-grade components for the extensions on the I-USV and the I-SCC system design. This requirement makes the overall system safer, more robust, and closer to a potential future reality.
4. Keep the system design modular and flexible, where possible. This flexibility smoothens the likely design iterations and potential future system extensions.

Finally, the authors provide a technical I-SCC and I-USV system design based on its concept and requirements, as represented in Figure 2.6. The GNSS and IMU sensors (the autonomous part of the sensor subsystem and the Programmable Logic Controller (PLC)) stream their data, i.e., sailing information to the I-SCC. Similarly, the Stereo Cameras and the LIDAR produce observation information. The onboard PLC monitors all wanted parameters and alarms to provide the I-SCC technical, safety, and emergency information. The images provided by the cameras installed on the I-USV offer security and observation information. In the end, the PLC web interface orchestrates the desired actuation system states from the accumulated information.

In the same context, [76] discussed the design of an experimental platform to study the feasibility of autonomous inland cargo vessels regarding its specific technological challenges. The introduced fleet of self-propelled Watertruck+ [77] barges initiative introduces an economically feasible alternative to oversaturated road transport via a flexible model of waterway transport that is complementary to the current waterway transport. Its size matches the research potential for urban water freight transport. In addition, the propulsion system with two 360-degree-steerable thrusters offers the possibility of advanced motion control. The chosen marine-grade and industry-standard sensors can provide information

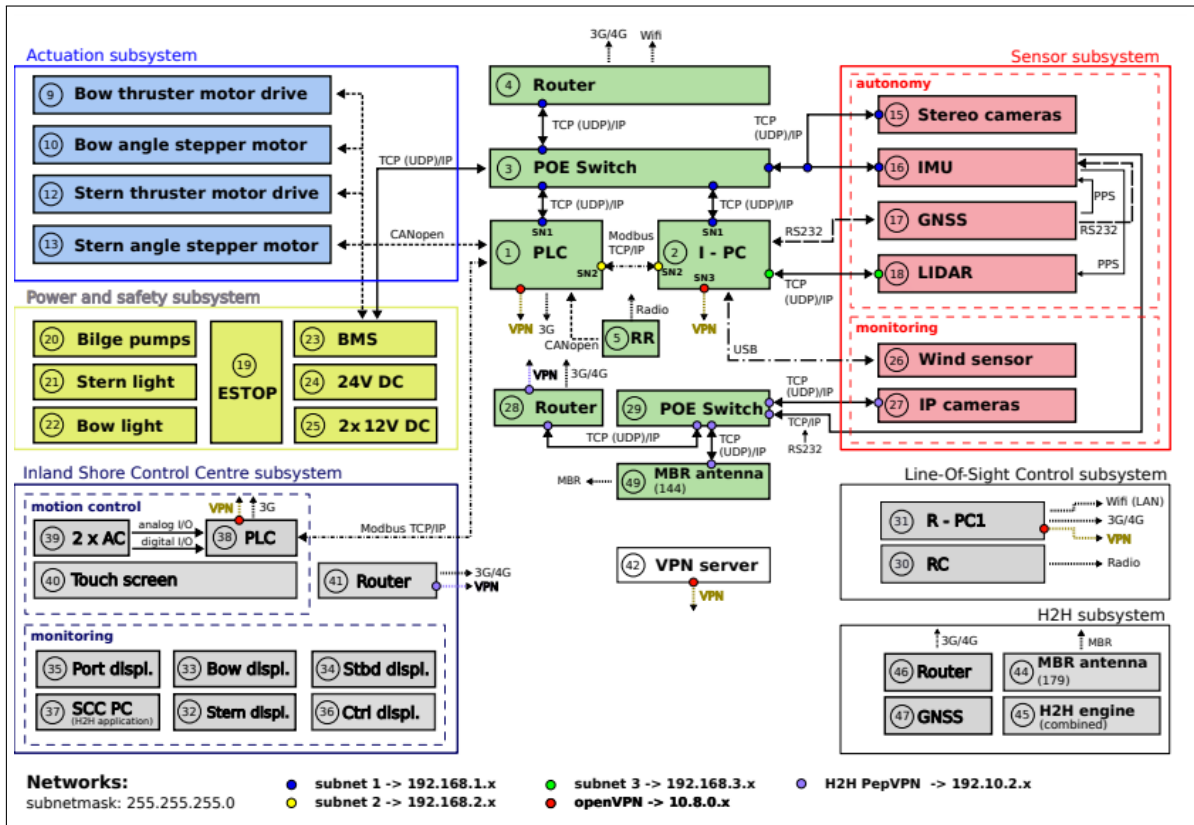


Figure 2.6: H2H-extended I-SCC and USV system design: main components and their communication links (adapted from [66]).

on actual operating vessels and could be transferred to a real-size vessel with minimal modifications required. Furthermore, the barges can sail individually on small waterways or couple together in convoys pushed by a push boat for transport on more extensive waterways.

However, in addition to the industrial relevance of the hardware concept, the challenges involved in actual implementation require an interdisciplinary software approach. This aim can be tailored to three research questions: (i) How can an I-USV perceive its environment? (ii) How can an I-USV automatically plan its path based on the environment information perceived? (iii) How to control an I-USV automatically to follow the planned path? Within this context, researchers feed each other with challenges and solutions.

2.2.3.3.3 Automated Environment Perception for an I-USV The perception of the environment of an inland vessel needs to be explored, and crucial information needs to be shared over the River Information System (RIS). The RIS is a support system used to track inland ships. This system uses various technologies to assist in observation, e.g., the Automatic Identification System (AIS), radars, and cameras. The AIS is an automated tracking system that uses information concerning the ship's position and additional data entered by the navigator, provided that the ship has an appropriate transmitter. However, the I-USV characteristics, mentioned in section 2.2.3.2, significantly influence its perception capabil-

ity. From the very limited literature in this area, one of the main challenges of unmanned ships is the need for analyses of their safety. To ensure the desired safety, two main categories of solutions can be found in the literature:

- ❖ The first group of studies focus on monitoring vessels' traffic by automatic recognition of vessels in inland waters [78].

The identification of ships in the channel plays an essential role in navigation aids and safety control of autonomous ships. Authors in [79] proposed a ship detection method using Synthetic Aperture Radar (SAR), which estimates the probability distribution of ships in SAR imagery. However, SAR-based ship detection methods have two main shortcomings: SAR is often limited to big vessels' detection but may not work well to identify ships in complex contexts. Second, shipborne radar systems usually work efficiently on remote sensing rather than near-side monitoring for ship recognition [80]. For the autonomous ships that work in complex voyage environments like inland rivers, efficiently detecting near and small ships around their channels seems essential to ensure the safety of the voyage mission. Thus, surveillance video systems attracted significant attention in the past decades. However, analysis of the existing works reveals that no system or method is currently available for the automatic identification of inland vessels. The only way to recognize them is to observe markings on their sides. RIS and VTS information systems have video monitoring, where cameras are mounted mainly on river bridges and in ports. It provides an excellent opportunity to observe ships from different perspectives. However, the entire process is not automated; instead, human workers operate these systems. Therefore, to minimize cost, automatic identification of vessels in limited areas is needed.

To this end, we can cite the advancement of the research within the SHREC project [81]. In the proposed system, the camera stream is analyzed using image processing methods based on convolutional neural networks for classification and text recognition to identify a vessel correctly. The system is designed to use multiple existing video streams to identify passing ships on inland waters, especially non-conventional vessels. Tests to verify the proposed method were carried out on a private dataset containing 200 images of four classes of ships. They achieve an average accuracy of 90%. However, real-world conditions unveil more types of inland ships to classify. For this purpose, a more extensive database of images of ships should be gathered using video cameras at varying resolutions.

- ❖ The second group of research focuses on mapping the surrounding environment by automatically recognizing obstacles in inland waters [82].

Different methods were developed to address the problem of surrounding environment detection. [83] proposes a vision-based method to detect the outline of ships via maritime surveillance videos. This method can effectively detect ships in both maritime and non-maritime backgrounds. [84] proposes a hybrid ship detection method that integrates deep learning methods. Specifically, they utilize Deep Neural Networks and Region Proposal Networks (RPNs) to obtain a 2D bounding box of target ships. Similarly, Zhao et al. [85] also propose a two-stage neural network for ship detection and recognition. [86] proposes a Histogram of Oriented Gradients (HOG)-SVM (Support Vector Machine) method to detect

ships on the images from a ship-mounted camera. However, most of the above methods are not feasible to be applied by autonomous ships since they are based on static cameras on ports. They thus do not match the need for moving autonomous ships. Hence, researchers in [87] worked on autonomous shipping for small target detection. They proposed a detection method based on deep learning using a camera to detect small ships in the river or near-shore ocean environments. First, they generate an artificial dataset of small ships using Generative Adversarial Nets (GANs) [88]. Then, they compare some DNNs, such as Faster Region Convolutional Neural Network (Faster R-CNN), Single Shot Multi-Box Detector (SSD), and You Only Look Once (YOLO) on the produced dataset. Nevertheless, even though the proposed method achieved high accuracy, it can not be applied for autonomous driving assistance since it detects only ships.

For more objects, we cite the Roboat project [65], which is a 5-year research project and collaboration between the Amsterdam Institute for Advanced Metropolitan Solutions (AMSS) and the Massachusetts Institute of Technology (MIT), aiming to develop and deploy autonomous ships to learn about Amsterdam's canals. The developed autonomous urban vessels are equipped with a Lidar-inertial sensor, RGB camera, WiFi router and adapter, battery, computer, and microcontrollers. Their software operates using Robotics Operating System (ROS) middleware. Researchers [72] claim that Lidar and camera technologies are coupled to measure the distance to objects (Lidar) and label such objects (cameras with computer-vision algorithms). More precisely, with the lidar-inertial navigation system, Roboat can provide accurate state estimation and high-definition point cloud map of the city by performing real-time Simultaneous Localization and Mapping (SLAM). The drift of the navigation system is eliminated by constructing a factor graph, which takes advantage of loop-closure detection and other sensor corrections. Besides, it can view its surrounding environment via the onboard camera. Perception methods such as clustering and neural network classifiers are used on sensor readings to recognize objects in the canal environment. This enables the Roboat to avoid other boats and obstacles while navigating. However, till the date, detailed technical information about the methods they used is not disclosed.

2.2.3.3.4 Automated Path Planning for an I-USV As explained earlier in section 2.2.2.2, the purpose of path planning algorithms is to plan a collision-free feasible path from the starting point to the target point. Since we can not find works for I-USV, we consider here the USV in general. Existing works on path planning for USV classify it into global path planning and local path planning:

- Global path planning obtains static obstacle information of the driving area through the electronic chart. It uses an A-star algorithm, a distance optimization Dijkstra algorithm, a genetic algorithm, and an artificial potential field method to search for an optimal path from origin to destination.
- Local path planning determines the optimal collision-free path from the current path point to the next. Since the ship's environmental information is complete or partially unknown, it relies on real-time sensors detecting the surrounding environment and obstacles.

Numerous path planning methods have been released in recent years. The most important ones are listed in Table 2.4 with their description and reference projects.

| Classification | Method | Description | Reference |
|---|--|---|------------|
| Global path planning based on environmental information | A-star algorithm | The A-star algorithm performs a Best-first search of the most probable paths leading to the goal. | [89] |
| | Dijkstra algorithm | Dijkstra algorithm is an algorithm for calculating single source shortest path in a weighted directed graph. | [90] |
| | Genetic algorithm | The genetic algorithm simulates natural selection and evolution theory mechanisms to search for the optimal solution. | [91] |
| | Artificial Potential Field (APF) method | APF method is typical online path algorithm. It uses the idea of "water flows to low places" and can naturally understand the generation law of ship routing. | [92] |
| | Ant Colony Optimization (ACO) | ACO algorithm is a bionic optimization algorithm to simulate the intelligent behavior of ants. It has the advantages of robustness and distributed computing mechanism. | [93], [94] |
| Local path planning based on sensor information | Flower Pollination Algorithm (FPA) | FPA is a metaheuristic algorithm, that was developed from the characteristic of the biological flower pollination in a flowering plant. | [95] |
| | Rapidly-exploring Random Trees algorithm (RRT) | RRT is a random data structure. It rapidly expands like a tree to explore most of the space and find a viable path. | [96] |
| | Fuzzy logic | Fuzzy logic is robust and can avoid the characteristics of traditional algorithms, which are sensitive to positioning accuracy and highly dependent on environmental information. | [97], [98] |
| | Neural network | Neural network has the abilities of large-scale parallel processing of data and strong fusion of knowledge and can implement efficient and intelligent path planning. | [99] |

Table 2.4: A summary of path planning algorithms for USV

2.2.3.3.5 Automated Control of the I-USV We summarize in Table 2.5 typical algorithms for control methods for USVs. Unmanned ship navigation may use the mentioned efficient real-time intelligent control methods for collision avoidance to ensure navigation safety.

| Method | Description | Reference |
|---------------------------------|--|---------------------|
| Support Vector Machine (SVM) | SVM is a fast and dependable classification algorithm that performs very well with limited analysis data. It takes the characteristic parameter of multi-sensor as input and outputs the ship's next heading action. An SVM system provides real-time control for a ship's local path planning. To improve the accuracy of collision avoidance decisions, the SVM can be combined with other machine learning methods to monitor learning models, analyze collision avoidance data, and identify patterns. | [100], [101] |
| Finite State Machine (FSM) | FSM is a timing machine that divides a complex problem into smaller, easier-to-manage chunks. It is defined by four elements: current state, condition, action, and secondary state [109]: the current state refers to the navigation state in real-time; the condition is used to determine the state of migration in the next cycle; the action is related navigation operations; the second state is the next compared with the current state. | [102] |
| Line-of-Sight (LoS) | The core idea of LoS guidance is that the ship converges to a constant LoS heading angle between the ship and the target. The LoS angle (Φ_{los}) is calculated in terms of the current (x,y) and LoS coordinates (x_{los}, y_{los}). The equation is $\Phi_{los} = (y_{los} - y)/(x_{los} - x)$. | [103], [104], [105] |
| Multi-Sensor Data Fusion (MSDF) | MSDF refers to the collection, processing and collaborative combination of data collected by various knowledge sources and sensors to provide auxiliary decision-making. Various estimation algorithms available for MSDF include Kalman filtering based approach, hybrid multi-sensor data fusion, fuzzy logic based adaptive Kalman filter, and crisp decision algorithm. | [106], [107] |

Table 2.5: A summary of control methods for USV

The mentioned techniques investigate heterogeneous objectives to ensure efficient control. However, in real implementation, the I-USV control may consider combining the presented objectives to fulfill a high level of autonomy.

2.3 Cooperative and Connected Inland Ships

Optimizing the performance of inland transportation requires, in addition to the automation of the individual vessels, cooperation among them. Autonomous but not connected ships rely only on local intelligence without cooperating with others, leading to some problems. Therefore, to guarantee the viability of a fleet of ships, autonomous ships should be equipped with more advanced connectivity capabilities. This section defines the cooperative and connected vehicles concept. Then, we highlight the fundamental technologies for cooperative and connected autonomous inland ships in particle. Finally, we review the existing efforts related to Connected and Cooperative Inland Ships (C-CAIS), their opportunities, and challenges.

2.3.1 What is a Cooperative and Connected Vehicle?

Human drivers can communicate with pedestrians by expression in their eyes and gestures, through which both drivers and pedestrians can know who will go first. Automated vehicles are, however, incapable of intentional communication, notwithstanding sensors.

2.3.1.1 What is a Connected Vehicle?

Nowadays, wireless communication technologies are applied in different areas of daily life. Vehicles are equipped with wireless communication devices, enabling them to communicate with other vehicles, with no need to guess their intentions. According to the Center for Advanced Automotive Technology [108], a Connected Vehicle (CV) is a vehicle with vehicle connectivity, which can receive beyond field-of-view information from Vehicle-to-Everything (V2X) communication and interact with other road users. Specifically, it uses any of diverse different communication technologies to communicate with the driver, other vehicles on the road (Vehicle-to-Vehicle (V2V)), roadside infrastructure (Vehicle-to-Infrastructure (V2I)), pedestrians (Vehicle-to-Pedestrians (V2P)), and the Cloud (Vehicle-to-Cloud (V2C)). Associated with the component of intelligent transportation systems, CVs form the Internet of Vehicles (IoV) [5]. The latter is a distributed network that supports data created by connected mobile vehicles that can interact to establish a network. Communication offers new opportunities for developing new applications for vehicles, such as remote door locking, stolen vehicle detection, and car-sharing. Connected vehicles are already available in some vehicles in countries such as USA and Japan. For example, in Japan, the Toyota Prius [109] provides drivers with: a right turn collision warning, a red light caution, a traffic signal advisory change, and an emergency vehicle notification.

Moreover, this concept may be extended by adopting Connected Autonomous Vehicles (CAVs), and thus, vehicular communications become the Internet of Autonomous Vehicles (IoAV) [110]. Equipped with onboard sensors and V2X communication devices, CAVs can expand their horizon and substantially improve their perception capabilities, as shown in Figure 2.7. Simultaneous internal perception and external communication can minimize blind zones of AVs and then accurately perceive the situation to make correct interpreta-

tions and decisions. Hence, CAVs can achieve better safety and performance than AVs and improve vehicle efficiency and commute times.

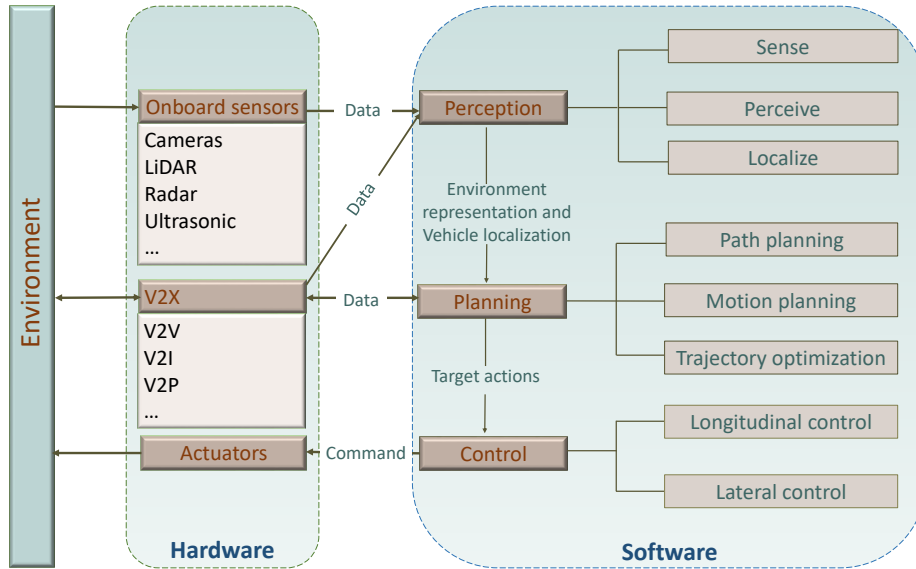


Figure 2.7: Standard components of a connected autonomous vehicle pipeline listing the various tasks.

2.3.1.2 What is a Cooperative Vehicle?

AV and CV technologies have inherent shortcomings [111]. The limitations of AV sensing technologies are highlighted by the fatal accident of Tesla car with autopilot in 2016 and Uber self-driving car in 2018. In both accidents, AVs’ automated driving systems failed to monitor the driving environment and identify imminent collisions dangers in the way they were supposed to. This raised significant concerns about the reliability of autonomous vehicle technology under extreme weather or road conditions. In addition, ADS models used for AVs are mainly operated in a black box without clear explanation and transparency. On the other hand, CVs use the message exchange only to build mutual awareness and do not benefit from connectivity to perform other tasks. In reality, specific issues cannot be resolved by individual CVs but rather by cooperation between them. For instance, suppose several vehicles are driving at similar lower speeds and occupying all lanes. It may be impossible for higher speed vehicles approaching from behind to overtake them, causing unnecessary traffic congestion. In such a case, it would be effective for these vehicles to cooperate and resolve this issue through agreed adjustments.

Given the limitations of CAV and AV, there are increasing research and development interests in Cooperative Connected Autonomous Vehicles (C-CAV) technology. C-CAVs are an autonomous association of vehicles by which each vehicle communicates wirelessly with other road users with the ultimate aim of achieving benefits for many areas of traffic management and road safety. The basic idea is that vehicles are equipped with onboard units,

routers, and antennas: thus, they can receive information from roadside infrastructure, process information, display information to the driver (or passengers on public transport), and communicate information with other vehicles or with roadside infrastructure fitted with the right technology. Furthermore, C-CAVs may negotiate driving trajectories to their mutual benefit and to the benefit of overall traffic flow and safety.

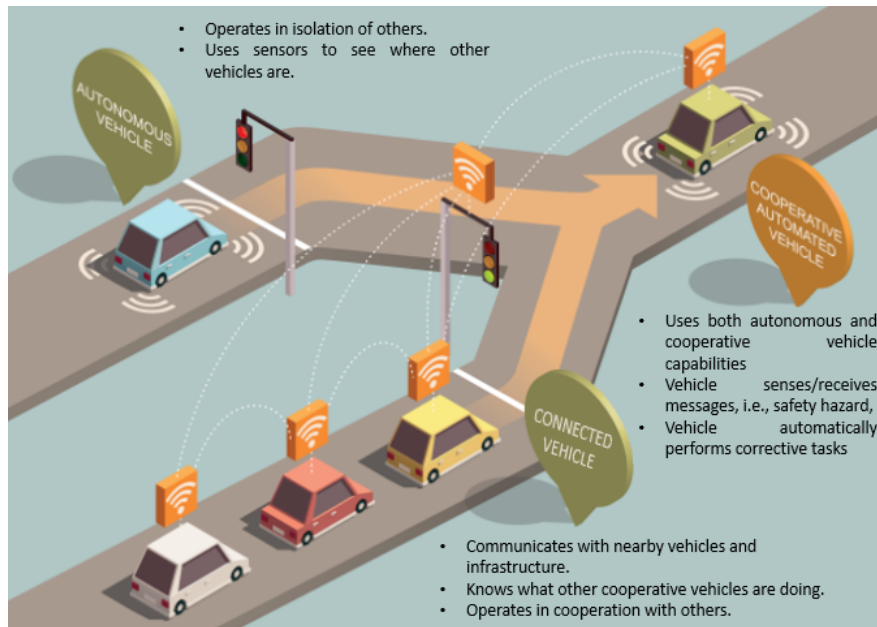


Figure 2.8: Journey to Cooperative Automated Vehicles technology.

2.3.1.3 C-CAVs Use Case Applications

Numerous standards are used across all different types of transportation, such as vehicle safety standards, road standards, and rail standards. In particular, Intelligent Transportation Systems (ITSs) standards [112] [113] define how ITS systems, products, and components can interconnect, exchange information, and interact to deliver services within a transportation network. They support multiple applications; some are illustrated in Figure 2.9:

- Cooperative lane changing (Figure 2.9a): With the development of V2V communication, sharing information among multiple vehicles is possible. Cooperative lane changes can improve safety and lane-change efficiency.
- Cooperative merging (Figure 2.9b): As traffic demands increase rapidly on highways, effective merging strategies based on cooperation between autonomous vehicles can significantly elevate road safety and improve traffic operations.
- Cooperative overtaking (Figure 2.9c): Cooperative overtaking can improve overtaking through the real-time information exchange between traffic participants, including road infrastructures, nearby vehicles, and others.

2.3.2 Underlying Technologies for Cooperative and Connected Autonomous Inland Ships

Achieving efficient Cooperative and Connected Autonomous Inland Ships relies on some existing technologies. We define in this section the necessary technologies enabling their deployment, including the communication systems, the communication architectures, and the different types of communication.

2.3.2.1 Communication Systems

Connected ships can adopt different communication technologies according to their different requirements, and their location [114]. In addition to satellite systems traditionally considered for maritime ships connectivity, the terrestrial solutions, described in Table 2.6, should be available when the vessel is close to shore. Multihop features, cellular technologies, and long-range terrestrial systems with end-to-end resource management in critical data communications are useful technologies for connectivity.

| | IEEE 802.11p for ITS | WiFi | LTE/4G | 5G millimeter wave (mmW) | VHF digital radio | HF |
|---------------------|----------------------|---|--|---|--|-------------------------|
| Spectrum | 5.9 GHz | 2.4/5GHz | 450 MHz-3.7 GHz | 24-86 GHz | 30-300 MHz | 3-30 MHz |
| Bandwidth | 10 MHz | 20/40 MHz | from 1.4 MHz to 20MHz | up to fewGHz | 25 kHz channels, can be bundled together e.g., to 100 kHz | up to 48 kHz |
| Max bit rate | 27 Mbps | 600 Mbps | 75/300 Mbps forUL/DL | up to 20 Gbps | VDES: up to 307 kbps in ship-to-ship or ship-to-shore, 240 kbps for satellite link | up to 240 kbps |
| Tx range | < 1 km | typically < 100 m, up to 10 km with fixed service | typically < 2 km up to 70 km with directional antennas | < 10 m for 60 GHz WiFi, tens of kilometres with fixed links | up to 85 km | Thousands of kilometres |
| Cost | Cheap | Cheap | Expensive | Cheap (WiFi) Expensive (Cellular) | Cheap | Cheap |

Table 2.6: Comparison of terrestrial communication systems for autonomous ships

Moreover, currently, the 3rd Generation Partnership Project (3GPP) is considering and developing a system specifically for maritime communications to support the needs of future maritime users [115]. One of the requirements of this “cellular-Maritime” system is to support up to 100 km coverage. It will also support the interworking between the 3GPP system and the existing/future maritime communication systems for seamless data exchange between users ashore and at sea or between vessels. Thus, the 5G system will support service continuity for maritime users between land-based 5G access and satellite-based access networks owned by the same operator or by an agreement between operators [116]. Motivated by this advancement, we introduce the C-IAShips Architecture in Chapter 5.

2.3.2.2 Communication Architectures

The IoS enables the monitoring of vessels and onboard equipment in real-time. A typical IoS structure is shown in Figure 2.10. It consists of vessels, island base stations, a climatic observation system, communication satellites, buoys, wireless transmitting and receiving devices, and the command system [117].

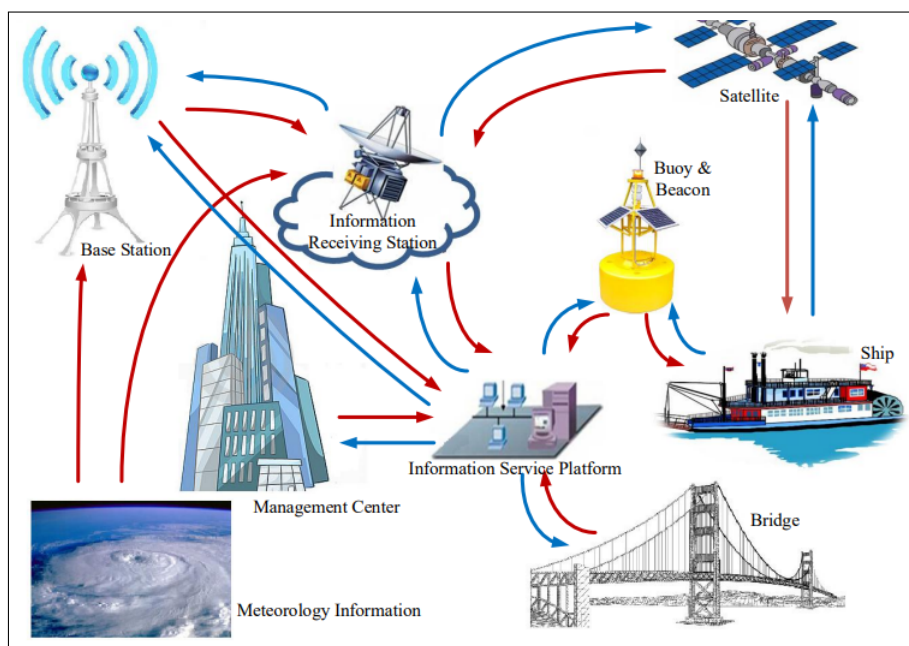


Figure 2.10: A typical IoS structure (adapted from [117])

In recent literature, various architectures have been proposed for adopting IoT in the maritime industry. We compile a comprehensive IoS architecture consisting of five layers: Perception, Network, Data Resource, Applications, and Exhibition, as shown in Figure 2.11.

- ❖ **Sensing layer:** The first layer is responsible for collecting data from various sources located either on ships or at the shore. For instance, ship heading, ship position, ship speed, water level, traffic information, and bridge capacity information are some examples of data collected through the sensing layer.
- ❖ **Network layer:** The network layer provides communication among the various devices/objects part of the IoS system and comprises different network technologies. This layer aims to guarantee a successful and smooth data connection and reliable data transmission.
- ❖ **Data resource layer:** This layer is responsible for storing and managing the collected data. At the same time, this layer provides information to the application layer. Several databases and object stores in the data resource layer must fulfill various requirements, including uniform database naming, management, and uniform information coding [118]. Furthermore, to handle the wide variety of data, this layer must support current mainstream data formats such as XML, JSON, CSV, and binary. New big data

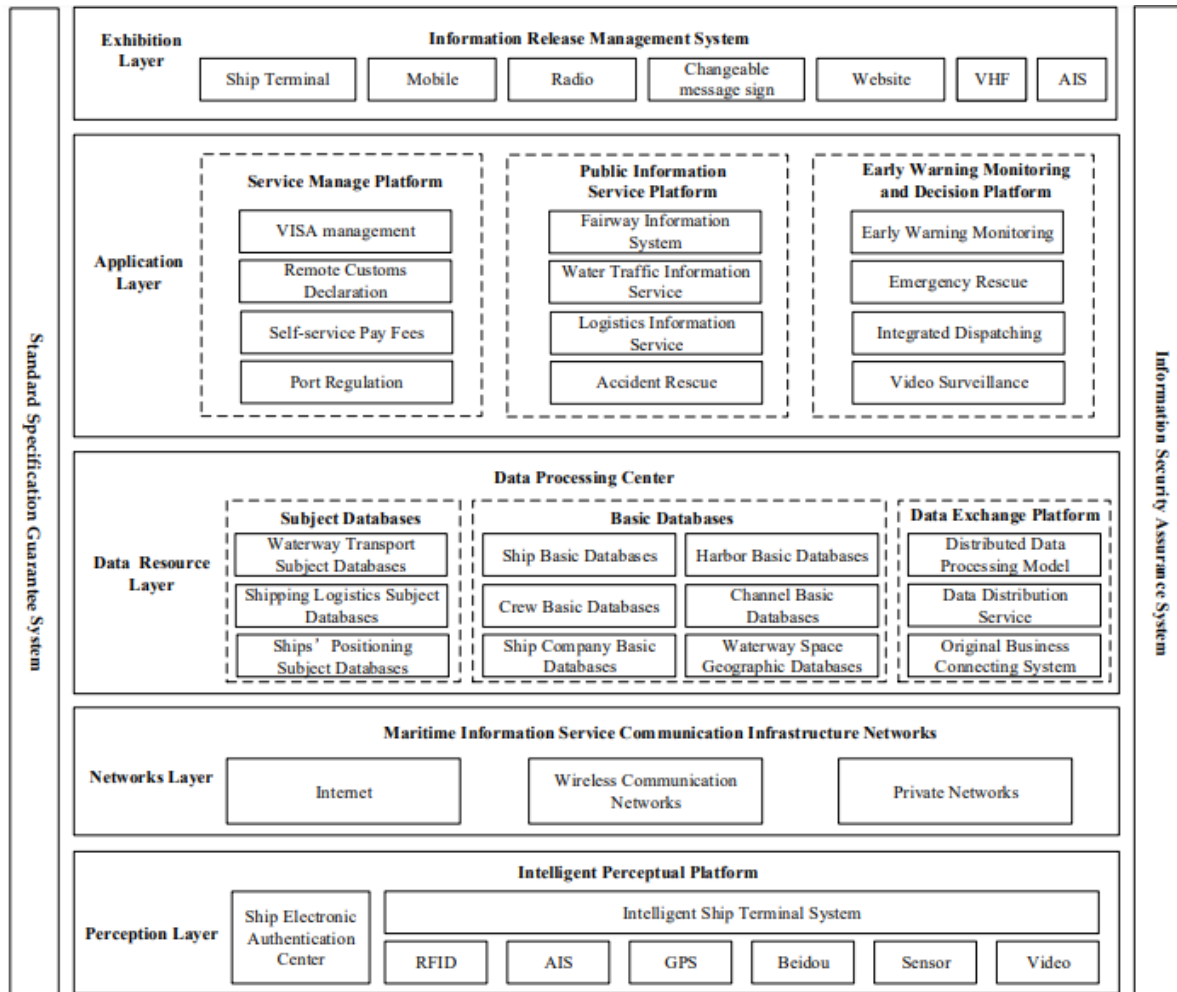


Figure 2.11: Overall architecture of IoS

platforms such as Hadoop, Spark, Kafka, and MongoDB can also store and process a tremendous amount of collected data.

- ❖ **Application layer:** This layer focuses on developing applications and services that meet customer requirements. Customers can choose the service or application from the services list, and then the selected service is offered through related resources [118]. For instance, safety enhancement, route planning, real-time cargo monitoring, fault detection/prevention, and automatic berthing are the essential services this layer provides.
- ❖ **Exhibition layer:** Several application systems are developed to utilize resources from the data resource layer to analyze, calculate, and process data in the application layer. Consequently, the applications are used to meet the requirements and demands of different maritime customers. On top of that, the exhibition layer works as a service window for communicating with customers in real-time [119]. For instance, smartphones, intelligent terminals, websites, and social media are used for information sharing with maritime customers.

2.3.2.3 Ship-2-Everything (S2X) Communications

Traditional ship communication system only relies on AIS to provide low data services such as position, course, heading, destination, tonnage, speed, etc.

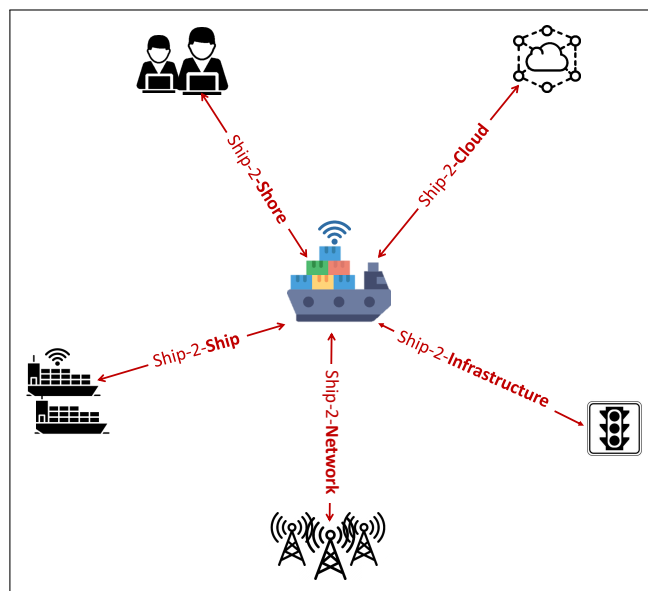


Figure 2.12: Ship-2-Everything (S2X) Communications

- ❖ **Ship-2-Shore (S2Sh) communication:** The communication system sends the ship's position, course, speed, and ship status flags in remote monitoring. In the process of status investigation, SCC can query the detailed status information of the ship and the surrounding situation. Hence, the communication system needs to transmit radar images, IR or video images, HDTV images, and automation. This allows the SCCs to view unmanned ship operations' real-time data and historical data and make decisions based on the data.
- ❖ **Ship-2-Ship (S2S) communication:** Similar to the V2V communication [120], ships can communicate. Ship-2-Ship communications use on-board dedicated short-range radio communication devices to transmit messages about a ship's status and receive the same information from the other ships. In this manner, they can infer the relative position and the speed direction. Besides, some researchers introduce the Sea Ad hoc NETWORKs (SANETs) [121], a bracket of communication ships that can compose and sustain a network among themselves without the help of a main leading administrator.
- ❖ **Ship-2-Cloud (S2C) communication:** More shipping companies [122] [123] are now willing to use Cloud computing-based systems to manage the IT environment securely and get access to the most critical information, even with limited connectivity. For IoS, the concepts of distribution of integrated resources and integration of distributed resources are expected to be very important. The maritime cloud [124] holds maritime identities and provides basic authentication, integrity, and confidentiality methods.

- ❖ **Ship-2-Infrastructure (S2I) communication:** Vehicle-to-Infrastructure (V2I) communication is the wireless exchange of data between vehicles and road infrastructure. Similarly, ships need to communicate with the river infrastructure surrounding them through S2I communication. On the one hand, S2I aids in minimizing the time that vessels need to pass through infrastructures, such as a lock, a movable bridge, an intersection, and a terminal. On the other hand, infrastructure components such as road signs and traffic lights can wirelessly provide information to the ship, which can help control ships' traffic in emergencies, such as ambulances, fire brigades, or police.
- ❖ **Ship-2-Network (S2N) communication:** S2N represents the connectivity between the ship and a network operator providing access to the ship. This communication can provide real-time information and cloud-based services that improve road safety, providing immense guidance regarding waterway updates and serving as a hotspot for internet connectivity.

However, although adding connectivity to vehicles has its benefits, it also has challenges. By adding connectivity, there can be issues with security, privacy, data analytics, and aggregation due to the large volume of information being accessed and shared.

2.3.3 Opportunities and Challenges for Connected and Cooperative Inland Ships

Autonomous vessels operating in inland waterways need to perform reliable navigation and interact safely and with operated vessels. Therefore, some challenges related to the inland domain characteristics should be considered. In the next section, we discuss such challenges related to the Internet of Inland Ships.

2.3.3.1 Inland Domain Characteristics Related to IoIS

There are certainly similarities between the Internet of Inland Ships and its ground-based counterparts, the Internet of Ships, and the Internet of Vehicles in general, such as the interconnection of smart devices and standard architectural components and services. However, there also exist some critical characteristics of the fluvial domain that differentiate IoIS from IoS and IoV, as summarized below:

- ❖ **Big and heterogeneous data:** In the IoIS system, the objects of the information collection are bridges, ships, ship locks, channels, meteorology, hydrology, etc [125]. All the information is in static and dynamic states. Due to the variety of information sources, the data formats of different sources are quite different. A clear and present challenge that faces C-CISs is the high degree of heterogeneity of the devices in terms of communication capabilities.
- ❖ **Hard data collection:** Due to the frequent movement of ships, data gathered in the shipping sector may be incomplete, inaccurate, or unreliable at specific times or particular locations. For example, ships are not always connected to provide real-time

information, while sometimes data may be lost or delayed due to poor internet connection. These facts often create problems for quick and intelligent decision-making in the maritime industry. New data collection technologies must be integrated into the shipping industry to cope with this crucial challenge. Moreover, the automation of data capturing is also required to reduce manual data entry.

- ❖ **Capacity and Scalability:** The foreseeable demand to handle the ever-growing inland traffic will stress the IoIS system and drive higher capacity. Efficiency is vital to maximize the system capacity constrained by the highly scarce communication resources. Physical and higher layers of the IoIS system are hence to be optimized to enable spectrally efficient communications. In addition, the system must be scalable with future growth when resources are added in response to the growing demand for capacity and increasing bandwidth needs.
- ❖ **Safety and security considerations:** Shipping is considered a safety-critical application but currently lacks a standardized approach to Cyber-security [6]. Since IoIS connects several geo-distributed objects, such as ships, locks, warehouses, and industrial systems, from different vendors, building an efficient and secure communication network in this environment is not easy. Unauthorized access of IoIS objects/data can cause dangerous results, even loss of lives, so principles of safety-critical systems must be included, and stricter criteria must be enforced. Furthermore, C-CISs are expected to face a high information security risk due to their open distributed network environment. The revealing of location information results in significant privacy concerns, making location privacy a significantly challenging task.

2.3.3.2 Existing Initiatives Related to C-CAISs

A considerable literature has recently grown up on the theme of ship wireless communications. However, much of the research up to now has been descriptive in the offshore area. There has been little quantitative analysis of wireless communication in inland waterways. Although the term of Internet of Inland Ships was first used in 2013 [126], only the effects on the inland river environment, the RIS rules meeting by establishing an exhaustive architecture have been examined since. On the other hand, only a few studies focus on the cooperative control of vessels in inland waterways. We can find mainly cooperative control in ports, cooperative waterway intersection scheduling, and cooperative collision avoidance.

2.3.3.2.1 RIS Rules Exploration Previous works mainly explore the basic rules of RIS and its main opportunities and concern with manned ships while considering the specific character of inland navigation and the topography of inland waterways. As mentioned earlier, the RIS system aims to unify the technical and legal standards and realize the inland ship's collaboration and normalization. In [127], the authors use 3G wireless, Radio Frequency Identification (RFID), and ZigBee to establish a dynamic ship management system for inland waterways. They claim that their proposed system would facilitate the safety and efficiency of inland shipping and also the sustainable development of the Chinese shipping industry. However, they explain only how their techniques will be employed

and do not provide explicit details about their global architecture. To optimize the use of the Guadalquivir river and improve the control of ships' navigation, [128] proposes a centralized Cloud-based integration architecture and a Web Portal. It claims to exchange data between internal and external sources in real-time while providing data storage and management security. The collected data is displayed in a web portal, presenting dashboards of real-time information to the I-SCC, providing notifications of obstacles, and managing the ships with high-precision data. However, the researchers focus only on the development and design of the architecture but never validate it through simulations or analysis.

2.3.3.2.2 Overall Architecture In [129], the authors proposed a novel design of the Inland Shipping Management Information System (ISMIS), based on IoT techniques, for improving the inland shipping management level. The proposed design is based on a wireless sensor network to supervise objects dynamically with the help of cloud computing and RFID middleware technology.

2.3.3.2.3 Information Processing and Information Fusion To deal with the big and heterogeneous data issue mentioned earlier, researchers use information fusion as the most critical stage of intelligent information processing. Information fusion is a multi-level, multi-aspect information process including multi-source data detection, correlation, combination, and estimation. The distributing structure of the information fusion technology preprocesses the sounding data to generate tracking information and then makes a global judgment in the center. The work [126] describes a detailed case of the information fusion between the Vessels Traffic System (VTS) and the AIS. In the VTS system, the radar is the main facility to collect information and data. Therefore, most characteristics of the VTS system are similar to that of the radar. Generally, the VTS system has accuracy in monitoring and tracking the ship moving status information. In addition, it has an excellent extension to fuse much heterogeneous information. The AIS system can get the ships' dynamic information, including location, speed, and navigational status. As a result, the information fusion technology could fuse the advantages of these two different systems in the fusion center. The AIS system could help improve the location information accuracy of the VTS system. In contrast, the VTS system uses its advantages to capture and track dynamic information. With the fusion of related trajectory data, the administration could get more stable and more precise ship position information and movement information.

2.3.3.2.4 Cooperative Control in ports Some research concentrates on the Inter Terminal Transport (ITT) in a port regarded as a small waterway network [130]. ITT refers to the transportation of goods between terminals within a port. In [131], a fleet of inland Automated Guided Vehicles (AGVs) are used to handle a set of ITT requests for the advantages like handling the expected large throughput instead of exploiting limited land, energy saving for terminals with longer distances by land than by water, etc. A closed-loop scheduling and control approach is proposed: by solving a pick-up and delivery problem, a sequence of terminals to visit for each waterborne AGV is generated; a cooperative distributed model, predictive control method, is applied to control the waterborne AGVs to execute the schedules. The other type of research on cooperation in ports is the Vessel Rotation Planning

Problem (VRPP) i.e., which decides the sequence of multiple terminals that an inland vessel visits in the port area. In [132], the VRPP is firstly proposed, in which the terminal and vessel operators cooperate to obtain better alignment. In [133], the authors compare four approaches to solve the VRPP, which concerns deciding on the optimal sequence of vessel visits to different terminals in a large seaport.

2.3.3.2.5 Cooperative Waterway Intersection Scheduling Vessels passing through intersections are comparable to the situation of vehicles crossing non-signalized intersections. In road transport, intersection crossing is one of the most challenging problems and attracts much attention. In [134], a multi-layer framework is employed to improve the efficiency of transport in a canal network. The cooperation among vessel controllers and intersection controllers is achieved through iterative negotiations. More specifically, the authors propose a framework for the cooperative control of vessels in waterway networks with two types of controllers: a Vessel Controller (VC) for controlling an individual USV and an Intersection Controller (IC) for solving the conflicts of vessels at an intersection. A vessel controller uses sensors to get self-state information and information on obstacles. The navigation system creates pictures of the current situation and informs the Guidance system of collision risks based on the obtained information. Combining with the predetermined global path, optimal trajectories with specified objectives and constraints can be determined. The commands are then sent to actuators for autonomous navigation.

2.3.3.2.6 Cooperative Collision Avoidance Traditional collision avoidance methods consider that each ship tries to predict other ships' actions, either by assuming that others have a constant speed [135] or according to holonomic or kinematic models [136]. Then, they allow ships to cooperate only for a limited time, for instance, when they identify a high collision risk. One method widely used in the literature relies on the geometrical characteristics of the traffic circumstances among vessels. It assumes that a surrounding region should be kept clear between ships and other objects, and it focuses mainly on ships' size and shapes under stable conditions [137]. Gang et al. [138] build a model estimating the Collision Risk Index (CRI) of encountered vessels based on the SVM with the fuzzy comprehensive evaluation. Another group of methods aims to collect a set of the ships' speeds or courses leading to collisions with other ships and present this set in figures to the officers. A collision alarm is triggered when the current velocity of the ship is inside of this set. However, most studies focused only on the collision risk between Own Ship (OS) and Target Ship (TS). Though [139] propose the collision risk assessing models related to multivessel tracking, only the collision risks between any two vessels in the multivessel encountering are investigated, which, nonetheless, ignores the global multivessel collision risks. As a result, it may be hard for surveillance operators to comprehensively understand the collision risk, especially multivessel collision risk in multivessel encountering. Another limitation of existing works is their unrealistic assumption that all the environment situations and information are perfectly observed by each ship. In real-world circumstances, each ship has to observe and sense the situation from its angle. The capability of sensing and detection is subject to the navigational equipment on the individual ships, which will raise awareness beyond the consideration of prior work. Thus, we introduce a novel cooperative collision detection system in Chapter 6.

2.4 Conclusion

We showed throughout this chapter that the advancement in cooperative and connected autonomous inland ships has been proliferating in the last years. They have proven to be an exciting asset for various applications with the aim to enhance the inland shipping. However, this nascent technology has not yet reached maturity; therefore, it still faces numerous challenges and issues that need to be addressed before an efficient and safe deployment. Throughout this chapter, we surveyed a comprehensive overview of the cooperative and connected autonomous inland ships by presenting the different challenges related to their continuous progressing. We first conducted a comprehensive overview of autonomous inland ships while discussing their main enabling competencies and the different challenges and concerns facing their development. Then, we highlighted several contributions from the literature working on this issue. Then, we surveyed on cooperative and connected inland ships. We listed not only technical issues, but also communication and networking challenges to enable ships to connect and cooperate. Then, we presented a detailed review of existing solutions.

PART I

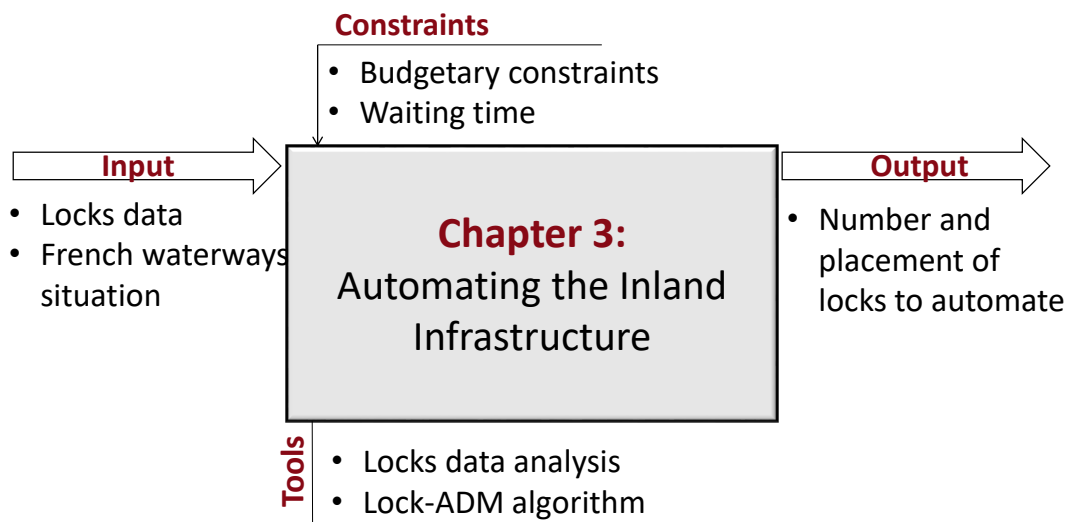
AUTOMATION

The previous chapter has given a frame of research initiatives in various fields contributing to the conception of cooperative and connected autonomous inland ships. In this part, we focus on the fundamental essence of this advancement by studying the feasibility of automation in the fluvial environment.

AUTOMATING THE INLAND INFRASTRUCTURE

Everything in life goes back to the basics.

– Kron Gracie



| | | |
|-------|--|----|
| 3.1 | Introduction | 49 |
| 3.2 | French Waterways Situation Analysis | 50 |
| 3.2.1 | River Network | 50 |
| 3.2.2 | Traffic Density | 50 |
| 3.2.3 | Passage Scheduling | 51 |
| 3.2.4 | Opening Time | 52 |
| 3.2.5 | Does Automation Have an Impact on Traffic Density? | 52 |
| 3.3 | Locks Automation Related Works | 53 |
| 3.4 | Problem Formulation and Proposed Solution | 54 |
| 3.5 | Lock-ADM: Optimal Number of Locks to Automate and Their Placement | 55 |
| 3.5.1 | Lock-ADM - Stage 1: Calculate the Optimal Number of Locks to Automate | 55 |
| 3.5.2 | Lock-ADM - Stage 2: Order Locks According to their Importance Score | 57 |
| 3.5.3 | Lock-ADM - Stage3: Select Best Locks to Automate by Studying their Impact on the Network | 60 |
| 3.6 | Implementation and Results | 64 |
| 3.6.1 | Locks Importance Score Calculation (Stage 2) | 64 |
| 3.6.2 | Metaheuristics Comparison (Stage 3) | 67 |
| 3.6.3 | Comparison with Other Algorithms | 67 |
| 3.6.4 | Impact of the Population's Initialization | 68 |
| 3.7 | Conclusion | 69 |

3.1 Introduction

Different kinds of issues that are not directly related to the autonomy of ships are yet to be overcome, such as infrastructure automation. As already mentioned in the motivation part of the previous chapter, although river transport is competitive, it represents only a small part of national transport in France, even though France has the most extensive network in Europe with over 8,500 kilometers of waterways. This is mainly due to the lack of flexibility in its infrastructure compared to road transports. Locks are the oldest and most used waterway structures that enable ships to surmount water level differences safely [140]. However, they generate an enormous waiting time: for example, on the path between Lyon and Paris, ships have to cross 220 locks. The crossing represents almost half of the total travel time [141]. Long waiting time increases the duration of the trip and, consequently, the total cost, which, combined with fuel consumption and tolls, contributes to the crossing's overall burden. Additionally, the delay sometimes becomes unacceptable, especially for ships with tight deadlines. To handle this problem, three solutions were proposed in the literature: (i) adding water-land transshipment as a parallel channel to the lock ([142, 143]), (ii) expanding the existing locks or building ship elevators ([144, 145]), (iii) or enhancing the efficiency of operation of existing locks ([146, 147, 148]).

In the first solution, arriving ships can pass through the lock or unload their cargo at the quay, and the other modes of transport deliver it to their final destination. This technique does not solve the problem; instead, it transfers it to a different transport mode, which is not optimal for previously explained reasons. Additionally, it creates another challenging problem: the co-scheduling between the lock and quay integrated system. The second solution is efficient because it attempts to address the source of the problem, but it is time-consuming and costly. Although the delay of ships can be decreased by scheduling locks algorithmically, the effect of the third solution is limited, especially when the locks have many constraints related to external factors such as lock-keepers and opening times. Therefore, this thesis focuses on combining the two last solutions by modernizing only important locks and scheduling the crossing via a proactive real-time method to minimize the ship's waiting time. We introduce in this chapter the first approach and the second approach is discussed in Chapter 7.

We found in the literature some works that focus on increasing the efficiency of specific lock sequences on various waterways around the world, such as the Kiel Canal [149] (98 km long with two locks), the Welland Canal [150] (43 km long with eight locks), and the upper Mississippi [151] (2092.147 km long with 29 locks). However, none considered a network of inland waterways in its entirety, which makes their solutions irrelevant when considering the vast French waterways network. Reducing waiting time by locks automation is complicated to consider, but this chapter makes a first step towards incorporating it while considering the French waterways' specificities. We illustrate three main contributions in this study, as follows:

- We analyze the french waterways' current situation to discover the river network specificities and its current limitations and constraints. We use real data collected from the official Voies Navigables de France (VNF) portal.
- We propose a new formulation of the lock automation problem while based on the

knowledge acquired from the first analysis. Thoroughly, we introduce the Lock-ADM algorithm to find the optimal number of locks to automate and their placement.

- We validate the effectiveness and efficiency of our algorithm through extensive simulation results. We analyze these results and suggest possible improvements.

The rest of this chapter is structured as follows: Section 3.2 details and analyzes the French waterways' current situation. We investigate the actual locks states' impact on the travel time to identify the possible improvements to be implemented. Section 3.3 provides background information on the automation of locks and reviews related literature. In Section 3.5, the decision-support system for selecting the best locks' location to be automated, Lock-ADM, is introduced. We show and analyze the simulation results in Section 3.6. Conclusions and potential future works are discussed in Section 3.7.

3.2 French Waterways Situation Analysis

This section studies the current locks states impact on the travel time to identify and understand the specific issues to be addressed. We begin by studying the French river network's topological components. Then, we analyze traffic density behavior and investigate the current passage-scheduling strategy through the locks and their opening time. Finally, we show how locks automation can enhance traffic density performances.

3.2.1 River Network

A river network is a realization of a spatial network describing the structure that permits the inland ship movement. The French river network is composed of 1,632 locks. In Figure 3.1a, we draw the French navigable waterways and their characteristics, while in Fig 3.1b, we collect from Google Maps the accurate positions of the locks in the North-Eastern quarter of France to provide a zoomed view on the features of the distribution of locks. Figure 3.1a shows that there are no isolated locks, and each lock has at least one path to all other nodes. Furthermore, we distinguish in red in Figure 3.1b, the locks' presence with a much higher number of connections than the other locks.

3.2.2 Traffic Density

Traffic density is an approach to study traffic flow and predict its behavior in the future [152]. Here, we define traffic density as the number of ships passing through a lock. The data needed for this study are collected from the lock operators' files on the official Voies Navigables de France (VNF) portal [153]. We collect existing data from January 2005 to September 2019 to analyze the traffic density through all existing locks. We then store it in Comma-Separated Values (CSV) files for each lock. As the data was entered manually by the lock-keepers, we found irrelevant and missing parts. Therefore, we proceed to replace the missing values with approximate values. We calculate the average of neighboring values

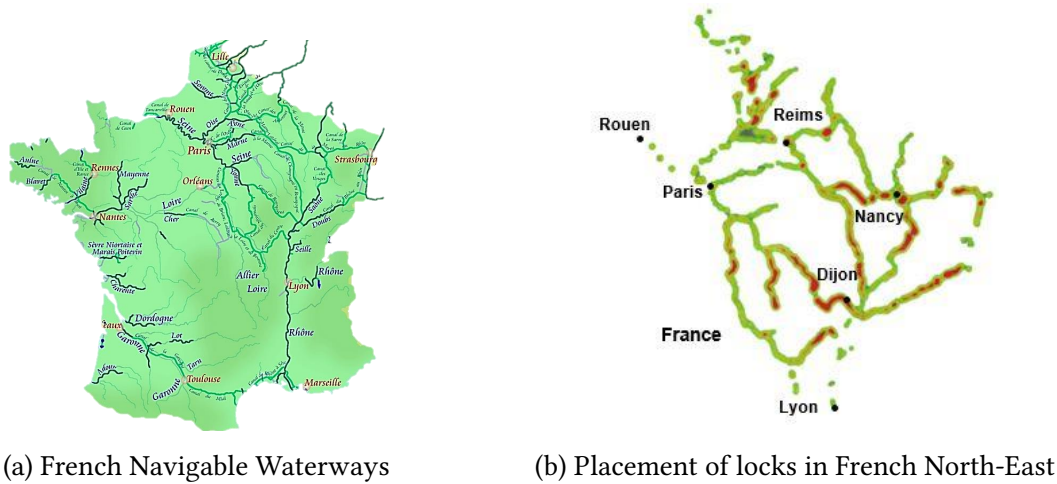


Figure 3.1: The actual French Navigable Waterways map.

and replace the missing values with them. Besides, we remove from the analysis the locks with missing successive rows higher than one year. Figure 3.2 plots the distribution of different traffic densities for different locks. We can deduce that locks do not all have the

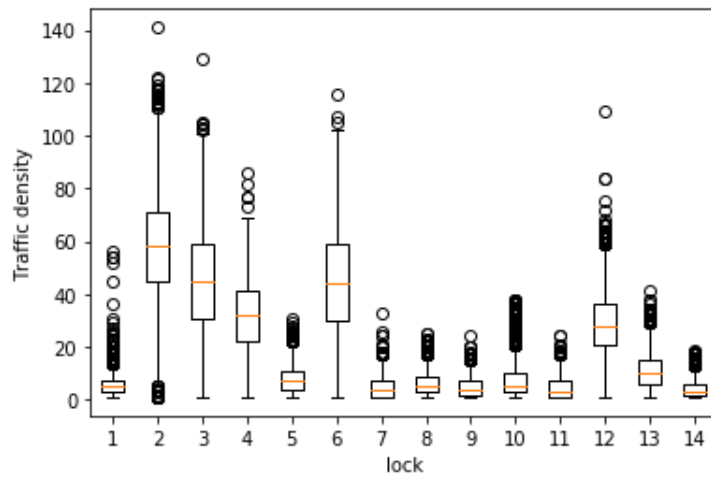


Figure 3.2: Different traffic densities for different locks

same properties in the river network. This difference can cause congestion at some points, which leads to extra waiting time to cross locks. To handle this issue, a good passage-scheduling strategy should be settled.

3.2.3 Passage Scheduling

Current passage through locks is managed in order of priority [154]: ships with urgent service reasons first, then freight ships with authorization, and in third place, the other categories of ships pass according to their order of arrival. This traditional scheduling strategy involving some priority-based scheduling is known as First Come First Served (FCFS). By

comparing the performance of various control policies, several researchers in similar domains ([155, 156, 157]) prove that the FCFS policy does not ensure better time management because it prioritizes equality rather than efficiency. However, overall efficiency and transportation safety are sometimes sacrificed. Solving this problem may lead to win-win solutions since reducing ships' waiting time yields to economic, operational, and environmental savings for container terminals and shipping companies.

3.2.4 Opening Time

Another cause of extra waiting time in locks is related to their opening time. In fact, as lock-keepers manage most of the locks, passage through the locks is limited by each one's opening hours:

- Type 1: open from 8am to 6pm with a midday break from 12 to 2pm.
- Type 2: open from 7am to 7pm.

The locks do not operate on public holidays, and there are also occasional changes in opening hours. Additionally, on the entire river network, all commercial ships must announce their entry into the network and they must book the lockage service within fixed timetables by calling the customer-user relations department. All these constraints generate additional waiting time. Lock automation may be an efficient solution to reduce waiting time, but it also needs to be beneficial for ship management companies.

3.2.5 Does Automation Have an Impact on Traffic Density?

To study the impact of automatizing a lock, we collect the data of some automatized locks. Then, we calculate the difference between traffic density before and after the automation. Figure 3.3 illustrates some useful information about the traffic density one year before and after the automation of one lock. Figure 3.3 shows that locks' automation has advantages

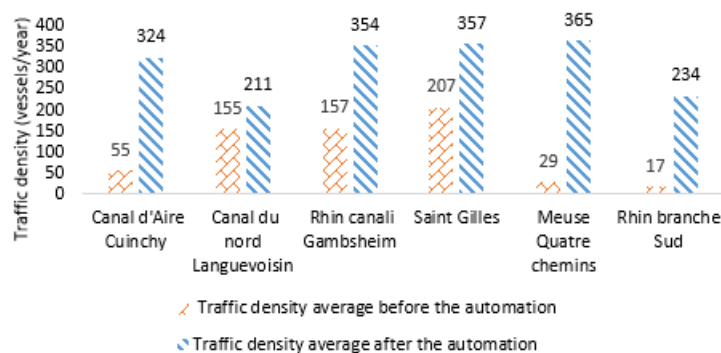


Figure 3.3: Illustration of six different locks before and after the automation

in traffic density and, therefore, toll taxes. However, this benefit varies from one lock to

another (an increase of only 56 passages for Languevoisin lock and an increase of 336 passages for Meuse Quatre Chemins lock) with an average increase of 204 crossings per lock per year. We notice that locks with similar traffic densities will not have the same traffic density after automation (Canal du nord Languevoisin and Rhin canalli Gamsheim). It is therefore necessary to make a study before deciding which locks to automate. In the next sections, we will explain the stages to make such a decision.

3.3 Locks Automation Related Works

Optimizing the operational management of the lock system offers significant benefits. First, automation measures and, inevitably, operational management of the transport conveyor of lock system can reduce the number of locks operations and transportation costs. This will save energy and time resources. Second, in a high flow of ships, the automation of locks will increase the capacity of the canals and relieve them. However, there is very little work in the literature on the automation of waterways. Different types of automation are possible such as ships automation or waterways infrastructure automation. Inland waterway infrastructure includes locks and dams as well as navigation channels. Since we are interested in improving the waterway navigation by expanding the existing infrastructure, locks automation will be the primary focus of this study.

Ardavan [158] points out that the development of inland ports increases transportation efficiency since it creates a smoother flow on the highways, creates a cleaner environment, increases the capacity of the ports, reduces the demands on portland, and promotes inland economic and logistics integration. Gervais et al. [159] propose a linear programming model to assess the short-term economic benefits of enlarging five 600-foot locks on the Upper Mississippi River (UMR) to 1,200 feet. They estimate the total costs of lock enlargements to 4 cents a bushel and the total benefits to 0.3 cents per bushel. These results suggest that large-scale improvements on the UMR are not warranted since only small economic benefits are redistributed to grain industry groups. However, this judgment cannot be generalized since they only study the short-term impact and focus on a single region. In addition, they do not take into account the potential savings to vessels by reducing the waiting time at the five locks. Thus, a more comprehensive analysis would undoubtedly increase the total benefits. Similarly, Wilson, Dahl, and Taylor [144] study the impact of expanding the capacity of a lock on the delay costs for grain shipped on the UMR. They show that increased lock capacity reduces delay costs and significantly increases barge shipments. However, the authors do not explain why they chose only this specific lock. They do not study its specificities and importance among others and do not consider the impact of these improvements on the entire inland waterway network.

We can also cite [145], which aims to identify the set of maintenance projects to fund to minimize the total shipping costs on the river networks. The proposed model effectively makes meaningful recommendations regarding the maintenance project funding. However, they do not discuss the importance of existing projects, so unimportant and cheap projects can be funded while essential and expensive projects will not. Moreover, it considers five levels of budget allocation that means it can allocate only a percentage to one project. However, they do not ensure its continuity over the following years unless the same project requests an allocation again.

In summary, this literature review indicates research directed toward the cost-benefit analysis of lock infrastructure improvements with a lack of generalization and in-depth real data analysis. The objective is to fill this literature gap, laying on the French waterways analysis presented in Section 3.2 to decide on the relevant locks to automate.

One closely related literature is the positioning problem, which intends to find the optimal positions for placing an infrastructure such as antennas, sensors, or drones. However, these works do not consider the fixed structural constraints imposed by the existing network. Therefore, their methods are not suitable for our work since the locks are already installed. We aim to automate an optimal number of existing locks to bring more gains to the whole network. The most influential nodes discovery and selection area of research can also be somehow similar to our problem. However, in the literature, they use different metrics and measures which are not adaptable to our problem. For instance, to detect keywords in documents such as PageRank, the algorithm used by Google Search, and in [160], authors use textual information and keywords to rank nodes. Another example is detecting the most influential users/ communities in social networks [161], where nodes are grouped into communities based on the users' interests and shares. Similarly, we can not use these metrics in our situation.

Therefore, this study introduces the Lock-ADM algorithm to find the optimal number of locks to automate and their placement, using adapted metrics. In the next section, we provide more details on our problem formulation and introduce the proposed solution.

3.4 Problem Formulation and Proposed Solution

Automating all the locks is costly and time-consuming. However, as shown in Section 3.2.5, it can bring significant financial benefits. These benefits vary from one lock to another. Additionally, we notice that for some locks, these benefits can be negligible. Hence, they should not be automated in priority. Therefore, in this study, we propose to select only relevant locks to automate.

Choosing the locks to be automated implies selecting the optimal number of locks and the best candidates among them. These best candidates ensure that journey times are reduced by minimizing the waiting time in locks, minimizing costs, and maximizing benefits to the whole network. We use graph theory as a powerful tool to characterize the river network by considering locks as the nodes and the path between them as the edges. Finding a group of best locks in a network is an example of finding a maximum weight clique in a weighted graph problem. As [162] discussed, the latter problem has been proved to be a Nondeterministic polynomial-time-hard (NP-hard) problem. As such, our proposed research problem is also an NP-hard problem. Therefore, we develop a robust solution, Lock-ADM, shown in Figure 3.4, which consists of three stages used independently to solve several constraint functions. We start by calculating the optimal number of locks to be automated. In this stage, the objective is to find the optimal number n of locks to be automated with minimum cost. To address this goal, we use exact solvers such as Gurobi, CPLEX, and LINDO. However, for large-scale instances, Particle Swarm Optimization (PSO) [163] could be used. The second stage aims to select the n best locks to automate according to their importance in the network using different criteria, including centrality measurements. Generally, in experiments, we find more than the expected number of locks. Therefore, a third stage is

required to select the exact n best locks, using a metaheuristic.

More details of the Lock-ADM are given in the next section.

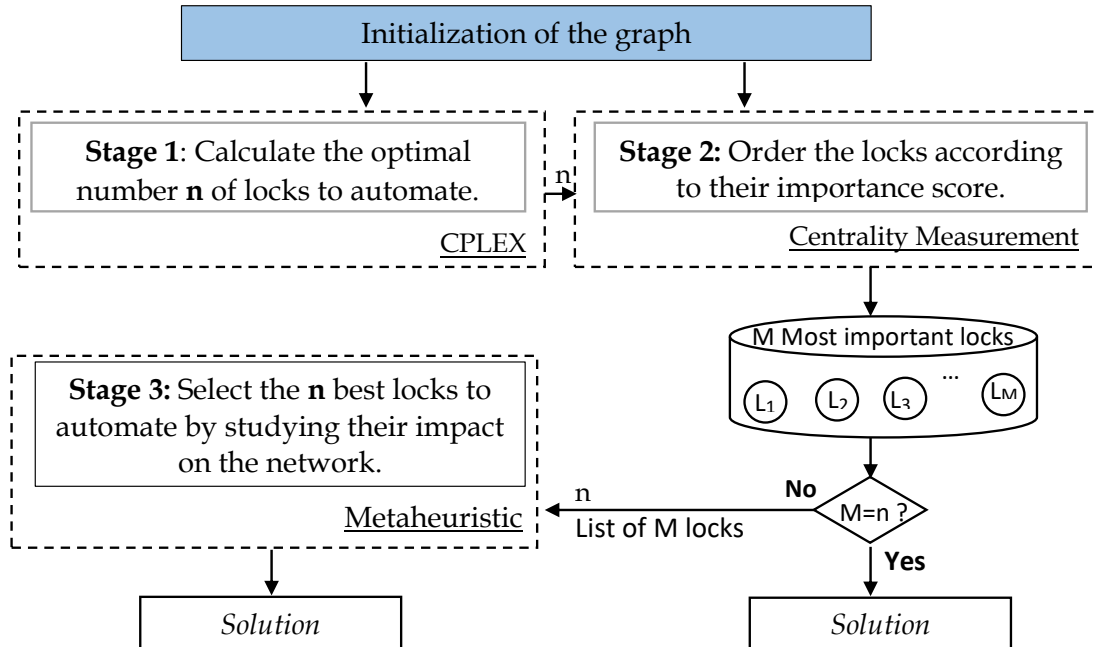


Figure 3.4: Lock-ADM workflow

3.5 Lock-ADM: Optimal Number of Locks to Automate and Their Placement

Automation is making its way into our daily lives and is certainly changing the way we travel and move goods. In order to stay competitive towards other modes, minimizing the waiting time by automating locks is the objective of lock managers and ship owners. This section details the process of locks automation. We begin by formulating the problem. Then, we describe the proposed solution and its three stages.

3.5.1 Lock-ADM - Stage 1: Calculate the Optimal Number of Locks to Automate

Since automating a lock is time-consuming and costly, the algorithm should calculate the optimal number of locks to automate while allowing to recover the automation cost in fewer years while respecting budgetary constraints. This section describes the optimization

problem and defines the objective function [164] and the constraint conformance function to solve.

3.5.1.1 Problem Parameters and Assumptions

In Table 3.1, we define the parameters notation and values [153]. Usually, there are two lock-keepers per lock, so the annual lock-keepers salary is $2 * \text{locker_salary}$. We assume that only $\text{percentage_budget} \%$ per year of the company's budget (VNF in our case) can be dedicated to the automation. As shown in Figure 3.3, locks' automation has advantages in traffic density. Indeed, an increase in the number of passages involves an increase in the earnings from tolls. The gain value is estimated regarding the current prices [153].

| Notation | Meaning |
|--------------------|--|
| nb_lock | The number of locks to automate(variable) |
| nb_Year | The number of years over which the automation costs are reimbursed(variable) |
| nb_max | The maximal number of locks |
| $automation_cost$ | The cost of automating one lock |
| $locker_salary$ | The annual lock keeper's salary |
| max_budget | The maximum budget dedicated to automation by the company |
| $gain_toll$ | The gain brought by the automation of a lock |

Table 3.1: Problem parameters

3.5.1.2 Problem Formulation

The objective is to maximize the gain thanks to locks automation with the minimum investment cost. So, we define the objective function as follows:

$$\begin{aligned}
 & \textit{Maximize} \\
 & (nb_Year * nb_lock * gain_toll \\
 & + nb_Year * locker_salary \\
 & - nb_lock * automation_cost)
 \end{aligned} \tag{3.1}$$

subject to:

1. $1 \leq nb_lock \leq nb_max \Rightarrow$ The number of locks can not exceed the number maximal of locks in all the network,
2. $nb_lock * automation_cost \leq nb_year * (max_budget) \Rightarrow$ The cost of automation must not exceed the maximum automation budget,
3. $nb_lock * automation_cost \leq nb_year * nb_lock * gain_toll + nb_year * locker_salary \Rightarrow$ The expenses must be less than the profits.

This function is a constrained Non Linear Programming (NLP) problem.

3.5.1.3 Problem Resolution

Optimization, which consists in selecting the best option among many possible choices without violating constraints, is considered an NLP problem. There are various methods for solving NLP problems, and no single method is the best for all problems. Small-scale instances, such as our problem (1,630 locks), can be solved optimally in less than 1 minute by exact solvers such as Gurobi, CPLEX, and LINDO. For large-scale instances, such as locks all over Europe or road tolls, metaheuristics like Particle Swarm Optimization (PSO) [163] could be used. PSO algorithms are considered computationally efficient algorithms. They are now widely used in optimization due to their low cost of memory, high efficiency, easy implementation, and modification (generating hybrid techniques) of high quality [165].

After calculating the number of locks to automate regarding the budget constraints, the next two stages of the algorithm identify their location. We start by ranking all the locks in the following subsection.

3.5.2 Lock-ADM - Stage 2: Order Locks According to their Importance Score

This section proposes an effective ranking method based on degree centrality values and traffic density to select the n best locks. Firstly, the traffic density score of each node is calculated using real data. Then, the nodes' properties in the network are used to compute their centrality degree score. Finally, a weighted sum of the relevant scores is considered for ranking the nodes' importance.

3.5.2.1 Problem Parameters and Assumptions

We consider a directed network \mathbf{G} . $G = (V, E)$ has $N = |V|$ nodes and $M = |E|$ edges. It also can be defined as an adjacency matrix $A = a_{uv} \in R^{N,N}$, where:

$$a_{uv} = \begin{cases} 1 & \text{if the nodes } u \text{ and } v \text{ are directly} \\ & \text{connected.} \\ 0 & \text{otherwise.} \end{cases}$$

The symbols used are detailed in Table 3.2.

3.5.2.2 Lock-ADM - Stage2a: Identifying Influential Nodes Based on Traffic Density

This stage identifies the role that each lock plays in the network by studying some traffic-related factors. To do this, we calculate the weight (W) of each node according to the traffic density as follows:

$$W(v) = \frac{\sum_{year} CROSS_{v,y}}{\sum_i \sum_{year} CROSS_{i,y}}, \quad (3.2)$$

Table 3.2: List of important symbols

| Symbol | Description |
|-----------------|--|
| V | Set of nodes $V = \{v_1, v_2, \dots, v_V\}$ |
| E | Set of edges, $E = \{e_{12}, e_{13}, \dots, e_{ij}\}$, $i, i \in E, i \neq j$ |
| G | A directed network $G = (V, E)$ |
| $A = (a_{i,j})$ | The adjacency matrix of a graph |
| $d_{i,j}$ | The distance between node i and j |
| sp_{ij} | The total number of shortest paths from i to j |
| $sp_{ij}(v)$ | The number of shortest paths that pass by v |

where $cross_{i,y}$ is the total number of ships that crossed the lock i during the year y and $\sum_i \sum_{year} cross_{i,y}$ is the total number of ships that crossed the lock i among 14 years. As shown above, in Figure 3.3, locks with similar traffic densities will not have the same traffic density after automation (Canal du nord Languedoc and Rhin canall Gamsheim). Therefore, we will study the topological characteristics of the river network to consider the network's physical structure in the following stage.

3.5.2.3 Lock-ADM - Stage2b: Identifying Influential Nodes Based on Their Centrality in the Network, Neighbors, and Edges

This section aims to review well-known measures of centrality named Degree, Betweenness, Closeness, Random Walk Betweenness, and Eigenvector, which are mainly used to analyze social networks such as food web, internet graphs, and biological networks. By considering centrality measures, we could identify the most critical nodes in the network. According to the considered information, these measures can be divided into three categories: local measure, semi-local measure, and global measure [166].

❖ Degree Centrality:

We define the first local measure, the degree (deg) of a vertex v as the number of links (edges) it has. Then, we calculate the standardized Degree Centrality (DC) of a vertex v , for a given graph G as follows:

$$DC(v) = \frac{deg(v)}{N - 1} \quad (3.3)$$

The higher the degree, the more central the node is. Degree centrality is a good measure of the total connections a node has, but it does not necessarily indicate the importance of a node connecting others or how central it is to all the group.

❖ Betweenness Centrality: The betweenness centrality (BWC) of a vertex is a global measure of the fraction of shortest paths between any two vertices going through

the vertex. It is one of the widely used shortest path-based centrality metrics for the complex network analysis. Then, we calculate the standardized Betweenness centrality of a vertex \mathbf{v} , for a given graph \mathbf{G} as follows:

$$\text{BWC}(\mathbf{v}) = \sum_{i \neq j \neq v} \frac{sp_{ij}(v)}{sp_{ij}} \quad (3.4)$$

A node would have a high BWC if it appears in many shortest paths. Nodes with high betweenness may have considerable influence within a network by virtue of their control over information passing between others.

- ❖ **Closeness Centrality:** The Closeness Centrality (CC) counts among the global measures that identify nodes that can spread information efficiently through a graph. It is calculated as the normalized inverse of the sum of the length of the shortest paths between the node and all other nodes in the graph as follows:

$$\text{CC}(\mathbf{v}) = \frac{N - 1}{\sum_j d_{v,j}} \quad (3.5)$$

Thus, the more central a node is, the closer it is to all other nodes.

- ❖ **Random Walk Betweenness Centrality:** Random walk closeness centrality (WBc) [167] is a global measure that describes the average speed with which randomly walking processes reach a node from other node in the network. It is similar to the closeness centrality except that the remoteness is measured by the expected length of a random walk rather than by the shortest path. The measure is based on random walks, counting how often a node is crossed by a random walk between two other nodes. Then, we calculate the random walk betweenness centrality of a vertex \mathbf{v} , for a given graph \mathbf{G} as follows:

$$\text{WBc}(\mathbf{v}) = \frac{\sum_{s < t} I_v^{st}}{\frac{1}{2}n(n-1)}, \quad (3.6)$$

where $I_v^{st} = \frac{1}{2} \sum_j A_{vj} |T_{vs} - T_{vt} - T_{js} + T_{jt}|$, for $v \neq s, t$ and $I_s^{st} = 1, I_t^{st} = 1$. T is a generated matrix by removing last row and column from the Laplacian matrix and then back substitution of zero rows and columns into inverse Laplacian matrix.

- ❖ **Eigenvector centrality:** Eigenvector centrality is a semi-local measurement of the centrality of a node in a network based on the weighted sum of centralities of its neighbors. It measures the influence of nodes in road network. The eigenvector centrality $Ec(v)$ of a node v is given by:

$$Ec(\mathbf{v}) = \frac{1}{\lambda} \sum_k a_{k,v} Ec(k) \quad (3.7)$$

where $\lambda \neq 0$ is a constant. In matrix form we have:

$$\lambda Ec = EcA$$

The eigenvector centrality network metric takes into consideration not only how many connections a vertex has, but also the centrality of the vertices that it is connected to.

3.5.2.4 Discussions

It is obvious that different techniques look at varying networks' properties to assess nodes' importance. In general, the best measure ought to coordinate more information on networks to calculate node importance. The six evaluation techniques can identify the capability of each node. However, different centrality measures evaluate the importance of nodes from various points of view. The centrality measures of DC, BWC, CC, and WBc are single indices to evaluate node importance, which considers only one of their attributes. Therefore, we introduce a formula that combines three foreknowledge: "Important locks have lots of information about them" (DC, W). "Important locks are enclosed by other important locks" (BWC, CC, Ec). "Having many links even unimportant makes the lock important" (DC, WBc). It does not require a manual assignment of weights to node properties. The general formula to rank the nodes we develop is as follows:

$$Degfin(v) = \sum_i \alpha_i c_i(v) \quad (3.8)$$

Where $c_i(v)$ are the different measurement of centrality (DC(v), BWC(v), CC(v), WBc(v), Ec(v), and W(v)) and α_i are variables to set according to each problem ($\sum_i \alpha_i = 1$). We will discuss later how we choose them in our implementation. Experiments show that there may be more than the required nodes to be returned when several locks have the same maximum importance score, i.e., we can find $M > n$ locks having the same importance. Therefore, we get the list of M nodes with the best scores to the next step to measure their outer importance by studying their impact on the network.

3.5.3 Lock-ADM - Stage3: Select Best Locks to Automate by Studying their Impact on the Network

Selecting the best locks to automate by studying their impact on the network is also known as NP-hard since we only reduced the search setting, which is a subset of the first problem. It is time-consuming to obtain the optimal solution with deterministic methods by testing all the possible n-tuple combinations of the list, especially when there are many locks. Therefore, the use of a metaheuristic algorithm is required to select the best n-tuple of locks efficiently.

3.5.3.1 Metaheuristic Choice

To solve this problem, the metaheuristic algorithm must meet the following characteristics:

- Population-based: Our problem involves selecting certain locks among the locks list. Therefore, if we consider that the locks list forms a population where the locks are the individuals, the metaheuristic should be able to deal with populations and generate new ones in order to obtain the best solution.
- Fitness-oriented: Selecting the best locks is based on measuring its impact on the network. Thus, the metaheuristic should use a fitness function to evaluate the quality of a generated solution.
- Variation-Driven: If there is no acceptable solution in the current population according to each individual's fitness function, the metaheuristic should generate new solutions. Consequently, individual solutions should produce variations to generate new solutions.

In the literature, numerous evolutionary algorithms meeting the above criteria are proposed. Many researchers have studied the performance of these algorithms in different application fields [168, 169]. However, it is proven that these results are problem-dependent, and there is no best metaheuristic for any optimization problem. Therefore, in this study, we conduct a comparison approach of three well-known metaheuristics, namely Artificial Bee Colony algorithm (ABC) [170], Genetic Algorithm (GA) [171], and Simulated Annealing (SA) algorithm [172], to select the adequate one for our problem.

Figure 3.5 gives an overview of the used algorithms flow chart. The three main steps are population initialization, evaluation of fitness function, and generation of new population.

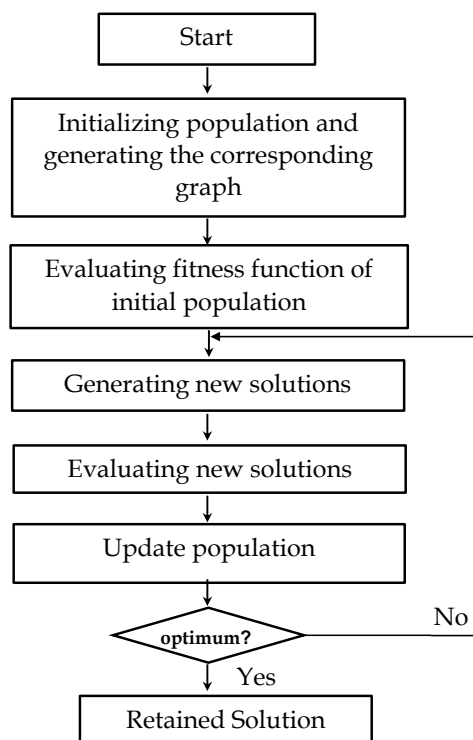


Figure 3.5: General flow chart of the used algorithms.

3.5.3.2 Initializing Population and Generating the Corresponding Graph

The first step is to generate an initial population in which a set of possible solutions are contained. We form the initial population randomly from the selected locks with the highest scores.

3.5.3.3 Evaluating the Fitness Function

The fitness function consists in calculating the average travel time between test points.

❖ Test Points

By test points, we refer to the locks considered to calculate the distance between them before and after the changes. Since we cannot consider all the existing locks in one test procedure, we need to select some test points covering most of the relevant locks to reduce the test process's time and cost. All the locations are randomly chosen from either side of the border locks. This choice ensures that each path can go through the maximum number of points that have been changed.

❖ Fitness Function

The proposed algorithm calculates the travel time between every 2 test points and gives the average travel time (ATT) between all the test points. The idea is to find the shortest path between two locks and then calculate the travel time to cross that path. The total travel time includes the circulation time taken by the ships between the two locks and also the possible waiting time necessary to cross the intermediate locks.

Several works [173, 174, 175] have been carried out to compare the path search algorithms. They prove that A* Search [176] is the best implementation for most of the graphs and found to be superior to other approaches in terms of distance traveled per time complexity ratio. This is supported by the minimal computation process needed and relatively short searching time. Therefore, we decide to use the A* **search** algorithm to find the best path, as illustrated in Algorithm 1. It is an informed search algorithm, as it uses information about path cost and uses heuristics to find the solution. A* is a modification of Dijkstra's algorithm [177], which is optimized for a single destination. Dijkstra's algorithm can find paths to all locations; A* finds paths to one location or the closest of several locations. It prioritizes paths that seem to lead closer to a goal. Each time A* enters a lock, it calculates the cost to travel to all neighboring locks, and then comes the lock with the lowest cost.

3.5.3.4 Generating New Solutions

We represent a solution as an n-tuple combination of locks. For example if we have n= 10, solutions will be represented as shown in 3.6:

To generate new solutions, each algorithm applies different techniques as follows:

Algorithm 1 Fitness Function

INPUTS: G, Test Points TP
OUTPUT: ATT (Average Travel Time)
for $lock_i, lock_j, i \neq j$ in TP **do**
 Find the shortest path from $lock_i$ **to** $lock_j$
 shortest_path= A^* (graph: G, from: $lock_i$, to: $lock_j$, metric: distance)
 Compute the total travel distance.
 sumDistance= Distance of *shortest_path*.
 Compute the total travel time.
 travel_time= $\frac{sumDistance}{ships' speed}$ + waiting time
end for
Compute the mean of the travel time between all the TP.
ATT= mean (travel_time)

return ATT

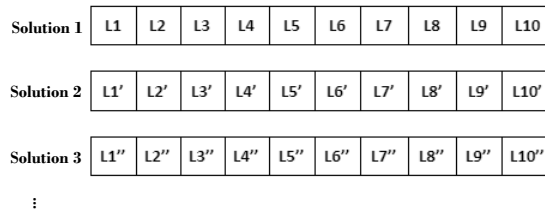


Figure 3.6: Solutions representation, where L1,L1',L2',.....L10' are the locks.

(a) **Artificial Bee Colony (ABC)**: It has mainly three phases:

- **Employed Bees phase**: each employed bee visits a solution and generates a nearby one by performing a local search. We retain the best solution among the original and the new one.
- **Onlooker Bees Phase**: Unlike the employed bees, the onlooker bees calculate a probability value for each solution shared by the employed bees and generates new solutions.
- **Scout Bees Phase**: If a solution cannot be improved for a predetermined number of tries during the onlooker and employed bee phases, then the employed bee associated with that solution becomes a scout bee. Then, the scout bee finds a new solution.

(b) **Genetic Algorithm (GA)**: The search progress is achieved by two wide operations, namely crossover and mutation. Both operations constitute the exploitation and exploration part of the search.

- **Crossover**: It is the process whereby two chromosomes (called parents) partially contribute characteristics to a new chromosome (child). The type of crossover we used

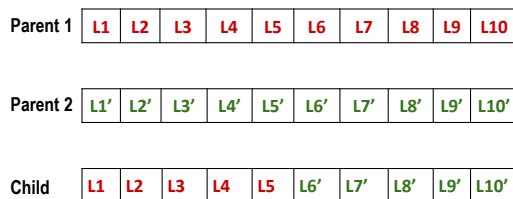


Figure 3.7: Crossover explanation

is single-point crossover, where the child inherits all genes from one parent up to the crossover.

- **Mutation:** It is used to explore new solutions in the solution space. It is important to avoid blocking in local optima.

(c) Simulated Annealing (SA): The neighborhood generation used in this work is based on n -opt moves, which is the number of instances for which the best-known solution quality was reached. For a given solution S , composed of n elements, we select n neighbors randomly.

3.5.3.5 Retained Solution

The retained solution of the algorithm is the optimal n -tuple of the locks that guarantee a minimum average travel time (ATT) within an input graph.

3.6 Implementation and Results

In this section, we detail the experimentation process and analyze the results. We first rank the locks using the importance score calculation findings (see Stage 2 explained in Section 3.5.2). Second, we compare the three metaheuristics to deduce which one best suits our problem (see Stage 3 explained in Section 3.5.3). Then, we test the efficiency of the entire algorithm, Lock-ADM, against both the Random-Selection algorithm and the All-Automated algorithm (automating all the locks).

3.6.1 Locks Importance Score Calculation (Stage 2)

On the one hand, we compute all the centrality measures (Degree, Betweenness, Closeness, Random Walk Betweenness, and Eigenvector) for all the nodes in the network. We use the real positions of locks in the network from GoogleMaps and the simulations are done using the `networkx` tool [178], which is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

The results from these simulations are presented in Figure 3.8.

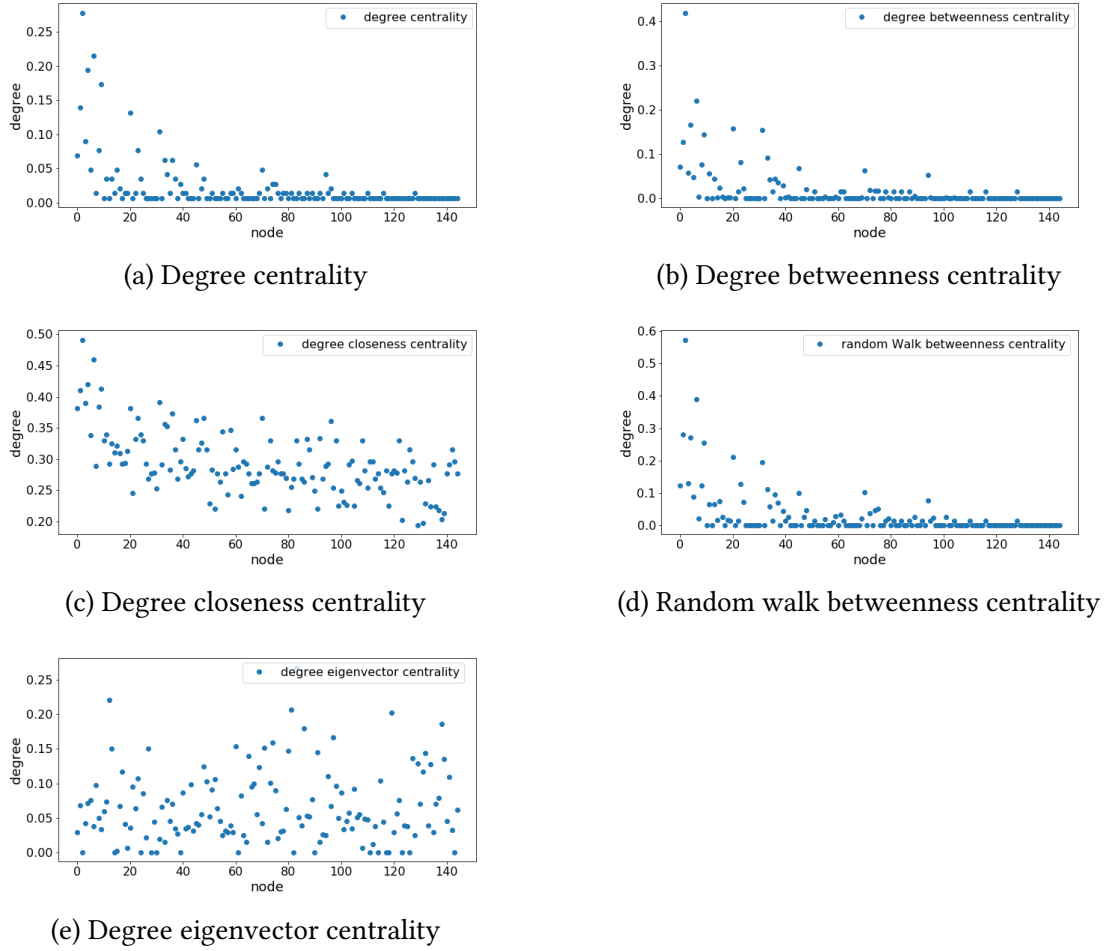


Figure 3.8: Centrality measures

Figure 3.8(a), Figure 3.8(b), and Figure 3.8(d) reveal a high correlation between DC, BC, and Wbc measures, while other measures are weakly correlated. In general, vertices with higher shortest path betweenness tend to have high random walk betweenness. This visual correlation is analyzed in more empirical evidence. Thus, we calculate the Pearson correlation coefficient \mathbf{R} between all the five measures.

The calculation formula of \mathbf{R} is given by:

$$R = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}} \quad (3.9)$$

where \mathbf{n} is the total number of samples, \mathbf{i} refers to a specific sample, \mathbf{x} and \mathbf{y} are the variables, and \bar{x} and \bar{y} are the means of the corresponding variables. If the two variables are positively linearly correlated, then $0 < R \leq 1$. If two variables are negatively linearly correlated, then $-1 \leq R < 0$. If there is no linear correlation between the two variables, then $R=0$. Generally, if $|R| > 0.8$, then the two variables are considered to have a strong linear correlation.

| | DC | BWC | CC | WBc | Ec |
|-----|-------|-------|-------|-------|-------|
| DC | | 0.97 | 0.62 | 0.99 | -0.12 |
| BWC | 0.97 | | 0.59 | 0.93 | -0.13 |
| CC | 0.62 | 0.59 | | 0.6 | -0.16 |
| WBc | 0.99 | 0.93 | 0.6 | | -0.11 |
| Ec | -0.12 | -0.13 | -0.16 | -0.11 | |

Table 3.3: Correlation coefficients for the centrality measures

The calculated values presented in the matrix of correlation coefficients for the centrality measures (Table 3.3) confirm the findings visually noticed from the graphs. DC, BWC, and WBc measures have high correlation coefficients: $R \sim 0.9$, meaning strongly correlated. Hence, it implies that it is unnecessary to use the three variables simultaneously; using one of these variables is sufficient for our study. Therefore, we kept the DC, CC, and Ec measurements in further calculations.

On the other hand, we plot the traffic density weight of the locks in Fig 3.9. We notice a low correlation between the traffic density weight and the degree centrality. This implies that most connected nodes are not necessarily the most chosen ones by ships. For instance, the 112nd node (encircled in red in the figure) has the highest traffic density weight (0.21) but has a low DC = 0.01 and we find that the 2nd node with the highest CD (0.27) has a traffic density weight equal to 0.13.

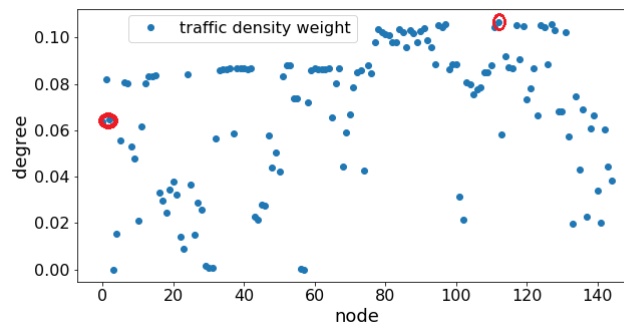


Figure 3.9: Traffic density weight

In our study, we give equal importance for each measure. So, to calculate the Degfin, defined in Equation 3.8, for each lock, we set $\alpha_{BWC} = \alpha_{WBc} = \mathbf{0}$ and $\alpha_W = \alpha_{Ec} = \alpha_{CC} = \alpha_{DC} = \mathbf{1/4}$.

3.6.2 Metaheuristics Comparison (Stage 3)

Once the locks are ranked, we implement the stage 3 of the Lock-ADM. We use a metaheuristic to select the n best locks among the locks with the highest rank. So, this second experimental study aims to compare experiments with the different selected metaheuristic algorithms with the same number of iterations. Table 3.4 illustrates the performance results in terms of execution time and best value solution for different instances.

| Number of nodes | SA | | ABC | | GA | |
|-----------------|---------------|--------------------|------------|--------------------|-----------------|--------------------|
| | Best value | Execution time (s) | Best value | Execution time (s) | Best value | Execution time (s) |
| 20 | 826.22 | 259.40 | 954.66 | 640.78 | 867.26 | 906.31 |
| 30 | 1863.03 | 386.34 | 2137.31 | 907.63 | 1720.69 | 930 |
| 50 | 1271.61 | 1316.5 | 1356.42 | 1501.92 | 1106.002 | 1702.95 |
| 100 | 1663.39 | 3484.05 | 1687.20 | 5086.28 | 979.00 | 5448.73 |
| 150 | 5578.17 | 5956.22 | 5629.66 | 7883.67 | 4127.13 | 9790.34 |
| 200 | 1966.51 | 8120.3 | 1971.30 | 10102.95 | 1329.12 | 11635.66 |

Table 3.4: Comparing metaheuristics

The experimental results show that the GA algorithm yields the best performance on average and the longest execution time. For small instances, the results are close $\sim 13\%$; however, the difference is $\sim 32\%$ for large instances. Although the ABC and SA algorithms are much quicker, the time required by the genetic algorithm is feasible for our research. Therefore, in the following experiments, we select the GA as metaheuristic.

3.6.3 Comparison with Other Algorithms

In order to demonstrate the effectiveness of the proposed algorithm, a number of randomly generated instances were tested. In particular, for a given graph, we investigate three possible test algorithms:

- All-Automated: where all the locks are automated.
- Random-Selection: where n locks randomly selected are automated. We repeat this process 10 times for each experiment, and we take the average of the value as the final result.
- Lock-ADM: where the metaheuristic selects n locks to be automated from the most important locks.

The first series of experiments aim to compare the solutions obtained during the execution of Lock-ADM with the results of the other test algorithms, namely All-Automated and Random-Selection. The results are summarized in Fig 3.10a and Fig 3.10b.

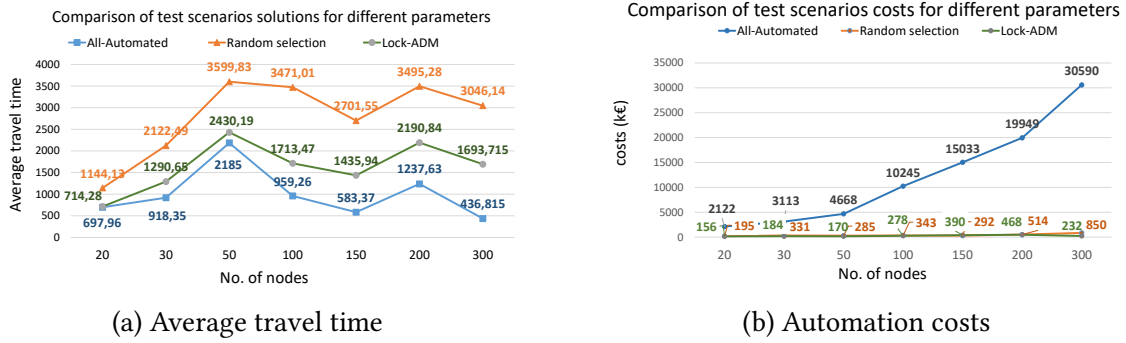


Figure 3.10: LockADM vs. other test algorithms for different number of nodes

We note that below 20 locks in the network, the locks’ required number is usually equal to the locks with the highest score ($M = n$). So, we do not need to run the metaheuristic, and we retain those locks.

We can conclude that Lock-ADM solution is very efficient in finding a solution for all the instances. Random selection of locks can provide a good solution for small instances, but it has a huge gap when the number of locks increases. Additionally, our solution results, especially for small instances, are very close to the All-Automated results. This significant finding implies that by automating a small number of locks, we can achieve high performance close to those when automating the locks but at a much lower cost.

3.6.4 Impact of the Population’s Initialization

These series of experiments aim to study the impact of the population definition. So, we compare GA’s performance with a random population initialization and Lock-ADM that used a predefined population calculated in the stage 2. Table 3.5 summarizes the GA parameters set used [179] for testing the genetic algorithm.

| Parameter | Value |
|------------------------------|-------|
| Number of locks in the graph | 1632 |
| Population size | 100 |
| No. of generations | 300 |
| No. of test points | 10 |
| Crossover rate | 0.3 |
| Mutation rate | 0.2 |

Table 3.5: Algorithm parameters for testing

Figure 3.11 demonstrates graphically the benefits of the population definition with the locks with the highest score. We can see that Lock-ADM converges rapidly for the optimal solution (within 100 iterations). However, the GA with random initialization takes over than 300 iterations and it does not converge yet. A summary of results with different number of locks is shown in Table 3.6.

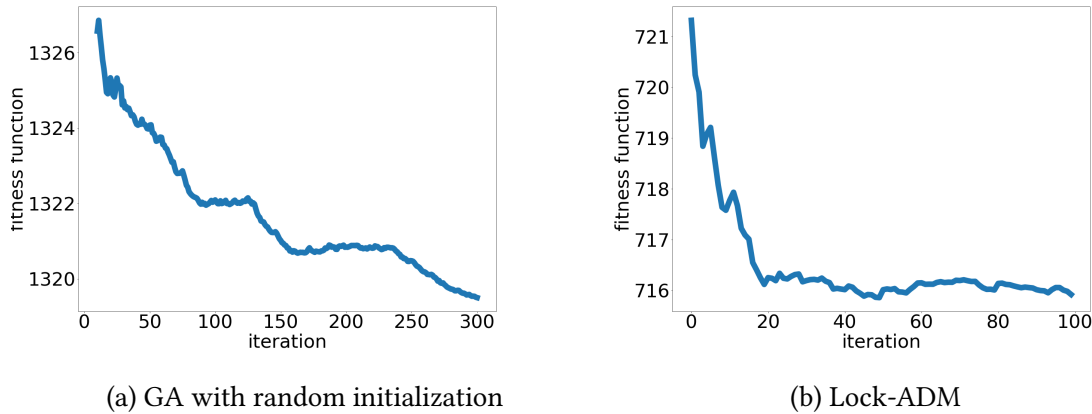


Figure 3.11: Convergence history for the Lock-ADM and the GA with random population initialization .

| Number of locks | All-Automated value | GA with random initialization | Lock-ADM | Deviation percentage |
|-----------------|---------------------|-------------------------------|----------------|----------------------|
| 20 | 667.31 | 1131.58 | 851.95 | 24.71 |
| 30 | 754.20 | 1864.89 | 1081.34 | 42.01 |
| 50 | 785.96 | 2057.38 | 1311.94 | 36.23 |
| 100 | 622.29 | 1319.50 | 715.92 | 45.72 |
| 150 | 1017.58 | 3406.74 | 2125 | 37.62 |
| 200 | 891.61 | 2134.14 | 1709.10 | 19.91 |
| 250 | 804.82 | 2177.46 | 1252.63 | 42.47 |
| 300 | 941.37 | 1866.93 | 1476.39 | 20.91 |

Table 3.6: The Lock-ADM and the GA with random initialization results compared to the All-Automated values for different parameters.

The first column represents the size of the graph in terms of number of locks. The second, the third, and fourth columns represent the All-Automated results, the GA with the random initialization, and Lock-ADM solution, respectively. The last column represents the percentage difference between the GA with the random initialization and Lock-ADM. We can conclude that starting a search with specific individuals based on preliminary analysis done in stage 2 instead of generating the population randomly enables to reach optimum design with less number of iteration and to get closer to the global optimum.

3.7 Conclusion

French navigation suffers from several deficiencies leading to long travel times, mainly due to aging infrastructure, especially its old locks. However, automating all existing locks requires enormous costs and could be unnecessary when automating a non-important lock. Therefore, this study gives decision-support systems for ship companies to adjust their river

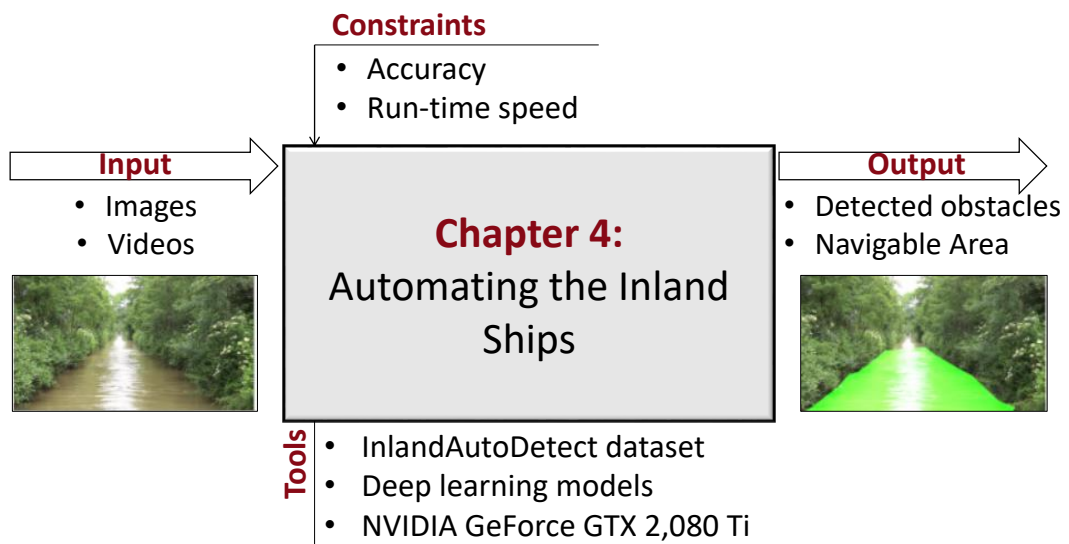
network management. We consider determining the optimal number and the best locks to automate optimally to reduce travel time and costs. We propose Lock-ADM, a three-stage algorithm for determining the number and locks' position to automate, respecting the budgetary and time constraints with a detailed explanation and the corresponding experimental results. The proposed algorithm essentially answers the following research questions: (1) What is the optimal number n of locks to be automated under budget constraints? (2) What are the most important locks in the existing river network? (3) If this number is greater than n , how can the n best locks to automate be selected efficiently?

Performance analysis shows that Lock-ADM ensures a reduced travel time close to that when automating all the locks (~ 33.7 % of deviation percentage) but at a much lower cost. This study serves as the basis of any problem aiming to change some nodes' states in a graph in a reasonable time. Many interesting research topics can be derived from our solution. For instance, studying locks' individual and external importance may inspire the most influential nodes discovery, selection, and modification in a graph for different applications. Additionally, the traffic increase analysis after automation may warrant further studies by considering other measures of importance, such as centrality measures. In future work, we can extend this research in several directions. One first extension is considering other methods and other constraints. We can, for instance, explore the method in [180]. Guo et al. that introduce a multiobjective resource-constrained selective disassembly sequence optimization using scatter search. Their goal is to minimize the energy consumption and disassembly time and maximize the disassembly profit. It should be noted, however, that there is no exact correspondence between this proposed method and the locks' automation.

AUTOMATING THE INLAND SHIPS

Your present circumstances don't determine where you can go; they merely determine where you start.

-Nido Qubein



| | | |
|-------|--|-----|
| 4.1 | Introduction | 73 |
| 4.2 | Related Work | 74 |
| 4.2.1 | Real-time Traffic Object Detection for Autonomous Vehicles | 75 |
| 4.2.2 | Real-time Drivable Area Detection for Autonomous Vehicles | 77 |
| 4.3 | InlandAutoDetect Dataset Construction | 78 |
| 4.3.1 | Data Acquisition | 78 |
| 4.3.2 | Data Model | 80 |
| 4.3.3 | Ground Truth Labeling Process | 81 |
| 4.4 | Object Detection Implementation Details | 82 |
| 4.4.1 | Hardware and Software Configuration Used | 82 |
| 4.4.2 | Data Preprocessing and Preparation | 82 |
| 4.4.3 | Architectural Configuration | 84 |
| 4.5 | Inland Object Detection Performance Evaluation | 90 |
| 4.5.1 | Boundary Conditions | 90 |
| 4.5.2 | Comparison Criteria | 91 |
| 4.5.3 | Experimental Results Analysis | 92 |
| 4.6 | Navigable Area Detection Performance Evaluation | 97 |
| 4.6.1 | Proposed System | 97 |
| 4.6.2 | Experimental Results Analysis | 98 |
| 4.6.3 | Discussions | 99 |
| 4.7 | Conclusion | 100 |

4.1 Introduction

Over the last roughly five years, autonomous ships have been and continue to be subject to an ever-increasing interest and have been the origin of numerous works and realizations. This is also the aim described in this chapter and further developed in the rest of this thesis. As an autonomous ship, the basic software proficiencies can be classified into three main groups: perception, planning, and control. During the Ph.D., we have focused mainly on the conception of a reliable perception system. Perception is one of the most critical modules in autonomous systems. It refers to its capability to collect and interpret sensory information to represent and understand the environment. It involves developing a contextual understanding of the environment, such as localizing obstacles, detecting traffic signs, and categorizing data according to their semantic meaning. Perception thus plays a crucial role in developing autonomous ships. It is the first critical part of the computational pipeline for their safe functioning. In particular, navigable area detection is the most urgent task in perception since it is the fundamental guarantee of ship security. Once the ship can select relevant data from the enclosing environment, it can accomplish the other planning and control tasks without human intervention. Path planning can be conducted in the navigable area to be guided to drive forward. Therefore, accurate navigable area detection assures safety for the autonomous ship in any environment.

The lidar-based and radar-based methods are essential parts of perception. It is easy for these sensors to obtain accurate depth information, but they are short of rich semantic information. Concretely, it is hard to detect navigable areas using lidars or radars accurately. Therefore, cameras are used to process the perception tasks in this chapter. In summary, we propose that navigable area and traffic objects detection are the essential perception tasks. With accurate detection of the navigable areas and traffic objects, the autonomous ship can sail safely and comfortably in most scenarios. The two tasks are then processed in a unified system.

In this following, we summarize our three main contributions in this chapter:

- Since there is no dataset corresponding to the types of inland conditions, we produce the first self-driving dataset, the InlandAutoDetect dataset, destined for the fluvial environment. The annotations are used for perception purposes, more precisely, for object detection through machine/deep learning processes. InlandAutoDetect dataset is open-source, and it is available online ¹ for other researchers working in this area,
- We evaluate the accuracy and performance of nine state-of-the-art perception models architecture for the problem of object detection in an inland environment: four versions of Yolo, Faster R-CNN, SSD, and three different versions of Retinanet. Among the detection models evaluated, we select the most efficient and propose an adequate configuration for inland navigation. Among these architectures, the Retinanet model with the ResNet101 backbone offers the best performance balance,
- We design a system to accurately delimit the area where the navigation is secure by simultaneously locating and mapping the area using the Retinanet model findings.

¹<https://widedhammedi.wixsite.com/phdproject/>

The proposed system considers adding a safety distance to avoid detected objects safely. Then, we analyze the results and suggest possible improvements.

This chapter is structured as follows: We present a brief review of the most commonly known existing real time object detection and drivable area detection for autonomous vehicles techniques in Section 4.2. In Section 4.3, we describe the InlandAutoDetect dataset properties, the number of images and annotated objects, the data model, and the annotation process. Then, we detail the training process implementation, and we explain the configuration made on the deep learning models in Section 4.4. Discussions of the results and performance evaluation of the traffic object detection are drawn in Section 4.5. Based on the obtained results, we describe the designed system to simultaneously locate and map the area to delimit a safe navigable zone in Section 4.6. Finally, we summarize our findings in Section 4.7.

4.2 Related Work

Traditional maritime surveillance systems rely on four main types of images: optical remote sensing images [181] [182], radar images [183] [184], infrared images [185] [186], and visible video images [187] [188]. Optical remote sensing images easily suffer from weather conditions like waves and clouds, making it difficult to achieve real-time monitoring in a long operation period and thus locating the danger more troublesome and less viable. Radar images can cover a wide range and penetrate occlusions. However, their imaging resolution is poor, so that the captured ship targets only take up a few pixels in the entire image. Moreover, radars have dead zones near their base area, and the powerful radio beams are also destructive to living creatures. This will bring many inconveniences to object detection and tracking. In addition, the high cost of radar systems makes it challenging to achieve uninterrupted work within 24 hours. Infrared images have obvious advantages, mainly at night or under the lack of light, but they fail to provide rich color information. Visible light video images can realize precise monitoring in the short-distance fluvial area. It holds many advantages: high resolution, rich in color and texture information, low price, low power consumption, and all-weather real-time operation. All these features make it the best data resource for surveillance systems. This motivates us to consider the development of a visible image camera-based system for autonomous navigation.

Indeed, Inland Unmanned Surface Ships (I-USV) must be equipped with an intelligent awareness system capable of automatically locating and classifying nearby objects in real-time. The detection must be fast enough to allow the ship to act if needed. In all these advancements towards object detection frameworks for autonomous systems, real-time object detection remains a fundamental problem to solve in building such systems [189]. Many algorithms have been proposed to solve this problem. Nevertheless, one of the most challenging difficulties in measuring these algorithms' performance is getting the suitable dataset in the suitable format. In this section, we detail the characteristics of the inland navigation waterways. Then, we benchmark existing object detection approaches and datasets. Finally, we study the drivable area detection techniques for autonomous vehicles. We mainly focus

on the adaptability of the top existing approaches for inland navigation, and we investigate the effect of such challenging conditions on their average performance.

4.2.1 Real-time Traffic Object Detection for Autonomous Vehicles

The classification consists of predicting a label for input based on the classifier's features. Object detection is a computer vision method that consists of localizing spatially in an image the Region of Interest (RoI) where an object appears. In particular, traffic object detection has become a key technology behind advanced driver assistance systems [25], mainly to perform security surveillance, road, sign, and pedestrian detection. In recent years, considerable progress has been made in this area due to the development of deep learning methods able to perform detection and classification simultaneously. However, even though these many efforts to solve this problem have been employed in recent decades, it is still challenging to create a robust and efficient object detection method due to difficulties from lighting variety, background clutter, and perspective alterations. We first explain the approaches of object detection, then the existing datasets designed for this purpose.

4.2.1.1 Object Detection Approaches

Most object detection methods can be divided into two approaches:

- ❖ **Hybrid Approaches:** Hybrid approaches consist in using a cascade of successful algorithms with two steps to identify an object. The aim of the first step is the visual extraction features to provide a semantic and robust representation. SIFT [190], HOG [191] and Haar-like [192] features are the representative ones. In the second step, a classifier is needed to distinguish a target object from all the other categories and make the representations more hierarchical, semantic, and informative for visual recognition. Usually, the Supported Vector Machine (SVM) [193], AdaBoost [194] and Deformable Part-based Model (DPM) [195] are utilized.
- ❖ **Neural Network (NN) Approaches:** Before the rise of deep learning, HOG detectors were the most popular solution. When deep learning methods began to be integrated into detection problems, the main objective was to replace HOG with a more accurate convolutional neural network (CNN) based detector. Currently, CNN architectures have emerged as a successful solution for problems related to visual object recognition [25], [26]. Over the last years, deep learning has achieved an impressive series of results thanks to its success in automatic feature extraction via multi-layer nonlinear transformations, especially in computer vision.

The choice of the most suitable object detection approach between these two popular approaches is crucial. We found that CNN-based approaches are more adaptable for learning and can yield better and more accurate results in several domains. Moreover, due to the diversity of appearances, illumination conditions, and backgrounds, it is difficult to manually design a robust feature descriptor to describe all kinds of objects perfectly. Therefore, it

has been proven that automatically learned features through the CNNs perform better than the hand-crafted ones [196]. In [197], A. Arinaldia et al. compare two systems for vehicle detection and classification, a system based on background subtraction with an SVM classifier (a hybrid approach) and a system based on the Faster R-CNN. The results show that Faster R-CNN outperforms the SVM classifier in terms of time and accuracy. This study is just one example of many other investigations that corroborate NN's superiority over Hybrid approaches in object detection.

On the other hand, NN-based object detection approaches can be classified into two main categories: region-proposal-based methods and one-step methods. In the region proposal-based methods, a region proposal is generated first, then the features inside the region proposal are used to classify the category and regress the location. Faster R-CNN [198] and R-FCN [199] are representatives examples. Faster R-CNN proposes a Region Proposal Network (RPN); it generates region proposals using neural networks. R-FCN inserts a position-sensitive score map into the ROI pooling layer. Both Faster R-CNN and R-FCN accelerate the detection speed and improve the accuracy greatly. SSD [200] and YOLO [201] are the representatives of the one-step methods. They combine the category classification and localization regression synchronously. However, this approach requires high-quality training data as support. It is thought that the recent success of object detectors is a product of the availability of more large-scale training data.

4.2.1.2 Benchmarked Datasets

Datasets are the most crucial aspect that makes algorithm training possible; without data, it is impossible to learn. Datasets provide a way to train and validate the computer vision algorithms and therefore play a crucial role in driving the research. We collect the characteristics of the existing maritime datasets in the literature in Table 4.1.

We can notice that the first datasets can only be used for a classification task as performed in [202] [209] [29] since they do not provide locations of objects (no detection). Moreover, the images constituting the [204][205][206] datasets are provided by a community of active photographers around the globe and are collected mainly for recreational purposes. We found that the annotations are inconsistent regarding class assignments, and there are only a few examples for some classes. The SMD and MarDCT datasets are used mainly for maritime ships localization and classification [210]. However, they implement an easy detection process, through which they assume that either the target ship or the sensor ship is stationary. The last two datasets focus on maritime ship detection based on deep learning models. They provide precise annotations and employ three object detectors (Faster R-CNN, SSD, and YOLO) to detect ships. However, they can only distinguish between ship types. Therefore, according to the presented literature review, we can confirm that there is no public annotated dataset designed for a fluvial environment with its specificities detailed earlier in section 2.2.3.2.

Table 4.1: Popular maritime datasets

| Dataset | Access | Description |
|---|--------------------|--|
| VAIS (Visible and Infrared Spectrums) dataset [202] | Publicly available | The most popular maritime dataset with 2865 images of ships from 6 different categories. |
| Marvel dataset [203] | Publicly available | A dataset with 140,000 labeled visible images of ships from 26 different categories. |
| FleetMon dataset [204] | Subscription-based | A dataset with 566,250 labeled ship images. |
| MaritimeTraffic dataset [205] | Subscription-based | A dataset with 2,600,000 labeled ship images of 500 different ships. |
| shipFinder dataset [206] | Subscription-based | A dataset with 459,543 labeled ship images. |
| SMD (Singapore Maritime Dataset) [207] | Publicly available | A dataset with 31,653 frames of 6 different object classes. |
| MarDCT (Maritime Detection, Classification, and Tracking) dataset [208] | Publicly available | A dataset used for detecting maritime ships with 3,066 labeled ship images. |
| SeaShips dataset [187] | Confidential | A dataset used for detecting maritime ships with 31,455 images including 6 ship types. |
| ABOShips dataset [188] | Confidential | A dataset used for detecting maritime ships with 9880 images including 9 ship types. |

4.2.2 Real-time Drivable Area Detection for Autonomous Vehicles

Drivable area detection is an essential component within the street scene examination for ADAS. Since an autonomous vehicle requires information on the road where it circulates, the system and algorithm should be as fast and straightforward as possible. Many conventional approaches detect the road using the following information set: edge, color, intensity, and shape. These approaches select a candidate road solution that maximizes the statistical distances between the two regions' color distributions created by the candidate solution. For instance, in [29], a gradient-enhancing conversion and an illumination-based lane detection algorithm are proposed. Gradient-enhancing conversion produces a color image from an intensity image that has maximized gradients at the lanes. However, we figure out that these methods based on colors are not efficient for unstructured roads without clear lane boundaries, such the inland pathways. Besides, they do not work well for complex traffic since they are influenced by lightning and the reflection of other objects.

With the recent development of deep CNNs, many deep CNN-based approaches have been proposed. Pan et al. [211] propose the Spatial CNN (SCNN), which generalizes traditional deep layer-by-layer convolutions to slice-by-slice convolutions within feature maps thus enabling message passing between pixels across rows and columns in a layer. Such SCNN is particularly suitable for long continuous shape structures or large objects, with solid spatial relationships but fewer appearance clues, such as traffic lanes, poles, and walls. In [212], the authors propose a deep neural network, called Road and road Boundary detection Network (RBNet), that can detect both road and road boundary in a single process. By implementing a Bayesian model, the RBNet can simultaneously estimate the probabilities of a pixel on the image belonging to the road and road boundary. This method improve the segmentation accuracy by utilizing the predicted results of the two tasks as guidance for each other. Oliveira et al. [213], and Lyu et al. [214] propose smaller CNNs based on an encoder-

decoder network architecture to reduce the processing time of road detection. Wang et al. [215], and Zohourian et al. [216] inject extra knowledge such as contour priors and location priors into their network architectures to improve the road segmentation performance. However, these methods require extra information such as road contour maps and location priors. Also, they may result in missing regions when illumination variations exist, including shadows and overexposure regions. Moreover, the networks' encoder-decoder structure often causes inaccurate road boundary due to the loss of details; the spatial information can easily be lost because of the down-sampling and interpolating operations contained in the encoder-decoder structure.

To sum up, in inland sailing, the pathways are unstructured roads because there are no drawn lines in the river canal. Hence, in our case, we aim to estimate the geometric structure of the lane boundaries of an unstructured pathway on images captured by the camera to delimit a safety zone for navigation. In the next section, we describe the InlandAutoDetect dataset properties, the number of images and annotated objects, the data model, and the annotation process.

4.3 InlandAutoDetect Dataset Construction

As explained in section 2.2.3.2, the fluvial environment characteristics are different from those of the other vehicles [217]. Thus, in this study, we created the InlandAutoDetect dataset to meet its specific requirements. The dataset is available to the research community following the data model we propose and provides annotated images in two formats, as described below. This section outlines the features of this proposed open-source dataset, the data model, and the labeling process.

4.3.1 Data Acquisition

The RGB images were acquired from different publicly published sources found online. We have obtained the permission of their owners, and we have anonymized the images by blurring the faces to comply with the European General Data Protection Regulation (GDPR) legislation. The image data was acquired in waterways as well as near ports. Only images containing at least one relevant object were selected from the vast corpus of collected data. Moreover, the selection was performed so that there is usually a significant scene change between any pair of selected consecutive images.

4.3.1.1 InlandAutoDetect Dataset Diversity

A good detection model should maintain sensitivity to interclass differences while giving stable test results. Due to the complexity of fluvial environments, all influencing factors need to be considered. Some real-world data, by its very nature, can be hard to predict. So, we will take the following measures to ensure the diversity of our dataset:

- **Background Selection:** In most detection tasks, especially face recognition, the detection accuracy is rarely affected by background variations because the face area is filled in a regular rectangle, easily separated from the background. Nevertheless, unlike human faces, the shape of inland objects is very irregular, which will lead to a lot of background information in the labeled bounding box. Background information will be identified as objects features and will distort final detection accuracy. To avoid the impact of a single background, we collected data under different backgrounds. To do this, we used images from cameras deployed in different locations.
- **Lighting Variations:** The illumination differences from different temporal periods and in various geographical areas are particularly significant in natural environments. Lighting variations can significantly impact image capture. Illumination throughout the day, in various geographical areas, and specific daylight hours in a given region can dramatically influence image detection. Therefore, to be exhaustive, we collected images under different lighting conditions by selecting videos at different periods and in various geographical areas.
- **Atmospheric Conditions:** The dataset includes various images of different atmospheric conditions throughout the day, including sunny, rainy, and cloudy skies.
- **Occlusion:** Since the inland waterways are narrow, a high number of ships may be in proximity to each other. This induces significant occasions when ships occlude each other or occlude other objects in the environment. However, ignoring occlusion is unreasonable and unrealistic and may cause errors later. So, we collected as much occlusion data as possible to let the subsequent trained model efficiently cope with occlusions. Two examples of occlusion are shown in Figure 4.1.



(a) Several parked ships occluding each other. (b) Trees hiding part of the infrastructure

Figure 4.1: Example image of a occlusion.

- **Visible Proportion:** As most ships in cameras are moving, only part of inland objects may appear on the screen when entering and leaving the camera’s field of view. They still belong to the objects we need to detect. We thus annotated both complete objects and incomplete parts at different visible proportions.

4.3.1.2 InlandAutoDetect Dataset Characteristics

Our dataset termed the InlandAutoDetect dataset includes a total of 3,377 images with 29,832 tightly annotated objects instances corresponding to 5 different categories of obstacles. Most images are high-resolution ($1,280 \times 720$), while others are lower (500×375). These images include challenging variations and are diverse in background, weather conditions, levels of natural occlusions and shadows, various ranges of a cloudy sky, direct sunlight, and daytime, to be exhaustive and close to real-world environmental conditions. The total number of instances is different for each class (see Table 4.3). Each image contains annotations of visible inland objects greater than 32×10 in the form of rectangular regions of interest.

The main characteristics of the InlandAutoDetect dataset are illustrated in Table 4.2 and a sample from the InlandAutoDetect dataset, with multiple challenge types, is shown in Figure 4.2.

| Number of videos | Number of annotated images | Number of classes | Resolution | Annotated object size | Challenging conditions |
|------------------|----------------------------|-------------------|----------------------|-----------------------|---|
| 5 | 3,377 | 5 | 500x375 to 1,280x720 | 32x10 to 585x194 | illumination, blur, shadow, reflexion, occlusion, noise |

Table 4.2: Main characteristics of InlandAutoDetect dataset



Figure 4.2: A sample from InlandAutoDetect dataset with multiple challenge types

4.3.2 Data Model

The data model lists a large number of objects that can be found in the fluvial environment and that can obstruct the progress of the ship. We classify these objects into five categories:

- **Riverside:** as there are no lane markings in rivers, specifying the exact location of a lane marking in a camera image may be very ambiguous. In this category, we captured the edges of the river canals as shown in Figure 4.3.
- **Ship:** in this category, we added different types of ships in shape, color, and size.
- **Person:** we added this category to detect fishers, swimmers, or people by the riverside since ships could be close to the riverside.

- **Infrastructure:** In this category, we collected the two types of construction that can be found around a river: locks and bridges.
- **Traffic sign:** as for cars, ships follow the traffic code by respecting existing traffic signs. So, in this category, we collected traffic signs that we can find in real situations.

We have tested two other versions of this data model: the first, considering the waves and debris, was declined due to the lack of data of these classes. We decided thus to combine the debris class with the infrastructure class. The second, considering multiple classes for different infrastructure and ships, was declined due to some classes' lack of data and the lower detection performance. Both versions achieved lower accuracy than this model.

4.3.3 Ground Truth Labeling Process

To prepare our ground truth annotations, we choose the labelImg tool [218], which is an open-source graphical image labeling tool that can be downloaded on GitHub. Each object in the target set annotation has two attributes: a class (riverside, ship, person, infrastructure, or road signal) and a bounding box representing an axis-aligned bounding box enclosing the object's extent.

In this project, we build an annotated dataset where each image has two separate files, one in each format: XML file, used by Darkflow, and Text file, used by Darknet.

On the one hand, the **XML file** is in the same format of the PASCAL VOC dataset [219], and it contains the object center in X, the object center in Y, the object width in X, and the object width in Y values. An example of an annotated image is shown in Figure 4.3.

On the other hand, the **text file** contains the category number, the object center in X, the object center in Y, the object width in X, and the object width in Y. So, we use the following equations to change XML files produced with LabelImg to text files:

$$* \text{ Object center in X} = \frac{(x_{\min} + x_{\max})/2}{\text{width}} \quad (4.1)$$

$$* \text{ Object center in Y} = \frac{(y_{\min} + y_{\max})/2}{\text{height}} \quad (4.2)$$

$$* \text{ Object width in X} = \frac{(x_{\max} - x_{\min})/2}{\text{width}} \quad (4.3)$$

$$* \text{ Object width in Y} = \frac{(y_{\max} - y_{\min})/2}{\text{height}} \quad (4.4)$$

To conclude, our annotation procedure is designed to be: (i) consistent, in terms of the class definition and placement of bounding boxes, (ii) accurate, by providing the fewest possible annotation errors, and (iii) comprehensive, by labeling all object appearing in the image.



(a) The details of a labeled object (ship): The blue, green and red rectangles indicate the shape of the image in the left, the object name, and its location, respectively.



(b) Multiple rectangular bounding boxes are drawn to delimit objects with corresponding class labels.

Figure 4.3: An example of an annotated image.

4.4 Object Detection Implementation Details

In this section, we discuss the implementation details of the proposed object detection methods and pre-processing techniques.

4.4.1 Hardware and Software Configuration Used

In our experiments, we use the deep learning library Keras built on the top of Tensorflow [220] with an NVIDIA GeForce GTX 2,080 Ti GPU [221] with 11GB of GPU memory on a Linux machine (ubuntu 18.04). We also use the CUDA Deep Neural Network library (cuDNN) [222], a GPU-accelerated library created by NVIDIA to improve deep neural network performance by providing optimized implementations for standard routines.

4.4.2 Data Preprocessing and Preparation

Data preprocessing is crucial since it helps enhance data quality to promote the extraction of meaningful insights. It refers to preparing the raw data to make it suitable for building and training deep neural networks models. Therefore, it has a significant impact on the performances by dealing with some problems such as over-fitting, which means that the network is highly biased to the data it has seen in the training set and cannot generalize the learned model to any other samples. To avoid this problem, we applied three methods:

4.4.2.1 Data Augmentation (DA)

When training neural networks, the recurring problem is that there is typically not enough data to maximize the generalization capability of deep neural networks, such as the case of the InlandAutoDetect dataset. Given the additional cost of annotating images for object detection, data augmentation may be an excellent solution to increase the amount of training data. Data augmentation is a widely used technique to enlarge the training dataset size virtually. This technique consists in creating new images by manipulating the original images. It has been shown to produce promising ways to increase the accuracy of detection tasks [223]. We made some alterations to our existing dataset by implementing a sequence of data augmentation techniques that can be found in real-world scenarios, such as rotating (angle=30° to angle=45°), adding random noise, Gaussian blurring, and random erasing in order to increase the amount of training data artificially. Data information about the number of classes and the number of annotations for each class is shown in Table 4.3.

Table 4.3: The detailed information of the InlandAutoDetect dataset before and after data augmentation.

| | Before DA | After DA |
|-------------------|-----------|----------|
| Total images | 3,377 | 6,753 |
| Number of classes | 5 | 5 |
| Riverside | 20,680 | 34,460 |
| ship | 4,692 | 14,076 |
| Persons | 1,768 | 5,304 |
| Infrastructure | 2,309 | 6,927 |
| Traffic sign | 383 | 1,532 |

4.4.2.2 Anchor Boxes Calculation

Anchor boxes are a common element in all recent object detection papers. Previously selective search and edge boxes were used to generate region proposals of various sizes and shapes depending on the objects in the image. However, it is impossible to generate region proposals of varied shapes with standard convolutions. Anchor boxes were introduced to overcome this issue [198]. Each grid cell in an image can be assigned with multiple prior boxes. These anchor boxes are pre-defined, and each one is responsible for size and shape within a grid cell. Object detection models use a matching phase while training to match the appropriate anchor box with the bounding boxes of each ground truth object within an image. Essentially, the anchor box with the highest degree of overlap with an object is responsible for predicting its class and location. For example, there are two anchor boxes to make two predictions per location in the image drawn in Figure 4.4.

Anchor boxes eliminate the need to scan an image with a sliding window that computes

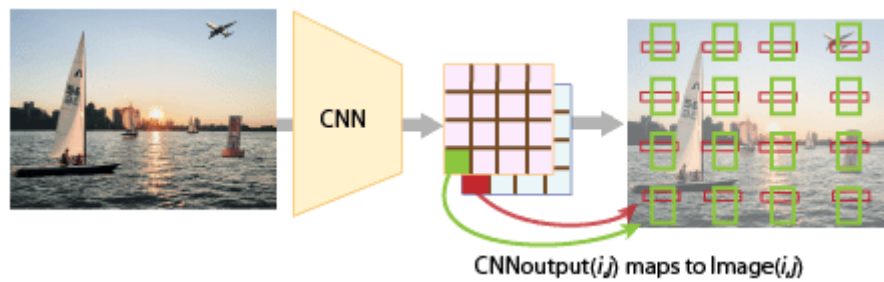


Figure 4.4: Two anchor boxes to make two predictions per location in the image.

a separate prediction at every potential position. This property is used to train the network and predict the detected objects and their locations once the network has been trained. However, if anchor boxes are not tuned correctly, the object detection model can never know that particular small, large or irregular objects exist and cannot detect them. Their shape, scale, and number impact the efficiency and accuracy of the detectors. Therefore, since the objects we aim to detect are different in terms of size and shape, we calculated correctly tuned anchor boxes based on ground truth labels, as described in [224] and [225]. The anchors estimated do not neglect tiny objects and assure that the resulting anchors guarantee high IoU between ground truth boxes. Thus, the smallest anchor box used is 21x27.

4.4.2.3 Weighted Sampling for the Training/Test Set

The InlandAutoDetect is class imbalanced since rare objects like traffic signs are under-represented, unlike riversides and ships. To address this problem, we finely sampled the dataset in train and test sets based on each class's frequency. In addition, we used the balanced weighted losses sampling to reweigh each class in the cross-entropy loss function to address the unbalanced dataset issue.

4.4.3 Architectural Configuration

We chose some network architectures from the most popular and successful object detection networks used today for this project. Each one has its benefits and drawbacks, with varying levels of accuracy and run-time speeds. In this section, we briefly discuss their methodologies and related design choices.

4.4.3.1 Object Detection Models

Nine versions of deep learning architectures were examined within the scope of the proposed system, which are Faster R-CNN, SSD, four versions of Yolo, and three versions of Retinanet:

- ❖ **Faster R-CNN** : this object detection architecture [198] is a very accurate region-

based convolutional neural network (R-CNN) [226] which improves Fast R-CNN [199] by introducing the Region Proposal Networks (RPNs). An RPN takes an image as input and outputs a set of rectangular object proposals, each with an objectness score. The predicted region proposals are then reshaped using an ROI pooling layer which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes. Thus, the Faster R-CNN architecture has two networks: a region proposal network for generating region proposals and a network using these proposals to detect objects, as illustrated in Figure 4.5.

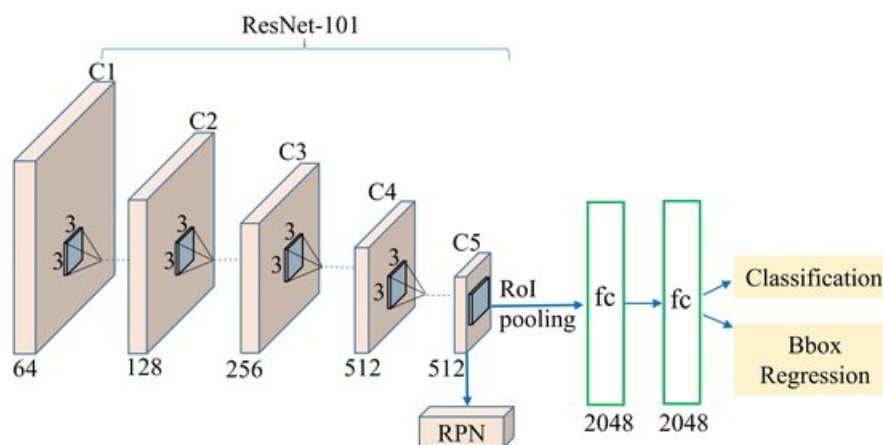


Figure 4.5: Schematic diagram of the FasterRCNN model [198], which consists of two stages. The RPN module serves as the ‘attention’ of this network. The RPN provides region proposals from that we give the ROI as input to the RCNN network.

Faster RCNN was used for different purposes, such as face detection [227], malaria parasites recognition [228], and for object detection in the wild [229].

- ❖ **Single Shot Multibox Detector (SSD)** : SSD [200] is a well-known approach for object detection in real-time. Faster R-CNN uses a region proposal network to create boundary boxes and utilizes those boxes to classify objects. While Faster R-CNN is considered the state-of-the-art in the accuracy, the whole process runs at 7 frames per second. Far below what real-time processing needs. Thus, SSD accelerates the process by eliminating the need for the region proposal network, and it uses multiscale features and default boxes to recover the drop in accuracy. These improvements allow SSD to match the Faster R-CNN’s accuracy at a higher speed. In practice, the model’s main components are a base network block that extracts feature maps and several multiscale feature blocks connected in series that apply convolution filters to detect objects, as illustrated in Figure 4.6.

SSD model was used for damage detection from post-event aerial imagery [230], garbage detection [231], target detection in Synthetic Aperture Radar (SAR) images [232], and ship detection using sentinel-1 SAR images [233].

- ❖ **You Only Look Once (YOLO) and its variants**: YOLO [201] is a high-speed multi-object detection algorithm. It takes the entire image in a single instance and predicts

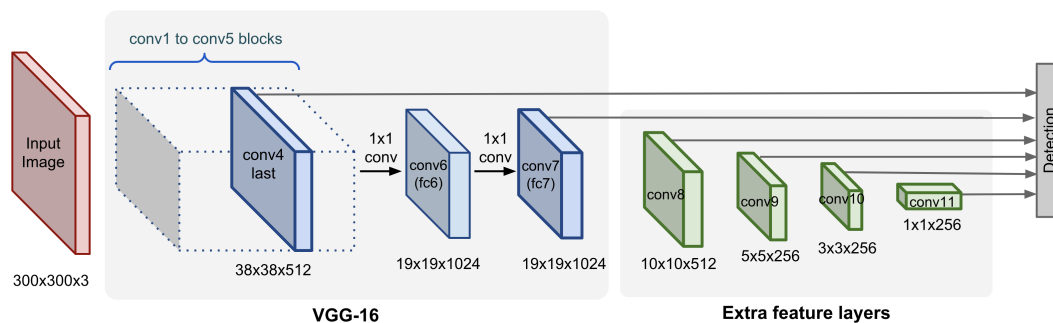


Figure 4.6: Schematic diagram of the SSD model Architecture [200]. The model adds several feature layers to the end of a base network.

the bounding box's coordinates and the class probabilities for these boxes. There are two updates for the original YOLO method, described in [201] and [234]. The newer architecture boasts residual skip connections, and upsampling and its principal characteristic is performing detections at three different scales. These modifications improve the detection performance. In this project, we use the improved versions of the models, which are Yolo v2, Tiny-Yolo v2 [234], Yolo v3, and Tiny-Yolo v3 [235]. The architecture of the model is illustrated in Figure 4.7.

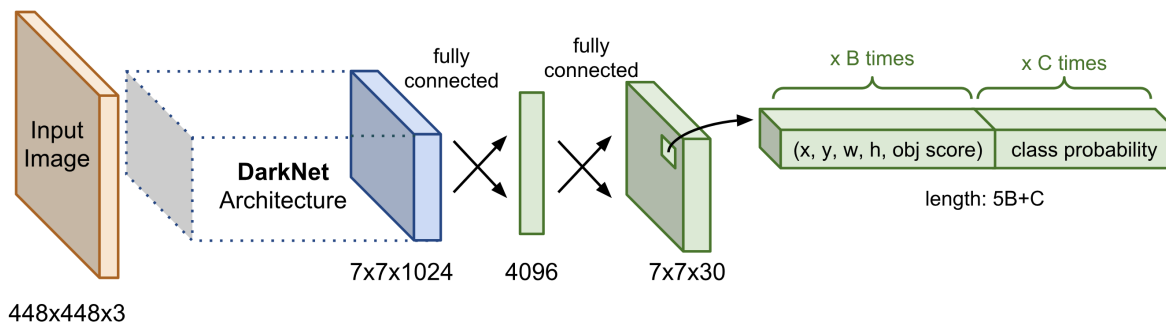


Figure 4.7: Schematic diagram of the YOLO model architecture [236]

Yolo was used for several purposes, such as automatic license plate recognition [237], breast masses detection [238], and pedestrian detection [239]

- ❖ **Retinanet:** RetinaNet [240] is one of the best single stage object detection models that has proven to work well especially with dense and small scale objects. It has been formed by making two improvements over existing single stage object detection models - Feature Pyramid Networks (FPN) [241] and Focal Loss [242]. The feature pyramid network is a structure for multiscale object detection. It combines low-resolution, semantically robust features with high-resolution, semantically weak features via a top-down pathway and lateral connections. The net result is that it produces feature maps of different scale on multiple levels in the network which helps with both classifier and regressor networks.

Retinanet was used for road damage detection [243], small object detection in aerial

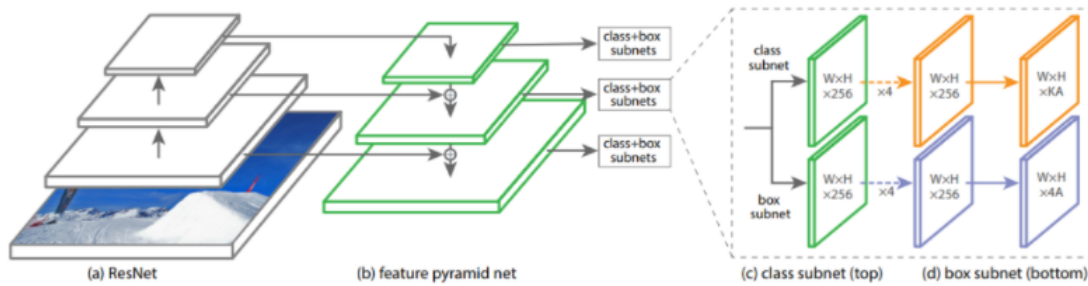


Figure 4.8: Schematic diagram of the Retinanet model Architecture [200]. The FPN is used on top of ResNet to calculate the feature maps at different scales, irrespective of the backbone’s input image size. It upsamples the spatially coarser feature maps from higher pyramid levels, and the lateral connections merge the top-down and bottom-up layers with the exact spatial sizes. Finally, the classification subnetwork predicts the probability of an object being present at each spatial location, and the regression subnetwork regresses the offset for the bounding boxes.

imagery [244], pedestrian detection [245], ship detection [246], and vehicle object detection in real scenes [247].

The standard Retinanet uses ResNet50 as a backbone. In this project, we tested the model with three different feature extractors or backbones.

4.4.3.2 Backbones (Feature Extractors)

The architecture of all deep learning models has a common component, which is the feature extractor or the backbone network. In other words, the backbone refers to the network which extracts the feature map from an input image, and it represents the first block in a deep learning architecture. We define here two well known networks:

- ❖ **Visual Geometry Group (VGG) networks:** The VGG [248] network is a convolutional neural network that gained notoriety by winning the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [249] competition in 2014. The model reached a 92.7% accuracy on Imagenet, which is one of the best scores obtained. The basic building block of a classic CNN is a sequence of: (i) a convolutional layer with padding to maintain the resolution, (ii) a nonlinearity such as a ReLU, and (iii) a pooling layer such as a maximum pooling layer. One VGG block consists of convolutional layers, followed by a maximum pooling layer for spatial downsampling. The VGG network can be partitioned into two parts: the first consists of convolutional and pooling layers, and the second consists of fully-connected layers. The convolutional part of the network connects several VGG blocks from Figure 4.9 in succession.
- ❖ **Residual Networks (ResNets):** The ResNet [250] is an innovative neural network used as a backbone for many computer vision tasks. This model was the winner of the ILSVRC challenge in 2015. Prior to ResNet, training very deep neural networks was complicated.

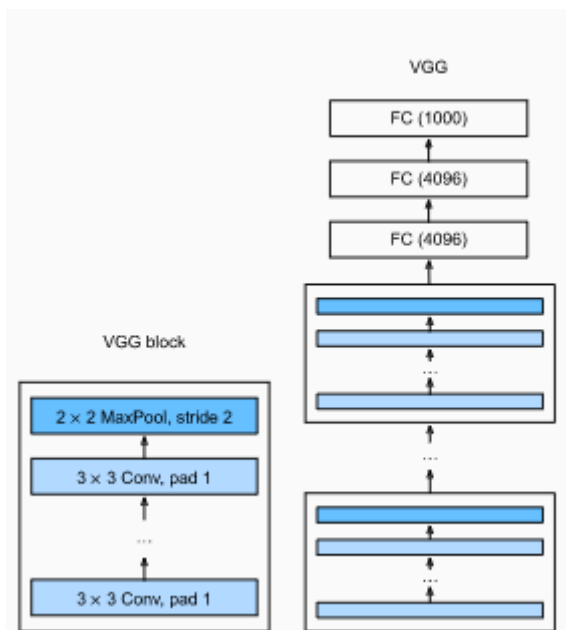


Figure 4.9: VGG architecture.

More specifically, machine learning experts try stacking more layers in their deep CNNs to solve a computer vision problem. These additional layers help to solve complex problems more efficiently as the different layers could be trained for varying tasks to get highly accurate results. However, they are hard to train because of the notorious vanishing gradient problem; as the gradient is back-propagated to earlier layers, repeated multiplication may make the gradient extremely small. As a result, as the network goes deeper, its performance gets saturated and even degrades rapidly. This degradation is not due to overfitting. Instead, it results from the problem of vanishing or exploding gradients. Therefore, ResNet was created to tackle this problem. The use of residual blocks based on the concept of "skip connections" helps to improve the accuracy of the models. These skip connections work in two ways. Firstly, they alleviate the vanishing gradient issue by setting up an alternate shortcut for the gradient to pass through. In addition, they enable the model to learn an identity function. As a result, they ensure that the higher layers of the model do not perform any worse than the lower layers. The right figure in Figure 4.10 illustrates the residual block of ResNet, where the solid line carrying the layer input to the addition operator is called a residual connection.

We show in Section 4.5 the importance of the feature extractor and how its accuracy impacts the final detector accuracy by modifying the configuration of the same model trained on the same dataset.

4.4.3.3 Fine Tuning

Transfer learning or domain adaptation involves taking features learned on one problem and leveraging them on a new, similar problem. More specifically, it consists of training the

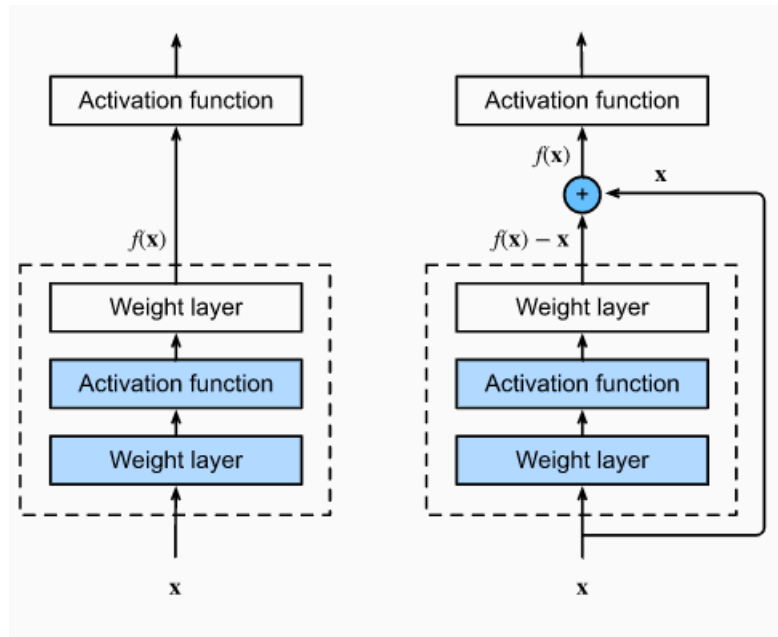


Figure 4.10: A regular block (left) and a residual block (right).

same model with another dataset with a different class distribution or other classes than in the first training dataset. It is usually done for tasks with too little data to train a full-scale model from scratch, such as the case of the InlandAutoDetect dataset. Fine-tuning, an approach of transfer learning, means making minor adjustments to achieve the desired output or performance. It involves using weights of a previous deep learning algorithm for programming another similar deep learning process. Weights connect each neuron in one layer to every neuron in the next layer in the neural network. The fine-tuning process provides ease of transferring knowledge and, thus, significantly decreases the time required for programming and processing a new deep learning algorithm as it already contains vital information. In our implementation, we initialize our models with weights of the same trained models on the Pascal VOC dataset, which is considered a reference dataset in the object detection problem.

To incarnate the fine-tuning in our context of the study, we employed the following four steps, illustrated in Figure 4.11:

1. We pretrained the neural network model, i.e., the source model, on a large dataset (the PascalVoc dataset in our case).
2. We created a new neural network model. This copied all model designs and parameters on the source model except the output layer. We added an output layer to the target model, whose number of outputs is the number of classes in our dataset, the InlandAutoDetect dataset. Then we randomly initialized the model parameters of this layer.
3. We trained the target model on the InlandAutoDetect dataset. The output layer is trained from scratch, while the parameters of the other layers are frozen based on the parameters of the source model.

- We unfroze the entire model we obtained above and re-train it on the new InlandAutoDetect dataset with a low learning rate.

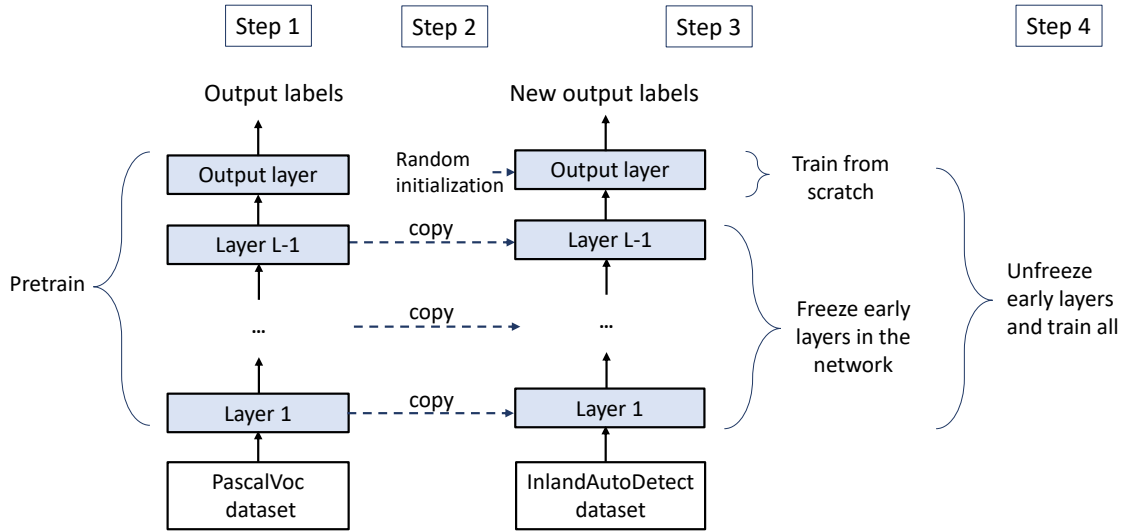


Figure 4.11: Fine tuning steps.

4.5 Inland Object Detection Performance Evaluation

In this section, we evaluate the effectiveness of the models mentioned above in the inland environment. We first started studying the environment requirements to deduce the boundary conditions corresponding to the biggest and heaviest ship reaction time. Then, we compared the performance of the trained models in terms of accuracy and run-time speed. To provide a realistic evaluation, we used a test set which is different from the training set, and it contains examples of every object mentioned in the data model with different backgrounds.

4.5.1 Boundary Conditions

Handling ships is considered as a complicated task since ships do not have an anti-lock braking system, an anti-skid braking system like those used on aircraft and land vehicles. Therefore, autonomous ships should be able to detect nearby objects in real time in order to react in real time and hence avoid the risk of a collision in a timely manner. Our objective is to find the most suitable object detector for an inland waterway environment with the best balance between accuracy and run-time speed. We started by studying the boundary conditions to ensure safe navigation for autonomous ships. To do this, we considered the worst situation in which the largest ship, containing the maximum allowed load and sailing at the fastest allowed speed must stop immediately. Then, to confirm whether a NN model is acting in real time conditions, we needed to calculate the time it takes for a ship to stop in real conditions and compare it to its time response.

The navigation authority responsible for the management of the inland waterways networks [153] indicates that "*unless otherwise specified, signposts indicate limited speed on river canals at 6 or 8 km/h and between 10 and 15 km/h on rivers*". The type of ship, dimensions, mass, and velocity are the most critical parameters for the ship's stopping distance and time. So, to calculate the minimum value of the stopping time (t_{min}), we should consider the maximum value of stopping distance (d_{max}) [251].

$$d_{max} = \frac{load_{max} * speed_{max}^2}{(2 * brake\ force)} = 5.9\ \text{meters}, \quad (4.5)$$

$$t_{min} = \frac{d_{max}}{\text{ships's speed}} = 1.41\ \text{seconds} \implies 0.7\ \text{FPS}. \quad (4.6)$$

Thus, the model should be capable to receive an input frame, analyze it, detect existing objects and send the findings to the control system in **less than 1.41 seconds**.

4.5.2 Comparison Criteria

We evaluated the different networks mentioned above using the following standard metrics:

- **mAP** (mean Average Precision): it is the first popular metric used for evaluating detection algorithms. The mAP is computed by calculating average precision (AP) separately for each class, then calculating the average over the class. A detection is considered a true positive only if the Intersection over Union (IoU) is above 0.5.

$$mAP = \sum_{q=1}^Q \frac{AveP(q)}{Q} \quad (4.7)$$

where Q is the number of queries in the set, and $AveP(q)$ is the average precision (AP) for a given query q .

The AP summarizes a precision-recall curve as the weighted mean of precisions achieved at each threshold, using the increase in recall from the previous threshold as the weight:

$$AP = \sum_n (R_n - R_{n-1})P_n \quad (4.8)$$

where P_n and R_n are the precision and recall at the n^{th} threshold. The recall refers to the percentage of total relevant results correctly classified by the algorithm, and precision means the percentage of relevant results.

- **IoU** (Intersection over Union): it is the second popular metric used for evaluating detection algorithms. It is a method that quantifies the overlap between the target bounding box and the predicted bounding box, as illustrated in Figure 4.12.

$$\text{Mathematically, IoU} = \frac{\text{Area of Overlap (Ground truth box} \cap \text{Detected box)}}{\text{Area of Union (Ground truth box} \cup \text{Detected box)}} \quad (4.9)$$

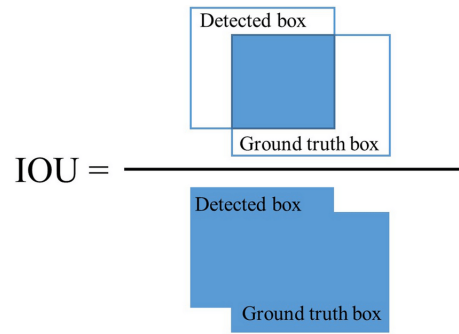


Figure 4.12: Intersection over Union calculation.

- **FPS** (Frame per Second): It is the third important metric used for evaluating detection algorithms, and it measures the number of frames processed per second.
- **WmAP** (Weighted mAP): we added this metric to keep track of each class importance thanks to the frequency. So that large and small classes have a proportional effect on the result concerning their size.

4.5.3 Experimental Results Analysis

We compared the seven different deep learning architectures (Four versions of Yolo, Faster R-CNN, SSD, and Retinanet). These models are trained using the same dataset, the InlandAutoDetect dataset. To evaluate their object detection performance, we first draw the models' predictions for the same two images. Figure 4.13, Figure 4.14, Figure 4.15, Figure 4.16, Figure 4.17, Figure 4.18, and Figure show the predictions of YOLO v3 model, Tiny-YOLO v3 model, YOLO v2 model, Tiny-YOLO v2, Faster R-CNN model, SSD, and Retinanet model, respectively.

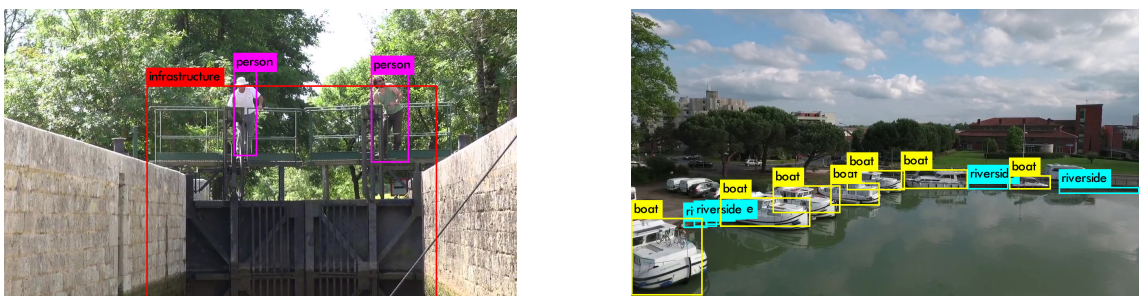


Figure 4.13: A sample of predicted images where the rectangles and text overlaid on the figures are the outputs generated by YOLO v3 at test time.

It is noticeable that all the models can detect almost all the classes with error percentages that differ from model to model and from image to image. To provide a deeper comparison, we show the experimental results in Table 4.4. We first compared them in terms of accuracy, so we used the first metric (mAP). We set the IoU threshold to 0.5, as suggested in [252]. This value ensures that at least half of the objects are correctly detected.

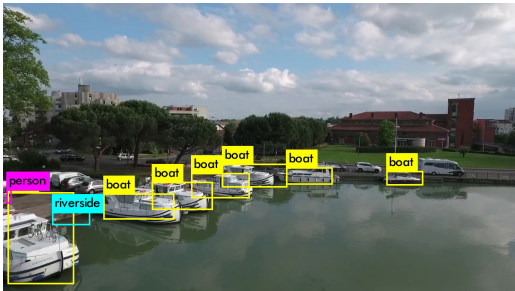


Figure 4.14: A sample of predicted images where the rectangles and text overlaid on the figures are the outputs generated by Tiny-YOLO v3 at test time.

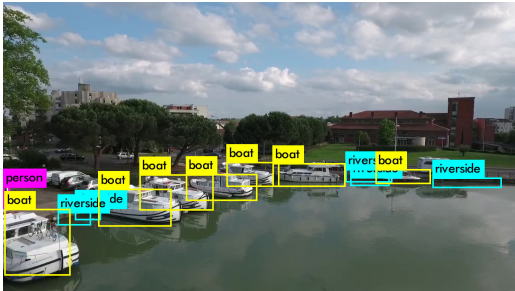
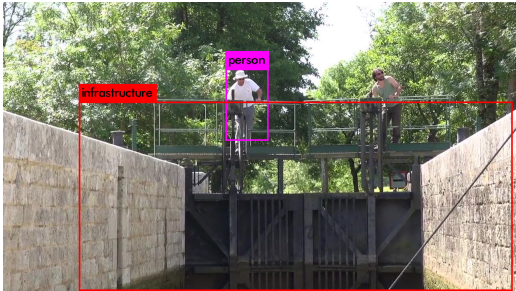


Figure 4.15: A sample of predicted images where the rectangles and text overlaid on the figures are the outputs generated by YOLO v2 at test time.

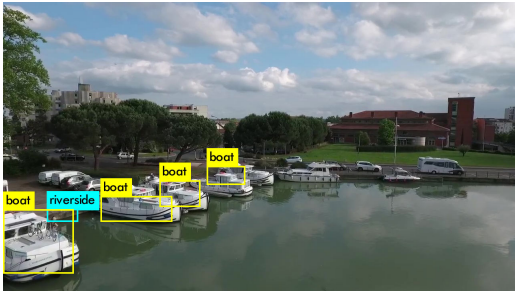


Figure 4.16: A sample of predicted images where the rectangles and text overlaid on the figures are the outputs generated by Tiny-YOLO v2 at test time.

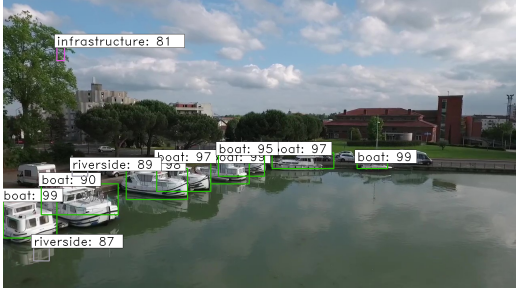
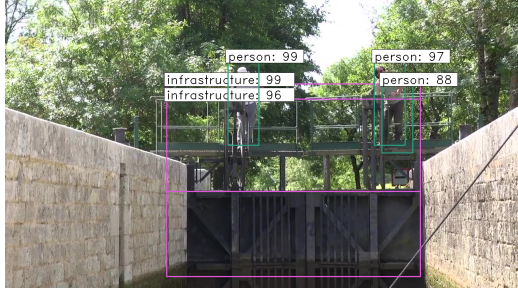


Figure 4.17: A sample of predicted images where the rectangles and text overlaid on the figures are the outputs generated by Faster-RCNN at test time.

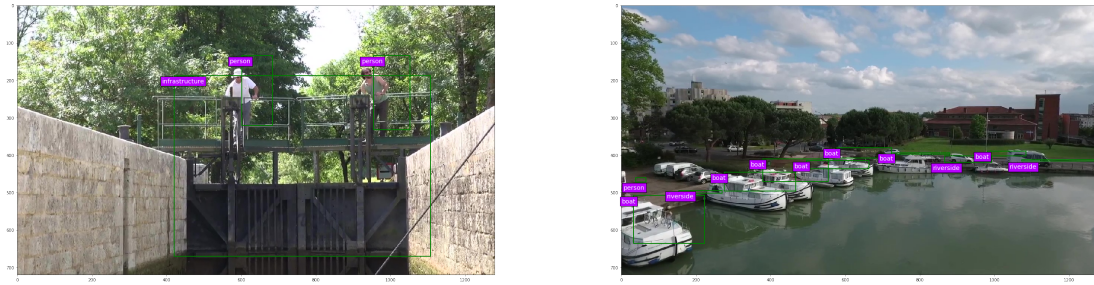


Figure 4.18: A sample of predicted images where the rectangles and text overlaid on the figures are the outputs generated by SSD at test time.

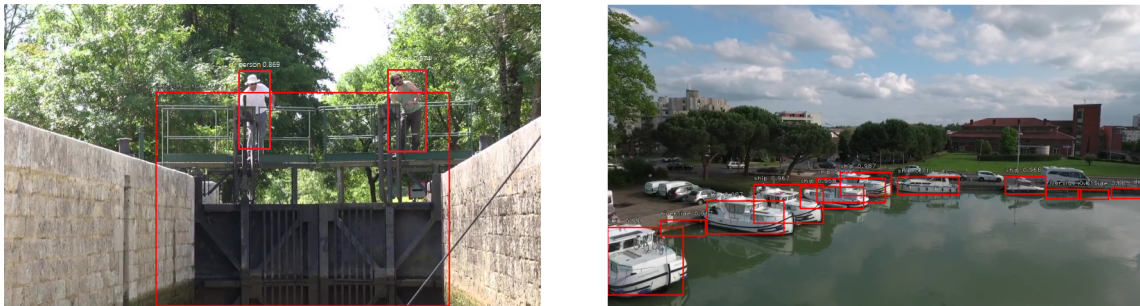


Figure 4.19: A sample of predicted images where the rectangles and text overlaid on the figures are the outputs generated by Retinanet at test time.

Table 4.4: Detection metrics in terms of mAP and AP.

| Model | WmAP (IoU=0.5) | AP | | | | |
|-------------------------|-------------------|------------------|-------------|---------------|-----------------------------|--------------------------|
| | | <i>Riverside</i> | <i>ship</i> | <i>Person</i> | <i>Infrast- ructure</i> | <i>Traffic signs</i> |
| YOLO v3 | 65.04 | 45.34 | 76.18 | 92.26 | 68.62 | 73.19 |
| Tiny-YOLO v3 | 41.98 | 19.34 | 72.32 | 54.2 | 51.67 | 52.30 |
| YOLO v2 | 43.1 | 12.52 | 77.93 | 57.30 | 65.98 | 54.01 |
| Tiny-YOLO v2 | 36.37 | 15.15 | 66.29 | 44.82 | 42.75 | 59.71 |
| Faster R-CNN | 69.56 | 48.31 | 95.41 | 79.61 | 86.46 | 86.23 |
| SSD300 | 51.15 | 32.57 | 69.10 | 69.25 | 59.64 | 50.95 |
| Retinanet (ResNet50) | 84.80 | 81.53 | 81.12 | 97.14 | 76.40 | 87.80 |

Data analysis shows that the state-of-art models have interesting results, and they perform as well as in other applications. Additionally, the result images reveal a correlation between detection in images and the accuracy values. The category "ship" is the most detected object from the image corpus with the highest average precision value throughout all the algorithms. We presume this is because it has a significant unique visual feature, allowing the object detection systems to differentiate the ships from other objects better.

Generally, categories that include small objects such as riversides and traffic signs perform worse than more significant objects. Given the small size of the riverside boxes, most algorithms cannot distinguish them from other image areas. This confusion is due to the class's complexity (differences of appearances, the presence of shadows, and variations of the day-time since, for instance, illumination is usually different during the night). Such situations are difficult even with human intervention.

Table 4.5: FPS for the different detection models.

| Model | FPS |
|-------------------------------|------|
| YOLO v3 | 96 |
| Tiny-YOLO v3 | 240 |
| YOLO v2 | 120 |
| Tiny-YOLO v2 | 160 |
| Faster R-CNN | 0.48 |
| SSD300 | 4.54 |
| Retinanet (backbone ResNet50) | 7.67 |

Moreover, to evaluate the models' performance in terms of processing speed, we collected the experimental comparative results using the FPS metric summarized in Table 4.5. We noticed that unlike previously obtained results on the Pascal VOC dataset, the SSD model is not the best detector. We could easily conclude that by reducing the objects' size, their performance decreases. However, it is still among the fastest algorithms. Faster R-CNN achieves similar detection performances on the two different datasets, although it is the slowest in run-time speed. Comparing the Yolo versions, we could confirm the third version of the YOLO model's precision improvement compared to the second one. We could also confirm that the tiny versions are faster than the large ones but less accurate. Yolo v3 can be an excellent option to deal with real-time object detection since it has the right balance of accuracy and run-time speed. Although the Pascal VOC dataset results show that SSD, Yolo v3, and Faster R-CNN overcome the Retinanet model, the obtained results, using the InlandAutoDetect dataset, show the opposite. We found out that the Retinanet model outperforms all the other models in analyzing the results. **So, from the standpoint of accuracy and run-time speed, we consider Retinanet to be the best method adapted to our needs.**

Table 4.6: Different Retinanet versions performances in terms of WmAP and FPS.

| Model | WmAP | FPS |
|----------------------|-------|------|
| Retinanet(vgg19) | 82.32 | 4.54 |
| Retinanet (ResNet50) | 84.80 | 7.67 |
| Retinanet(ResNet101) | 87.06 | 4.54 |

As shown earlier, Retinanet achieves the best performance regarding mAP (85.63%) and FPS (7.67). Since it has a higher mAP than the other models, we selected it for further evaluation. As stated before, various settings for a model may have an impact on the detection results. So, we modified this selected model's backbone network (features selection) to study the influence of this modification to improve the detection performance. We obtained the results in Table 4.6. We notice a significant variation of 3% between Retinanet with vgg19 backbone and Retinanet with ResNet101 backbone. Besides, the run-time speed was affected: the vgg19 and the ResNet101 backbones slow down the model. The experimental analysis shows that the Retinanet model with the ResNet101 backbone owns the required run-time speed defined above. Hence, we selected this architecture and integrated it into the proposed system for navigable area detection, as detailed in the next section. We generated additional images of the test images by varying quality distortions such as noise and blur for each image. Blur can result when a camera is not correctly focused on the object of interest. Additionally, blur can simulate the network performance on small or distant objects captured with low resolution. For the blur, we used a Gaussian kernel and varied the Gaussian standard deviation from 1 to 5 in steps of 0.5. The size of the filter window is set to 4 times the standard deviation. Noise may result from using low-quality camera sensors. This noise can be modeled as Gaussian noise added to each color component of each pixel separately. We varied the standard deviation of the noise from 10 to 50 in steps of 5. We considered each distortion separately and classified the quality from excellent to low.

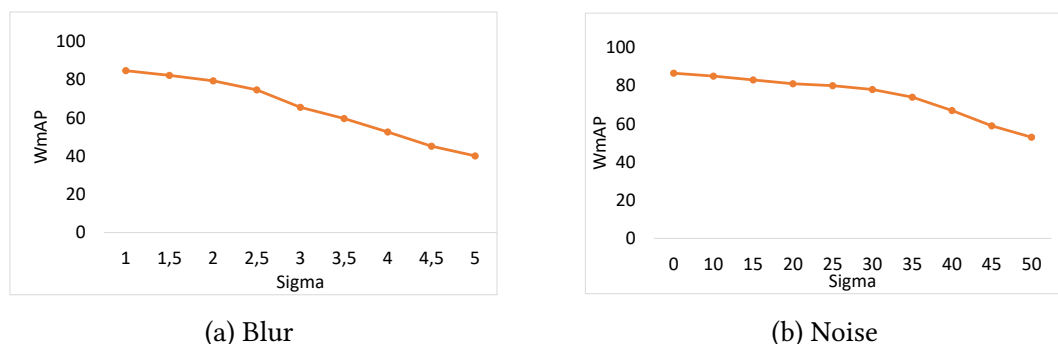


Figure 4.20: Accuracy rates under different quality distortions, using the Retinanet model.

Figure 4.20 shows the results of our experiments. We notice that the model is more sensitive to blur than noise. This reduction in performance can be explained because blur removes textures in these images. The network may be looking for specific textures to classify an image. The model is resilient to these distortions. The performance starts to decrease only at very low-quality levels. These results imply that we can be reasonably confident that our model will perform well even on low-resolution images. Experiments on our test videos have also shown that small and remote objects can be reliably detected even when using low resolution. In practice, current state-of-the-art automotive cameras have resolutions up to 8-megapixel (MP) to 2 MP [253] which implies that our proposed solution is robust in real situations.

4.6 Navigable Area Detection Performance Evaluation

Lane marking detection is a substantial part of the road scene analysis for ADAS, and the extraction of road markings is a critical element of lane position detection. In inland sailing, the pathways are unstructured roads because there are no drawn lines in the river canal but only some trees. Hence, in our case, we aim to estimate the geometric structure of the lane boundaries of an unstructured pathway on images captured by the camera to delimit the safe navigation area that allows it to navigate safely. This section explains the different steps of our proposed system and analyzes the experiment results.

4.6.1 Proposed System

We introduce a new system for road recognition that transforms it into an object detection problem. We use the Retinanet object detection model since we showed that it performs well compared to the other models in the previous section. We start by describing the proposed system’s assumptions: we assume that the camera is attached to the ship’s bow for maximum field of vision. We define the navigable area as the zone where there are no dangerous objects detected. In the first stage, the system seeks to overcome the detector’s possible negligence and ensure that all potential objects are detected. We first tested the Kalman filter to track the detected objects. We found that tracking all the existing objects in an image makes the algorithm resource-intensive, time of execution increases, and exceeds the boundary conditions. Therefore, we created a system that goes through a series of low-level image processing to guarantee that the desired information is obtained. The global system consists of mainly six steps (see Figure 4.21):

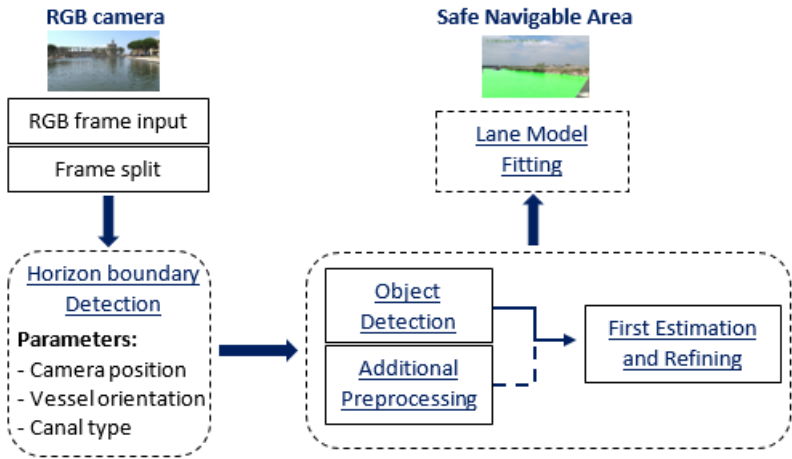


Figure 4.21: Autonomous navigable area detection system

1. *Horizon boundary of the river canal detection:* it consists in finding a border between the sky and the non-sky regions (ground, water, or mountains) given a grayscale or a

color image [254]. It depends on different features such as the camera's position, the ship's orientation, and the river canal type.

2. *Object detection*: Retinanet detector with ResNet101 as a backbone is used to generate predicted bounding boxes. Those boxes mark the objects that the ship should avoid, which are mainly riverside, other ships, infrastructure, and people.
3. *Useful points detection*: from these boxes, useful points are calculated, such as the centroids and the extreme endpoints of objects.
4. *Additional preprocessing*: to overcome the possible negligence of the detector and ensure that all potential objects are detected, we implement two techniques. First, we keep the history of detections of the last three images, and second, we filter the points with a 3-point averaging filter; this step is added to eliminate the outliers. It consists in calculating an average over the three adjacent areas of a point.
5. *First estimation and refining*: this step seeks to expand the safe zone beginning from the ship's bow to the nearest detected point (where an obstacle is detected). Then, we draw the geometric joining the points.
6. *Lane model fitting*: the last step aims to create a secure area around the objects by adding a safety distance to give detected objects a wide berth.

4.6.2 Experimental Results Analysis

We elaborate in this sub-section, the datasets and metrics used for evaluating our system. To confirm the feasibility and accuracy of our proposed method to extract the navigation zone, we performed several experiments using videos captured in real navigation conditions. Some results on the test images are shown in Figure 4.22.

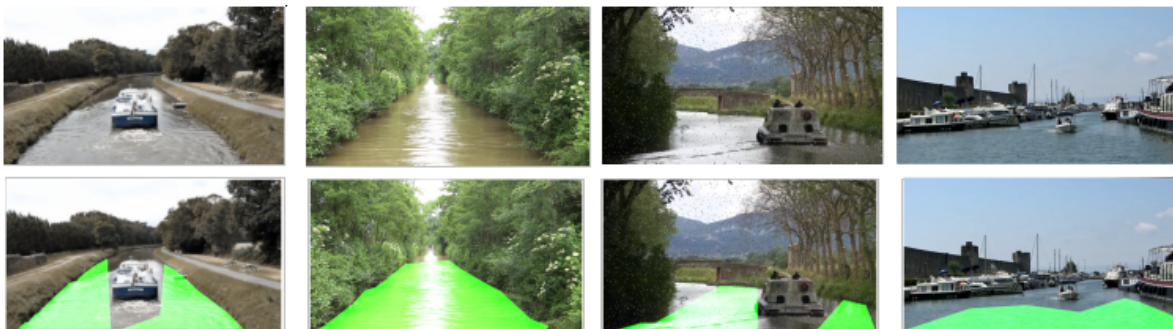


Figure 4.22: A sample from the testing set with complex conditions where the green area marks the estimated area where a ship can navigate safely.

Then, we conducted a numerical experiment to compare the proposed system's outcomes with the background truth. Since there is no method developed in the literature to evaluate the performance of such systems, we proceeded as follow: first, we manually determined the navigable area, which we call S_{man} . Then we determined the navigable area

with the system that we developed, which we call Ssys. Finally, we calculated the percentage of overlap between the two areas divided by the manually drawn navigable area.

$$Accuracy = 100 * \frac{(S_{sys} \cap S_{man})}{S_{man}} \quad (4.10)$$

where Q is the number of images in the test set.

We applied this process to five videos and 24 images, randomly chosen with different backgrounds, weather conditions, and existing objects. We found out that the average of the areas well detected over the test images is **88.13%**. The average run-time speed of all the process over the test videos is **0.8 seconds**, which is less than the boundary value calculated in subsection 4.5.1.

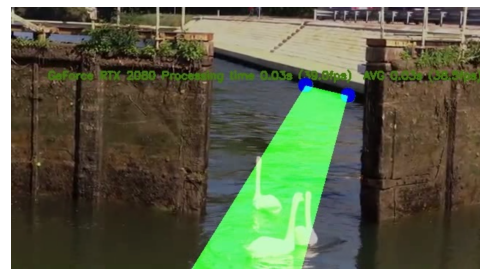
4.6.3 Discussions

The experimental results indicate that the proposed system for automatic delimitation of the navigable zone can realize road recognition and meet the ship's requirements to navigate automatically on a fluvial canal. The computation time of the proposed method satisfies the real time operation. Moreover, it is robust to noise, and it is invariant to ships' appearances.

However, the detection performances using our proposed autonomous navigable area detection system might be negatively affected by unfavorable visual conditions. We demonstrate a few such cases in Figure 4.23.



(a) Peaks due to predictions preserved from previous images.



(b) Detected area whose surface is less than the width of the ship.



(c) Very close ship sailing and unadjusted orientation of the camera.

Figure 4.23: Some false positives and false negatives navigable area detection.

In Figure 4.23a, some peaks are to be seen and which are due to predictions preserved from previous images. The system can recover from such inaccurate detections as long as the other ship moves forward. In Figure 4.23b, the detected area is accurate, but its surface is smaller than the width of the ship, so it can not navigate in this drawn area. In Figure 4.23c, the proposed model could not draw a significant area of navigation due to the unadjusted orientation of the camera. However, the detected area is still safe. All these drawbacks need to be improved in the future.

4.7 Conclusion

The fluvial environment is complex, which makes autonomous shipping in such an environment a complicated task that presents new challenges due to its specificities. In this chapter, we have addressed the specific challenge related to fluvial visual perception for autonomous navigation with the main goal of automating the delimitation of a safe sailing area. At first, we prepared a comprehensive open-source dataset, the InlandAutoDetect dataset, composed of 3,377 images. We exhaustively labeled the different objects that make up the river navigation, namely: ship, person, riverside, traffic signs, and infrastructure. Then, we investigated the performances of nine deep convolutional neural networks able to detect and recognize these objects: Faster R-CNN, SSD, four versions of YOLO, and three versions of Retinanet on the proposed dataset. The first results demonstrate the effectiveness of these models. Subsequently, we selected the most suitable architecture for a fluvial environment: Retinanet with the ResNet101 backbone as it establishes a delicate balance between precision and run-time speed. Finally, we proposed and studied a novel system for road recognition to map the safe navigation area accurately. This system is verified using different images and video clips with different backgrounds, weather conditions, and existing objects. Numerical results show that our system achieves a high accuracy detection rate of above 88%, performing in less than 0.8 seconds. These results demonstrate our system's capacity to react in real-time and validate its effectiveness for fluvial navigation.

Despite our proposed system's outstanding results, the perfect performance is still not achieved, essentially due to some missed detections. In particular, negative examples in the training set of the riverside class could be one of the reasons. Trying to annotate the riverside without leaving negative examples leads to other issues, such as highly imbalanced classes and overfitting. Therefore, we will focus on improving this part of the system by moving some classes to semantic segmentation in the future.

Nevertheless, using a single camera only is still inefficient for guiding an autonomous inland ship. Instead, other types of sources of information should be integrated. Hence, in the next part, we discuss the concept of cooperative ships sharing relevant data to enhance perception proficiency and other applications.

Data Availability

The dataset, trained models and the experimental result data used to support the findings of this study are available online for academic research purposes². The source code data are available from the corresponding author upon request.

²<https://widedhammedi.wixsite.com/phdproject/>

PART II

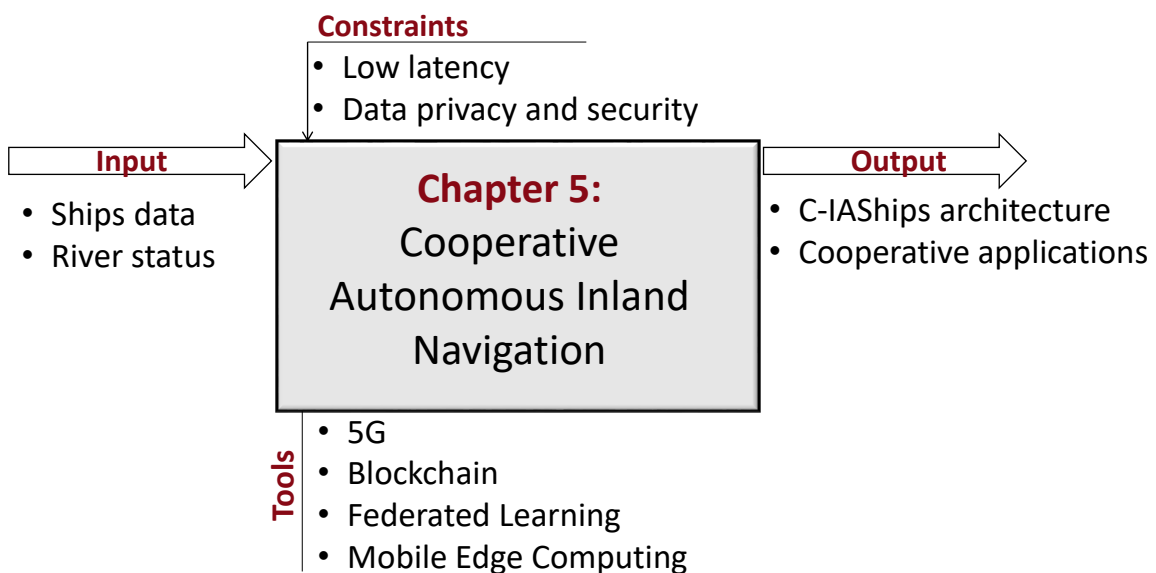
COOPERATION

Once the automation problem is properly addressed in the first part, the next logical step is to consider the cooperation between the different river entities. Thus, this part is dedicated to studying this cooperation's requirements, advantages, and achievable useful applications.

COOPERATIVE AUTONOMOUS INLAND NAVIGATION

If you want to go fast, go alone. If you want to go far, go together.

– African proverb



| | | |
|-------|--|-----|
| 5.1 | Introduction | 106 |
| 5.2 | Related Work | 107 |
| 5.2.1 | Mobile Cloud Computing (MCC) | 108 |
| 5.2.2 | Fog Computing (FogC) | 108 |
| 5.2.3 | Mobile Edge Computing (MEC) | 108 |
| 5.3 | Proposed Architecture C-IAShips | 110 |
| 5.3.1 | C-IAShips Architecture Requirements | 110 |
| 5.3.2 | C-IAShips Architecture Layers | 111 |
| 5.3.3 | Blockchain Solution for Security Enhancement | 112 |
| 5.4 | C-IAShips Architecture Advantages | 113 |
| 5.4.1 | C-IAShips Architecture Strengths | 113 |
| 5.4.2 | Cooperative Applications | 114 |
| 5.5 | Security Analysis | 115 |
| 5.5.1 | C-IAShips Architecture Vulnerabilities | 115 |
| 5.5.2 | Ships Privacy Protection | 116 |
| 5.5.3 | Security Attacks Detection | 116 |
| 5.6 | Conclusion | 118 |

5.1 Introduction

As shown in the previous chapters, Cooperative and Connected Autonomous Inland Ships (C-CAISs) continue to attract much attention. Their potential use in fret transport is expected to increase significantly in the near future. In the first part of this thesis, we focused mainly on the challenges related to automation. Specifically, we showed that ships' automation also implies the automation of the other river entities, mainly its infrastructure. Indeed, autonomous navigation can reduce the difficulty of ship control and management, reduce human mistakes, control fuel consumption, reduce costs, and increase revenue [7]. Nevertheless, autonomous ships that do not actively coordinate their actions with others may fail to fully understand their driving environment—relying only on onboard sensors to perceive their environment is the cause of several problems. First, misinterpreting the aims of the nearby ships may cause accidents [255]. Individual sensors fail to achieve high autonomy due to their technical limitations in difficult natural conditions, such as foggy or rainy days or nights. Besides, they can not efficiently divine other ships' intentions. Second, when the traffic is dense, each ship operating on its own may cause disorganization. According to French police reports [256], accidents that occur at intersections are due mainly to a misunderstanding between crossing ships or at locks when lock keepers do not get the alert messages at the right time. Third, conflicting time schedules could lead to inefficient utilization of infrastructure resources. For example, a ship traveling from Paris to Lyon spends almost half of its total journey time waiting at locks [257]. Therefore, ensuring efficient inland transport requires involving the cooperative components in the transportation process.

With the emergence of Cooperative Intelligent Transport Systems (C-ITS), several standardization organisms, e.g., 3GPP [116], are working to develop intelligent maritime transportation systems. They intend to achieve an Internet of Ships (IoS) vision while leveraging emerging technologies, including 5G and Artificial Intelligence (AI). As a particular case of IoS, the Internet of Inland Ships (IoIS) combines ships intelligence with ships networking, resulting in an intelligent river with communication and computing capabilities that provide outstanding inland transport services. However, this being said, such a cooperative environment has some critical requirements, which are discussed below, along with the technology enablers to meet them:

- (a) **Need for a low latency communication** : One of the main requirements of safety messages is to dispose of a low latency connection between not only ships but also ships and the network infrastructure. This will enable to inform ships in real-time, and thus to ensure their rapid reactions. This issue can be addressed by moving technology resources from centralized cloud computing to an edge network closer to the ships. Specifically, Mobile Edge Computing (MEC) is a new paradigm that can significantly reduce latency and avoid backhaul congestion via computation offloading and distributed content caching [258]. MEC provides a service environment and cloud computing capabilities with ultra-low latency and high bandwidth to help enable delay-sensitive and safety applications.
- (b) **Need to preserve ships' privacy**: In a cooperative process, ships need to share their identity as well as localization information periodically. If this information can

be continuously tracked, the ships' privacy will be leaked. This design may not appeal to competing companies when performing some secret tasks. To cope with this challenge, Federated Learning (FL) [259] has been advanced as an alternative solution to centralized networks, wherein each ship can maintain its private data locally and shares only processed data. At the same time, the whole process is executed collaboratively under the coordination of a central server that aggregates and shares the built knowledge among ships. This technique reduces communication overhead and avoids sharing ships' private information.

- (c) **Need to secure the shared data:** Unauthorized access to shared data can cause dangerous results, even loss of lives. Since communication channels are vulnerable to being compromised by attackers, principles of safety-critical systems must be included, and stricter criteria must be enforced. In this context, Blockchain technology has attracted growing research efforts in vehicular networks [260] [261]. It can facilitate establishing a secure, trusted, and decentralized intelligent transport ecosystem to address data sharing problems.

Therefore, in this chapter, we design a new Cooperative Inland Autonomous Ships (C-IAShips) architecture. It architecture proposes connecting several geo-distributed ships and river components to build efficient and secure inland navigation while considering its specific requirements. The architecture leverages the MEC concept, blockchain technology, and federated deep learning to deal with the challenges/requirements discussed above for effective cooperative deployment of autonomous inland-based ships.

The rest of this chapter is structured as follows: Section 5.2 gives an overview of related work where we investigate the existing types of network architectures. In Section 5.3, the cooperative inland autonomous ships (C-IAShips) architecture is described. Section 5.4 details the main advantages of the architecture. Moreover, we analyze the security proprieties that the C-IAShips architecture guarantees and discuss how it can resist the most common attacks in Section 5.5. Finally, conclusions and potential future works are discussed in Section 5.6.

5.2 Related Work

Traditionally, vehicular architectures require that the data be processed in a centralized way, e.g., a cloud data center managed by a third party. The main idea behind this type of architecture is to bring together a shared set of configurable computing resources (networks, servers, storage, applications, and services) to offer a service for multiple consumers using a multi-tenant model [262]. One of the essential benefits of using a central cloud-based solution is the minimal effort required to manage the overall system and swift possible growth, thanks to its high resiliency. Additionally, large-scale data centers can provide sufficient computing resources to serve an enormous number of ships [263].

Nevertheless, cloud computing is generally mounted on large computer data centers in various parts of the world. This centralization of resources implies a large separation between the ships and their clouds, increasing the average network latency and jitter [264]. Because of this physical distance, cloud services cannot directly access local contextual information,

such as precise ship position, local navigation conditions, and information about real-time ships' mobility behavior. For various delay-sensitive applications, such as collision detection and avoidance, these requirements (low latency and jitter, context awareness, mobility support) are needed. Thus, in recent years, various novel paradigms have emerged, such as fog computing [265], mobile edge computing [266], and mobile cloud computing [267] to address the limitations of the classical central cloud solutions. We provide throughout this section a comprehensive overview of these three different architecture paradigms, and we focus mainly on their characteristics and requirements.

5.2.1 Mobile Cloud Computing (MCC)

MCC is initially built on the notion of 'mobile delegation'. Mobile nodes with limited resources can delegate the storage of bulk data, and the execution of computationally intensive tasks to remote entities [268]. The goal of MCC is to enable the execution of rich mobile applications on a plethora of mobile devices with a rich user experience. It enhances mobile computing technologies and leverages unified elastic resources of varied clouds and network technologies. However, same as cloud computing, it is too costly to upload all content to the Internet cloud and too time-consuming to search and download content from the Internet cloud. Besides, most of the content picked up by ships has local relevance only and is best stored locally [269].

5.2.2 Fog Computing (FogC)

FogC intends to extend the cloud computing paradigm to provide computation, storage, and networking services between end devices and traditional cloud servers [270]. The fog architecture facilitates the creation of a hierarchical infrastructure, where the analysis of local information is performed near low levels. The initial definition of fog computing was revised and extended to become a new paradigm of its own rather than a mere extension of central cloud computing. With the new definition, the fog nodes can interact and cooperate [271], generating a three-tier architecture (Clients \leftrightarrow Fog nodes \leftrightarrow Central Servers) where centralized cloud servers coexist with fog nodes but are not essential for the execution of fog services [272].

5.2.3 Mobile Edge Computing (MEC)

MEC aims to "provide an IT service environment and cloud-computing capabilities at the edge of the mobile network [273]". It implies the creation of an open ecosystem, where service providers can deploy their applications across multi-vendor MEC platforms. Hence, telecommunication companies are the responsible of deploying this service environment in their infrastructure. The main advantages of deploying cloud services at the edge of mobile networks like 5G include: (i) achieving a lower latency, (ii) granting a higher bandwidth, and (iii) allowing access to radio network information and location awareness [262]. Thanks to all of these benefits, existing mobile infrastructure services can be optimized, and

novel services that are more demanding become possible. In practice, service deployment is not limited to mobile network operators; third-party service providers can also use it. MEC technology has passed the proof-of-concept stage and is being deployed in networks to enable real-time applications. Instead of receiving all files from large, regional data centers, end ships can receive data from local base stations to reduce latency and traffic in the backbone network [274]. Therefore, connected vehicles and Internet of Things gateways are among their potential applications.

To sum up, when analyzing the different paradigms' properties, we find that they all share the same primary purpose of extending cloud computing capabilities to the network's edge. However, they have underlying differences in how they want to fulfill that goal. They hence have different benefits and limitations [262]. Table 5.1 provides a comprehensible illustration of the main features of the edge paradigms (MCC, FogC, and MEC) compared to the central cloud.

| | Cloud | MCC | FogC | MEC |
|-----------------------------|------------------|------------------------------------|-----------------------|-----------------------------|
| Network architecture | Centralized | N-tier, decentralized, distributed | | |
| Deployment | Network core | Network edge, devices | Near-edge, edge | Network edge |
| Ownership | Private entities | Private entities, individuals | | Telecommunication companies |
| Hardware | Servers | Servers, devices | Heterogeneous servers | |
| Mobility | N/A | Yes | | |
| Latency, jitter | Average | Low | | |
| Local awareness | N/A | Yes | | |
| Availability | High | | | |
| Scalability | Average | High | | |

Table 5.1: Comparison of features of Edge paradigms [262].

First, on the one hand, MEC limits the deployment of edge computing platforms to mobile network infrastructures such as 5G. In comparison, fog nodes can also be positioned at other locations, such as private servers, access points, routers, gateways. However, on the other hand, MCC is more focused on the distributed nature of services and the cooperation between its devices. This fundamental difference in deployment and management of services differentiates between the three. For example, in MEC, only telecommunication operators can become MEC providers, as they own the mobile network infrastructure where the edge data centers are deployed. Nevertheless, any user (company or even resourceful end-users) can deploy their fog and MCC nodes, effectively becoming part of the service provisioning ecosystem or creating private cloud-like environments. In our study, the primary end-users, autonomous inland ships, are equipped with high-performance computing capabilities, data storage, and wireless communication devices. To deploy cooperation among them, we consider the heterogeneity of the river network, where high-speed links and wireless access technologies would coexist. Therefore, we utilize the central cloud to make global coordination and analytics, such as route planning, blocked waterway alerts, and accident warnings. We also deploy the MEC technology for

delay-sensitive services, such as collision detection and avoidance.

5.3 Proposed Architecture C-IAShops

This section introduces the Cooperative Inland Autonomous Ships (C-IAShops) architecture, connecting several geo-distributed ships and river components to build efficient and secure inland navigation while considering its specifications. We first give an overview of the architecture and its specific requirements. Then, we describe its three interacting layers and their roles.

5.3.1 C-IAShops Architecture Requirements

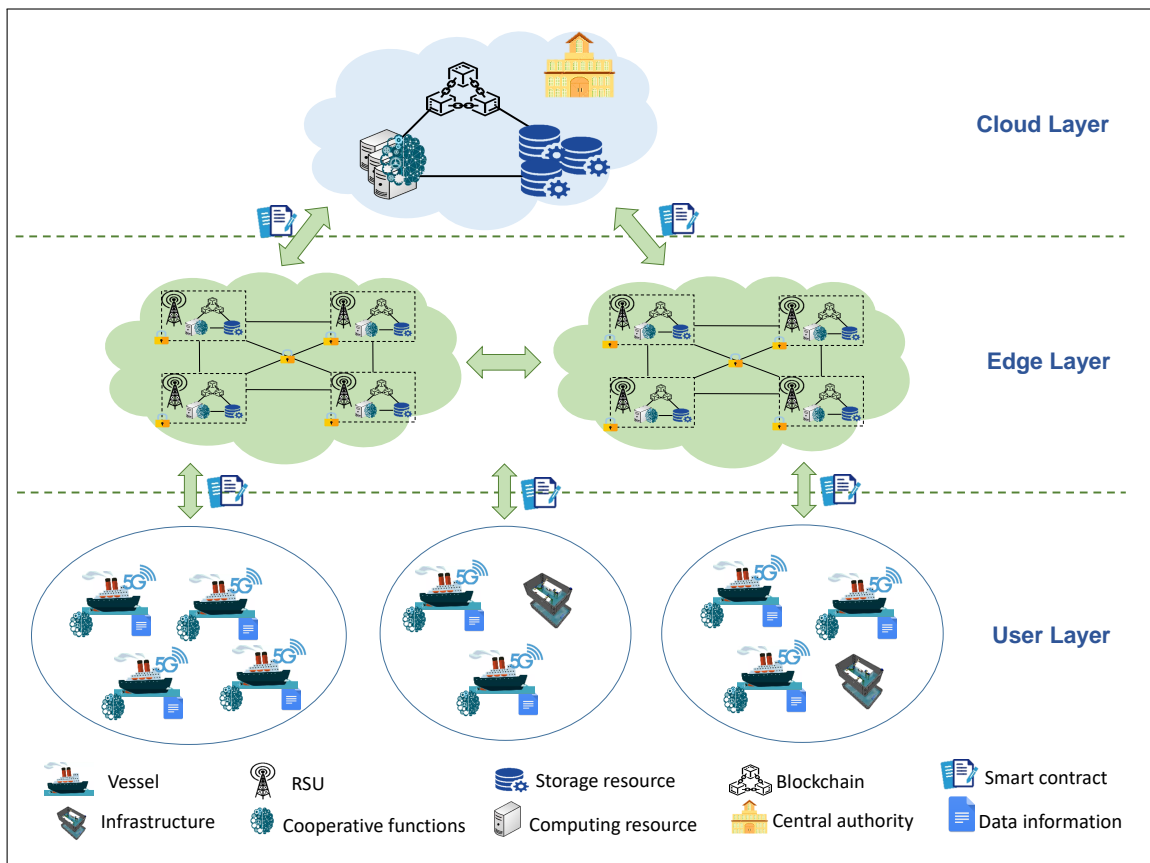


Figure 5.1: The Envisioned C-IAShops architecture is a blockchain-based federated learning architecture. It comprises three interactive layers: user layer, edge layer, and cloud layer.

As shown in Figure 5.1, the envisioned C-IAShops architecture is composed of a user layer, an edge layer, and a cloud layer. In the user layer, cooperative ships require intelligent onboard units to receive local data from onboard and external sensors, analyze and calculate the real-time traffic status, and upload the collected information to the edge layer. Roadside Units (RSUs) and MEC nodes are deployed along the rivers to support real-time

applications in the edge layer. Ships, RSU, and MEC nodes can communicate through 5G wireless short-range communication technology, which is envisaged to bring the communication era with higher reliability data transmissions and reduced delay. Edge nodes store data from ships and also deliver data to a central cloud to be managed by a central authority. The central authority is necessary for our architecture for three main reasons. First, ships can be chartered from various merchandise-owning companies for the short- or medium-term duration. Second, when a ship sail from point A to point B, the information related to this movement may pass through different systems, each being controlled by different entities such as locks, customs offices, trucking companies, and industry information portals. These entities may do not share a common IT infrastructure or have any agreed privacy and security standards. Finally, a ship's voyage is covered by a range of national, international, and private telecommunication organizations. Therefore, a central authority manages this heterogeneity in each country and establishes trusted coordination among the ships. However, wireless communication may belong to different telecommunication organizations without solid security protection, creating privacy and security concerns. Therefore, we apply the federated learning (FL) technique due to its superior performance features and benefits without leaking local data. We also use blockchain to ensure secure data transmission. The use of blockchain is because it offers greater security and creatively uses hash computing, proof of work, and distributed storage to make altering blocks nearly impossible.

5.3.2 C-IAShips Architecture Layers

The proposed C-IAShips architecture incorporates three layers: a user layer, including the different river entities, an edge layer composed of network infrastructures equipped with MEC servers, and a cloud layer managed by an administrative authority.

5.3.2.1 User layer

The bottom layer includes the autonomous ships and the different types of sensors (onboard sensors and external sensors deployed at river's edge). It has four different roles:

- **Data collection:** Data (e.g., ship type, ship position, ship speed, water level, and traffic information) is collected from various sensors.
- **Perception:** Ships equipped with onboard units process the collected data to perform some perception tasks to extract relevant environmental knowledge. Environmental perception refers to developing a contextual understanding of the environment, such as where obstacles are located and categorizing data by semantic meaning. Localization refers to the ability of the ship to determine its position with respect to the environment.
- **Each ship communicates with the nearest RSU to upload its data periodically to the edge layer.**

- Action: Ships receive command messages sent from edge or cloud nodes when needed. Command messages may be warning safety messages when a high risk of collision is detected. Cloud nodes may also send messages about the lock crossing order when multiple ships are nearing to pass a lock.

5.3.2.2 Edge Layer

The intermediate layer comprises nearby network infrastructures (RSUs), geo-deployed along the rivers, and equipped with MEC servers. Its responsibility is to perform aggregated analysis on received data from the user layer for further use. Network infrastructures can provide radio interfaces for ships to achieve seamless coverage and instant wireless communication. With computation resources, caching resources, and intelligent functions, MEC servers can provide distributed intelligent wireless computing and caching to implement computation-intensive and delay-sensitive applications, such as collision detection, safety enhancement, and emergency warning at the network edge. Edge nodes play an essential role in storing and managing ships data. However, the network infrastructures are semi-trusted because, on the one hand, they usually do not have strong security protection, which makes them vulnerable to being compromised by attackers. On the other hand, they would belong to different telecommunication organizations, which would create privacy problems. Ships, therefore, may not be willing to upload their data to them because of security and privacy concerns. To this end, we deploy blockchain when transferring information for better security and privacy protection. Blockchain can record all transactions generated in the wireless network and maintain a distributed ledger to increase security and privacy. Edge nodes can communicate and deliver the data to a central cloud through wired connections if necessary. This makes the C-IAShips architecture scalable and reliable.

5.3.2.3 Cloud Layer

The third and last layer comprises a cloud server, which can be a data center of the central authority in each country. It should be managed by an administrative authority such as Voie Navigable de France (VNF) [153] in France, which is the navigation authority responsible for managing the french inland waterways networks. The central cloud manages all edge nodes to collect the ships' characteristics, including their identification, type, weight, dimensions, order of priority, direction of travel, final destination, speed, and position.

5.3.3 Blockchain Solution for Security Enhancement

Ships may provide inaccurate or wrong messages, and MEC nodes may also fail or be hacked. These incidents affect the safety of autonomous driving of all ships. Blockchain technologies ensure transparency, decentralization, and traceability of reported data from ships, thanks to consensus mechanisms. In addition, the asymmetric encryption technology applied in the blockchain can protect the privacy of records and ships. In the circulation

of data collaboration, the blockchain monitors data flow and records the data usage activities to ensure that the behavior of the ships is authentic. The blockchain operations are summarized as follows: each ship sends a request message for uploading data to the nearby MEC node. The message contains a public key, the hash of the latest block, the current timestamp, and the data to upload. To guarantee the request messages' authenticity, messages are signed using each ship's private key. Each edge node collects and verifies local data transmitted in its coverage. All MEC nodes exchange and verify the data received, and then run the Proof-of-Work (PoW), the original decentralized consensus mechanism used in bitcoin [275]. Once the request is verified, each MEC node generates a block where the verified data are recorded and competes to find an available hash value based on parameters of the local data block. The generated block is added to the end of the blockchain. As shown in Figure 5.2, each block stores the timestamp, the hash of the current block and previous block, and the data received from different ships [260] [261].

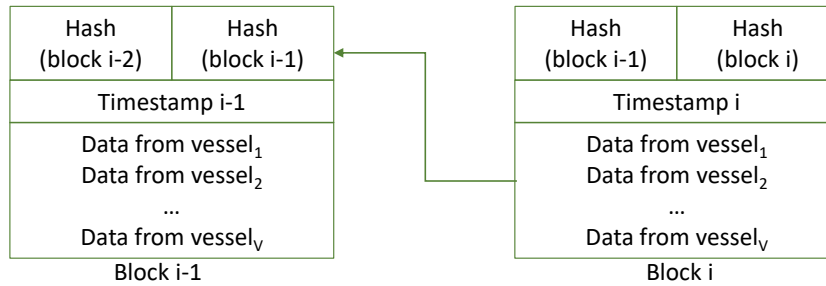


Figure 5.2: Data stored in blocks of the blockchain.

5.4 C-IAShips Architecture Advantages

The C-IAShips architecture has a multitude of advantages that make it engaging for traffic management applications. We illustrate in this section its essential characteristics and its supported applications.

5.4.1 C-IAShips Architecture Strengths

The C-IAShips architecture has a great potential to bring an extensive range of benefits, such as:

- ❖ **Low latency and effective communication:** Applying a decentralized architecture can significantly reduce the network traffic and energy consumption by sending a limited amount of data. Besides, the three interactive layers creates new opportunities for network operators and applications. First, we profit from the intelligence of the ships by implying them in some calculations. Second, using an edge layer allows better support for low-latency applications by placing them near their users. Third, it avoids high data traffic since only the necessary control information is sent to the cloud layer.

- ❖ **Geo-distribution and location awareness:** The C-IAShips architecture is not centralized. Instead, it can be deployed anywhere and can provide distributed services and applications. Hence, it supports location awareness, i.e., MEC nodes can be deployed in diverse locations.
- ❖ **Safety and security considerations:** Since the Internet of Inland Ships (IoIS) connects several geo-distributed ships and network infrastructures from different vendors, building an efficient and secure communication network is crucial. Our proposed architecture handles this issue using two complementary solutions. On the one hand, employing a federated learning technique can collaboratively train a learning model on their local data without revealing it to a centralized server. This secure aggregation ensures privacy preservation of data in the local updates. On the other hand, we utilize smart contracts, which are self-executing scripts that reside on blockchains and allow distributed multi-step processes. Our smart contract-based architecture enables data management automation with high efficiency and defends against second-hand data sharing without authorization.
- ❖ **Robustness and effectiveness:** Ships are frequently and unpredictably offline, on slow connections, or they are allocated with insufficient communication resources [276]. However, C-IAShips architecture uses Federal Learning, which does not rely on synchronization among all the ships. Hence, even when some ships lose connectivity, they can still build their local models.
- ❖ **A wide range of applications:** The C-IAShips architecture enables some cooperative applications to make river waterways safer, more efficient, and more environment-friendly. We investigate three cooperative application in the next section.

5.4.2 Cooperative Applications

Compared to individual ships, greater efficiency, operational capability, and more powerful applications can be realized by cooperative ones. Here, we discuss three such applications: Cooperative Lock Scheduling (CLS), Cooperative Route Planning (CRP), and Cooperative Collision Detection (CCD).

5.4.2.1 Cooperative Lock Scheduling

River locks are used for elevating and lowering ships between low-level and high-level waterways and are considered vital inland navigation nodes. The scheduling method of crossing the locks strongly influences the ships' safety and economic efficiency. Hence, our architecture considers improving the scheduling method using cooperative lock scheduling. In the first step, cooperative ships send their characteristics, including their weight, dimensions, order of priority, final destination, speed, predicted arrival time, and position to the cloud node. Then, the central can make schedules according to the data received while keeping an eye on the state of the locks, such as availability and waiting time. In

return, the scheduling order impacts ships' decision-making on departure time and speed choices. Chapter 7 presents a detailed exploration of this application.

5.4.2.2 Cooperative Route Planning

The principal purpose of ships is to transport goods and persons from one place named origin to another named destination. Therefore, when sailing in waterways, ships usually follow predetermined geometric global paths. Traditional individual route planning uses self-state information to create optimal trajectories, considering some constraints, such as voyage time optimization [277], fuel emissions reduction [278], weather constraints [279]. The ship here has an only objective, path following, considering that it is the only ship in the waterway. Such a technique is beneficial for each ship apart; however, it may cause conflicting situations when paths cross simultaneously. Cooperative coordination between ships helps to avoid these challenges. On the one hand, optimal global trajectories can be determined, as the central coordinator in the cloud node has complete information about the whole fleet of ships and achieves low cost, avoidance of delays, and efficient utilization of resources. For instance, ships can coordinate their voyage details to avoid congestion. On the other hand, iterative communications between the central coordinator and the ships create an agreement among ships to choose alternative routes if accidents or congestion occur in a specific area during the voyage.

5.4.2.3 Cooperative Collision Detection

Non-cooperative collision avoidance methods do not consider the communication between ships. On the contrary, each ship tries to predict other ships' actions, either by assuming that others have a constant speed [135] or according to holonomic or kinematic models [136]. Information exchange ensured via the C-IAShips architecture among the cooperative ships can provide additional information to help the central nodes make collision detection decisions. Details of this application are provided in Chapter 6.

5.5 Security Analysis

Since the C-IAShips architecture is essentially based on wireless communication channels, it should cope with some privacy and security threats caused by its complex and geo-distributed network environment. In this section, we define our architecture vulnerability by identifying the main malicious intruders. Then, we study how it can preserve ships' privacy and discover the well-known security attacks.

5.5.1 C-IAShips Architecture Vulnerabilities

The malicious attackers can be categorized into two types: passive and active attackers. The first type holds honest-but-curious attackers, and it aims to read the data transmitted

without modifying it. It mainly challenges the ships' privacy and is complicated to detect passive attacks since they do not affect normal functioning. The second type holds dishonest attackers, and it aims to destroy or alter the accurate data transmitted or influence the regular functioning with inadequate data. These two types can intervene in three different levels in our C-IAShips architecture, as illustrated in Figure 5.3:

- ❖ **Malicious Server:** Honest-but-curious servers can inspect ships' updates without altering the data. In contrast, a potentially dishonest server can inspect the updates and tamper with the data. The server here can be either the cloud server or one of the edge servers.
- ❖ **Insider Villain:** Honest-but-curious ships can track the other ships, while dishonest ships can send incorrect data to the server. These attackers are also known as byzantine attackers [280].
- ❖ **Outsider Villain:** When communicating the updates between trusted ships and servers, eavesdroppers on the communication channels can manifest. They snoop the IDs of real ships and pretend to be authorized ships. Then, they participate in the communication rounds either at the training or the deployment phase.

The C-IAShips architecture implements different techniques to address these vulnerabilities and achieve a high level of security protection, as we explain in the following sections.

5.5.2 Ships Privacy Protection

Although federated learning was first invented to ensure rigorous privacy protections by preventing data sharing during the training phase, recent researches [281] [282] prove that the transmission of the model updates can still reveal sensitive information about ships. Additionally, ships are also required to expose their sensitive data periodically to satisfy the RIS standard. Therefore, we use consortium blockchain to preserve privacy concerns by its asymmetric encryption and smart contracts. The automatic execution of smart contracts allows only authorized ships to send their data. This data is anonymous, and encrypted, and attached with digital signatures. In practice, ships use random pseudonyms as public keys to replace their original address, making the blockchain nodes unable to track the location of specific ships. Hence, the transparency characteristic of blockchain during data sharing avoids second-hand sharing without authentication, and the anonymous operations using pseudonyms during data sharing bring data privacy protection.

5.5.3 Security Attacks Detection

We discuss here our architecture ability to detect and prevent the well-known malicious attacks and provide security guarantees.

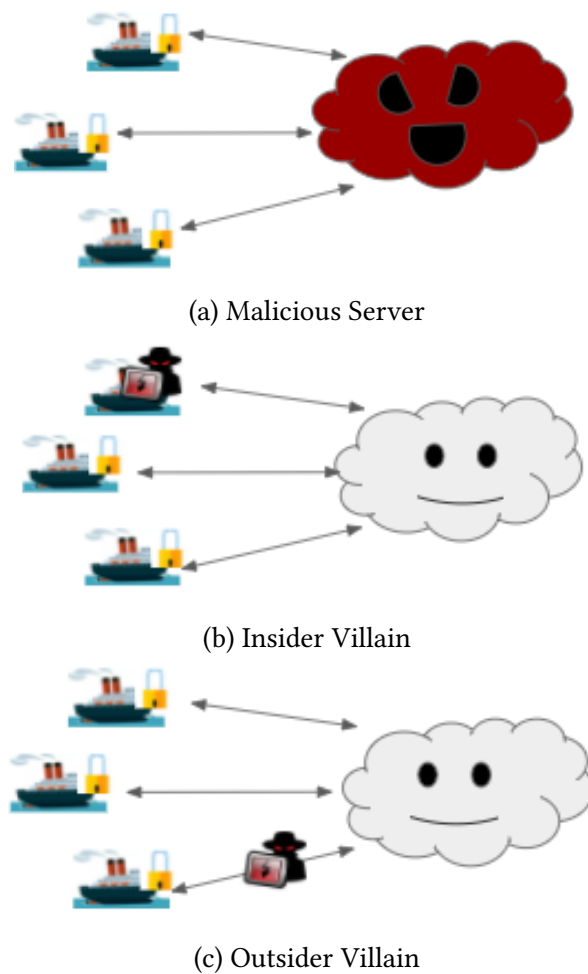


Figure 5.3: Different malicious actors can attack C-IAShips architecture: malicious server, insider villain, and outsider villain.

5.5.3.1 Fake Ships Attack:

As described above, the only ships capable to send information are considered valid only if their private keys exist in the blockchain managed by the legal authorities. A ship trying to create multiple entities must generate valid blocks for those entities, which is unfeasible in our architecture.

5.5.3.2 Model Update Poisoning

The intruder can work independently or within a group of malicious ships. Model update poisoning attacks directly corrupt the global model, by modifying the weights of the resulting local model before submitting it for aggregation. However, it cannot control the aggregation algorithm used to combine ships' updates into the global model, nor the benign ships' training tasks. However, this approach does not work against federated learning [283]. Aggregation removes most of the poisoned models' contribution, and the global

model quickly ignores the poison. For instance, in [284], they conduct an experiment where they add some random noise generating local models. They observe that even when adding 1000 copies of the sample to the training set, the attack is ineffective at causing targeted poisoning in the global model. This occurs because the malicious agent's update is scaled. Additionally, in our architecture, we use a random sub-sampling, in which each communication round, the edge server selects a random subset of ships to share with the global model. Thus, the attacker needs to be very selective, and then the poisoning is missed or at least significantly prolonged. This solution is used in [285], where they confirm that it does not impact the global model performance.

5.5.3.3 Data Poisoning (also known as byzantine attacks)

Throughout the normal process, one or more ships, who correctly behaved in previous communication rounds, may lately act maliciously and poison the global model. In our case, such a villain can send wrong information, such as wrong location and ship characteristics. Data poisoning attacks are one of the Federated Learning vulnerabilities [286]. The support of a central orchestrator in our architecture is handled by blockchain. In such practice, local models are shared and verified in the blockchain network while providing rewards to the ships. However, for the sake of people's safety, even if we hesitate about the honesty of some ships, we still process their data. We prioritize human security over data security in this stage of the study. This threat can nevertheless be resolved by adding the reputation mechanism in smart contracts. This mechanism aims to validate if a ship is malicious by studying its behavior. Smart contracts are triggered to block malicious activities by punishing their owner ships (reducing their coin balance), and setting flags to stop their future activities. For instance, we can compare the parameters each ship returns at each training round. These parameters should converge to optimum value and gradually stabilize after some training iterations. Ships returning values with high variance, inconsequent training epochs, probably misbehave. The edge server monitors the parameters returned from each ship for each training iteration and decides upon the criteria for any misbehaving ship.

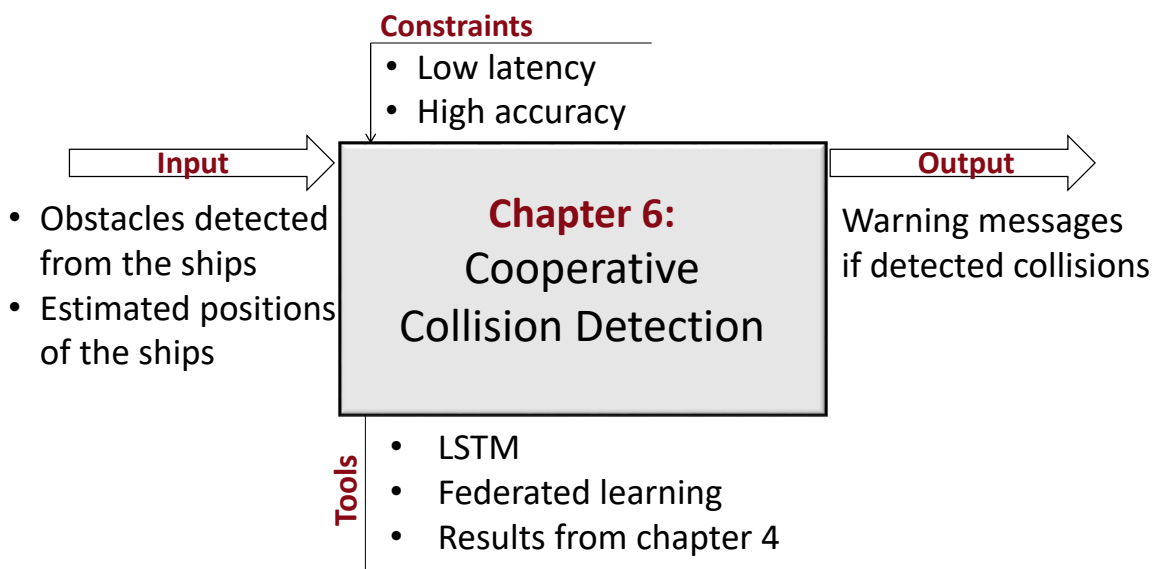
5.6 Conclusion

Optimizing the performance of inland transport systems requires involving the cooperative components in the transportation process. Compared to individual waterway entities, cooperative entities can achieve greater efficiency and more robust applications. However, cooperation brings new challenges mainly related to low latency responses, shared data privacy, and security protection. This chapter innovatively introduces the CIAutoShips, a blockchain-based federated learning architecture for cooperative intelligent inland transportation. Different applications can be performed in the different layers of our architecture; latency-sensitive applications on the edge layer and latency-tolerate applications on the cloud layer. The CIAutoShips architecture can achieve high communication efficiency and protect the privacy of ships from being leaked while realizing the different proposed applications. In the following chapters, we discuss two such applications, namely cooperative collision detection and cooperative lock scheduling applications.

COOPERATIVE COLLISION DETECTION

You don't need to know the whole alphabet of Safety. The A, B, C of it will save you if you follow it: Always Be Careful. .

– Colorado School of Mines Magazine



| | | |
|-------|---|-----|
| 6.1 | Introduction | 122 |
| 6.2 | Related Work | 123 |
| 6.3 | Problem Formulation and Proposed Solution | 125 |
| 6.4 | Federated Learning-based Collision Detection System | 126 |
| 6.4.1 | Main Modules Description | 126 |
| 6.4.2 | Cooperative Collision Detection System Training | 129 |
| 6.4.3 | Cooperative Collision Detection System Deployment | 130 |
| 6.5 | Performance Evaluation | 130 |
| 6.5.1 | Simulation Scenarios | 131 |
| 6.5.2 | Mobility Prediction Module Analysis | 132 |
| 6.5.3 | Federated Learning Benefits | 134 |
| 6.5.4 | Cooperative Collision Detection Efficiency | 138 |
| 6.6 | Conclusion | 140 |

6.1 Introduction

The inland shipping industry is growing continuously, and its traffic is becoming denser in many navigable waterways [287]. This increasing traffic volume of inland ships imposes a growing necessity to introduce new measures, tools, and solutions to improve efficiency in inland transportation. In this context, cooperative autonomous ships can achieve far more meaningful benefits in optimizing river ship control and management, reducing costs, controlling fuel consumption, and increasing revenue. Therefore, in the previous chapter, we introduced the Cooperative Inland Autonomous Ships (CIAships) architecture, which uses promising technologies to enable various cooperative applications.

Ship collision detection is an essential and fundamental concern because of its high frequency and its serious consequences in terms of property, equipment and human lives [288]. At early ages, researchers aimed to develop navigational assistance systems for enhancing situational awareness of human operators as humans are at the core of collision avoidance. However, many maritime accident investigations [289, 290] estimate that 75–96% of marine accidents involve human failures. Accordingly, autonomous ships have recently gained remarkable attention, focusing on solving collision problems by autonomous systems. Nevertheless, autonomous but not cooperative and connected ships, relying only on onboard sensors to perceive their environment, may fail to achieve high autonomy. Individual sensors have technical limitations in difficult natural conditions, such as foggy or rainy days or nights. Besides, they cannot efficiently guess other ships' intentions. Therefore, achieving high safety in inland transport requires involving the cooperative components in the transportation process.

In this chapter, we design a new inland ships collision detection system based on the CIAships architecture. It consists of continuously gathering data, such as localization data, from ships and processing them at the MEC level to predict collisions. Then, alerts will be sent to ships to avoid collisions between them. The main contributions of this work are summarized as follows:

- With the lack of real datasets corresponding to inland vessel mobility, we generate our simulation dataset based on accurate river maps. We select different graphs from the densest French river network using Overpass turbo [291], an online data mining tool for OpenStreetMap. Then, we use the Simulation of Urban MObility (SUMO) software to generate mobility data of ships and adjust ships density. The produced dataset is open-source, and it is available online for other researchers working in this area.
- We evaluate the accuracy and performance of four state-of-the-art deep learning models for ships' mobility prediction in an inland environment. We select the ConvLSTM model among the evaluated models since it is the most efficient and most robust. We implement the federated learning method to perform collaborative learning among ships without leaking their private data.
- We design a cooperative collision detection system to accurately detect collisions and alert concerned ships sufficiently in advance through analyzing the position of a fleet of ships using the ConvLSTM model findings. The process is deployed at the

MEC to ensure low latency communication and guarantee real-time reaction to avoid collisions between ships.

- We evaluate the effectiveness of the collision detection system in the inland environment using the proposed dataset and suggest possible improvements.

The remainder of this chapter is structured as follows: Section 6.2 gives an overview of related works where we investigate the existing collision avoidance systems. In Section 6.3, we formulate the problem statement and explain the proposed solution briefly. Section 6.4 explains cooperative collision detection and its two phases: training and deployment. Discussions about the results and performance evaluation are drawn in Section 6.5. Finally, conclusions and potential future works are discussed in Section 6.6.

6.2 Related Work

In order to prevent collision accidents, it is crucial to determine if there is a collision risk in advance accurately. Therefore, the ship collision detection problem has attracted continuous interest from researchers, and many efforts have been developed to handle this issue. In the literature, the collision prevention process and its information flow in the manned and unmanned ships can be abstracted as in Figure 6.1.

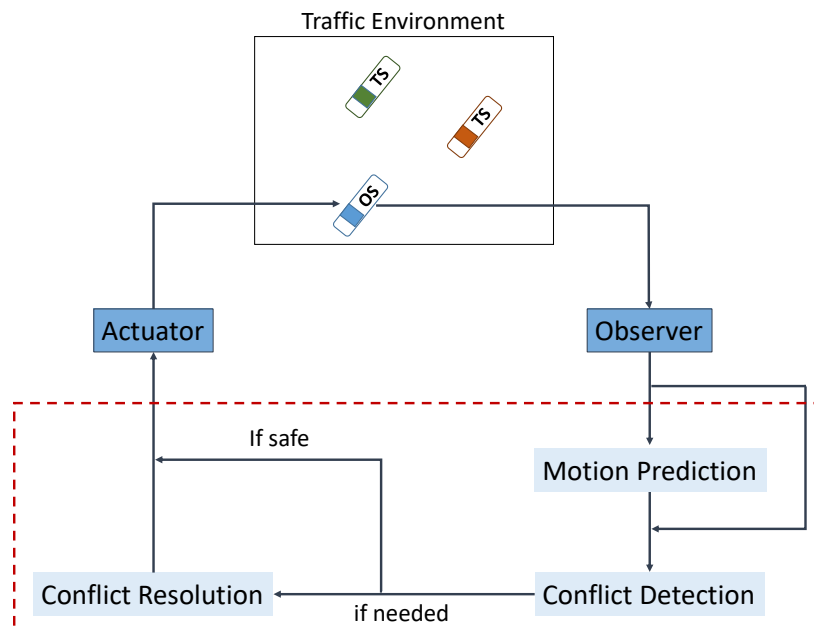


Figure 6.1: The collision avoidance process (adapted from [292]).

Five components are included [293]: (1) Observer, which contains various sensors offering data to support other modules; (2) Motion prediction module forecasts the future actions and trajectories of the Own Ship (OS) and the Target Ships (TSs), which is the basis for conflict detection and resolution; (3) Conflict Detection module, which evaluates the possibility of a collision and, if necessary, issues collision warnings; (4) Conflict Resolution

module, which determines the evasive solutions, and (5) Actuator, which implements the solutions.

The OS can forecast future actions and trajectories based on specific assumptions or through communication with the TSs. Hence, we can divide collision avoidance methods into five categories based on communication and cooperation levels, as shown in Table 6.1.

| | Cooperation level | | |
|-------------------|--------------------------|---------------------|--------------------------|
| | Non-cooperative | Cooperative | Competitive |
| Non-communication | Assumption based methods | Rule-based methods | Game theoretical methods |
| Communication | Intention-aware methods | Negotiation methods | |

Table 6.1: Collision avoidance methods' categories based on communication and cooperation levels

Traditional collision avoidance methods usually do not consider the knowledge transmitting and acquiring among the ships. **Assumption-based methods**, such as potential field [294], and velocity obstacles [295], act as if other ships sail with constant speed and heading [296] or according to holonomic or kinematic models [136] to predict their actions. However, those assumptions may not be accurate, making the methods unsuitable for collision prevention in more general collision avoidance cases. Moreover, most methods only provide one solution. Consequently, if those methods function as navigation assistance for human-operated ships, the operators are forced to follow the proposed solution blindly without understanding the decision processes.

Rule-based methods use a set of predefined rules to guide collision avoidance. For example, when a ship encounters other ships, the ship will turn 75 degrees to the starboard side [297] or enlarge rudder angle until the trajectory is collision-free [298]. However, a single rule cannot handle all kinds of encounters in a dynamic environment. Thus, multiple rules should be considered. As mentioned earlier in Section 2.2.3.3.1, the COLREGs are the most widely used rule in maritime navigation. They set out the navigation rules to be followed by ships to prevent collisions between two ships. So, when ships encounter each other, the controllers reorganize the encounter pattern and execute actions accordingly to comply with the corresponding rule. An overview of methods that consider COLREGs is provided in [299]. However, since the enumeration of rules for all scenarios is impossible, this method does not guarantee collision-free. If a case is not studied in advance, this method might not find out a proper solution. In practice, COLREGs heavily depend on human common sense in determining rule relevance, mainly when multiple rules apply simultaneously.

Moreover, the rule-based method relies on the assumption that all of the ships follow the rules. However, ships in the same situation may have different recognitions of the encounter pattern, or other ships may violate the rules. Cho, Han, and Kim [300] consider the probability of rules' violation to propose a probabilistic collision avoidance decision-making approach based on a graphical model consisting of system states' maneuvering intent and evolution.

Competition between vessels is rarely mentioned in existing studies. Miloh [301] design the problem of collision avoidance between two vessels as a pursuit-evasion game between a faster elliptical pursuer and a more maneuverable circular evader. Hoogendoorn et al. [302] present a method to model the decision-making process of the human operators according to the expected behavior of the TS. This method aims to include human

decision-making in comprehensive simulation models to describe the movement of vessels, including external effects due to wind, current, and waves; waterway geometry; and the interaction with other vessels. Lisowski [303] apply the differential game control systems for collision avoidance, considering the uncertainty of information and incomplete knowledge about other objects. However, it is challenging for this method to handle the encounter situations, which involve more than two players.

In **intention-aware methods**, ships can decide their collision avoidance actions according to the intentions transmitted by other ships. MIYAKE et al. [304] propose an intention exchange support system called the “Automatic Navigational Intention Exchange Support System using AIS (ANIESS).” ANIESS is a communication system to exchange navigational intentions (e.g., port-to-port passing) between encountered ships by using short-range communication means such as the application-specific message of automatic identification system (AIS). However, such a solution is not applicable in real situations as anti-collision is not a one-time activity. It is not reasonable that the ship makes a series of decisions from origin to its destination at once and carries them out one after another. This consideration is not only impractical but also makes the problem unnecessarily complex. In real situations, ships make decisions stepwise in real-time based on their observations and judgments on whether there is collision risk with ships. Therefore, in [305], Zhang et al. anti-collision decisions are made in a distributed way where each ship makes decisions according to its observations and the intentions of the other ships. In practice, each ship can only access its sensors and actuators. Then, each ship first broadcasts its intentions, such as turning and trajectories to ships within the communication range, and decisions are made based on the broadcast information. Ships perform computation and broadcast their intentions in a predetermined sequence. Since information is exchanged after each ship solves its problem only, the quantity of communication between ships and the computation time is limited.

On the other hand, **negotiation methods** emphasize closed-loop information exchanges. After the OS has broadcast its decision, other TSs’ actions based on this decision are sent to the OS as feedback. The OS will thus adjust its decision accordingly. In this way, agreements among the ships can be achieved through iterative negotiations. A ship can broadcast its own intentions and expectations about other ships, such as the actions that it wishes other ships would take and the trajectory it prefers rather than the trajectory it computes. When a controller makes decisions, it considers other ships’ expectations and adjusts its decisions. Thus, such an iterative negotiation framework has a greater potential to achieve overall optimal performance [306, 307].

Nevertheless, those methods allow ships to coordinate their behavior only for a limited time, for instance, when they identify a high collision risk. They assume that a surrounding region should be kept clear between ships. They focus mainly on ships’ size and shapes under stable conditions. Ship domain is treated as a circle with radius R , and other ships’ intrusion must be avoided.

6.3 Problem Formulation and Proposed Solution

Solving the collision detection problem is a matter of determining whether the ship is in danger and when to take evasive actions. Thus, in order to effectively prevent collision accidents, detecting a collision risk must be **accurate** and **rapid** to allow the ship to react

sufficiently in advance.

Unlike the existing works described in the previous section, this study measures the collision risk from a global and cooperative perspective on a central MEC node instead of each OS. The core of this process relies on information exchange among the cooperative ships to provide additional information to help the central nodes make collision detection decisions. To meet the requirements of the collision detection, two main conditions should be established. On the one hand, the information exchange should dispose of a low latency connection between ships and the network infrastructure to ensure rapid reactions from ships. The C-IAShips architecture addresses this issue since it is based on MEC technology and federated learning, as explained in the previous Chapter 5.

On the other hand, provided information should be timely and accurate. Hence, in this study, we fuse two data sources. First, we employ the camera to perceive the surrounding environment and detect possible obstacles. Second, we use the GPS to obtain accurate ships' positions and estimate their future positioning.

6.4 Federated Learning-based Collision Detection System

The C-IAShips architecture uses the federated computing method, where ships handle their locally distributed datasets and send processed data to the MEC server. The information interaction between ships and edge/cloud servers makes other ships get more information, so more aware of them. Figure 6.2 depicts an overview of the system model of the collision detection application. Each ship has three main modules on the user layer: the ship mobility prediction module, the obstacle detection module, and the MEC identification module. The first module allows the ship to predict its future position based on its GPS data history. The second one consists in detecting eventual obstacles from its camera. The third module uses the ship's future position to decide to which MEC node it should send the results of the two first modules. The collision detection application is deployed on all the MEC nodes on the edge layer, covering a large geographical area with low-latency access. When receiving data from all the ships, the MEC decides whether there is a risk of collision or not.

6.4.1 Main Modules Description

In the following subsections, we detail the different modules constituting the cooperative collision detection system.

6.4.1.1 Ships Mobility Prediction Module

This module is placed on each ship to predict its future coordinates based on the accurate coordinates and speed history obtained from the onboard sensors. We may regard ship mobility as a time series of multi-dimensional data, which Recurrent Neural Network (RNN) is able to model [308] [309]. The RNN is suitable to analyze the driving data, owing to its

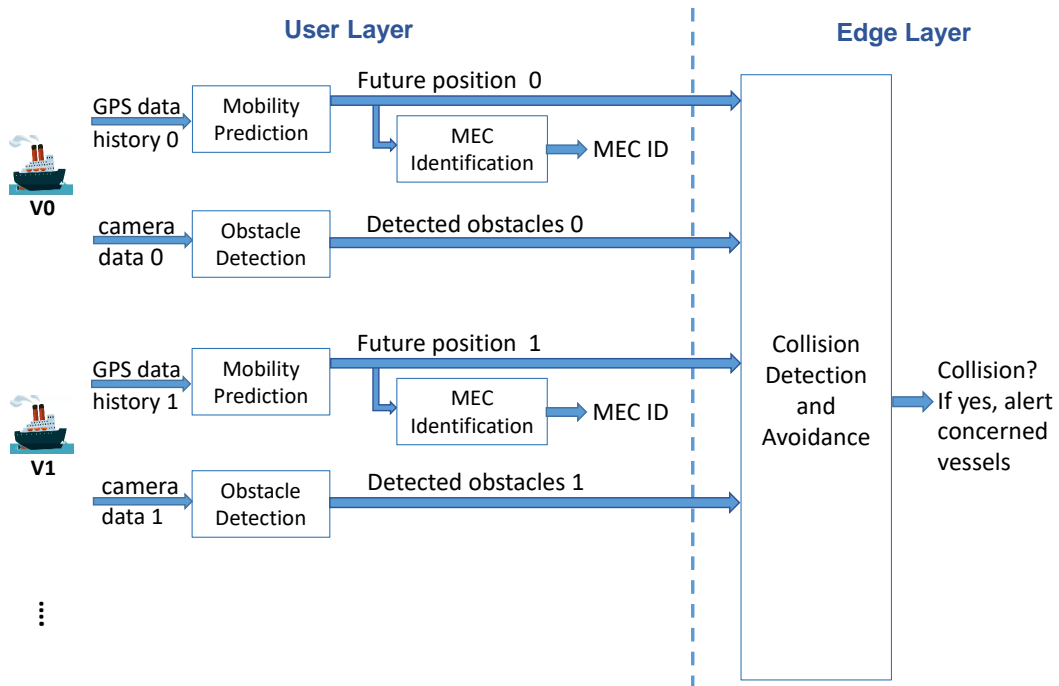


Figure 6.2: Cooperative collision detection system model

advantages of learning the temporal features of sequential information. Long Short-Term Memory (LSTM) [310] is a particular RNN model. It was proposed to solve the problem of gradient dispersion in the RNN model, which is dedicated to processing the time series data. Due to the strong ability of self-learning [311, 312, 313], the LSTM network is suitable for constructing predictive models for ship mobility prediction tasks. Therefore, in our study, we compare four different variants of LSTM algorithms, namely:

- ❖ Vanilla LSTM: It is interpreted as the original LSTM block with the addition of the forget gate and peephole connections [314]. It has a single hidden layer of LSTM units and an output layer used to make a prediction.
- ❖ Bidirectional LSTM (BiLSTM): It consists of concatenating the interpretations of two models, one taking the sequence of the input provided in a forward direction and the second in a backward direction [315]. BiLSTM, therefore, increases the amount of information available to the network.
- ❖ CNN-LSTM: A convolutional neural network (CNN) is a neural network developed for working with two-dimensional data [316]. The CNN LSTM is a hybrid model that involves using CNN layers for feature extraction on input data combined with LSTMs to interpret the feature extracted and support sequence prediction [317].
- ❖ ConvLSTM: It is a recurrent neural network introduced by Shi et al. in [318] to solve spatiotemporal sequence forecasting problems. The ConvLSTM differs from CNN-LSTM in that, for CNN LSTM, the convolution structure is only applied as the first layer, and sequentially LSTM layer is applied in the second layer. In contrast, in the ConvLSTM, the convolutional reading of input is built directly into each LSTM unit.

6.4.1.2 Obstacle Detection Module

This module is processed simultaneously with the ship mobility prediction module. The obstacle detection module of autonomous ships plays an essential role in safe navigation since the ship needs to detect and avoid nearby ships and infrastructure. We propose to use the obstacle detection system proposed in Chapter 4, which consists of detecting, classifying, and tracking objects in the near real-time fluvial domain. Inland obstacles are river edges, other ships, fishers, swimmers, or people by the riverside, and infrastructure such as locks and bridges. After comparing different perception algorithms of object detection, we found that the most suitable algorithm is the Retinanet algorithm with a resnet101 layer as a feature extractor [240]. We have 87.06 % as the accuracy of detection with 4.54 Frames per second (FPS). The output of this module is the position and the category of the possible detected obstacles.

Figure 6.3 draws a sample of predicted results where the rectangles and text overlaid on the figures are the outputs generated by obstacle detection module.



Figure 6.3: A sample of predicted images where the rectangles and text overlaid on the figures are the outputs generated by obstacle detection module.

6.4.1.3 MEC Identification Module

Figure 5.1 shows that each MEC node covers a defined geographical area. Therefore, ships should know their location to know with which MEC node they will communicate. We only use a basic function to identify the nearest MEC node based on the ship's future position, and we do not consider the MEC capacities. If there are many edge nodes in proximity, the load situation, quality of service, or other parameters may be regarded for node selection.

6.4.1.4 Collision Detection Module

This module uses a fusion of data received (predicted future ship positions and obstacle detection information) from all the ships to decide on the risk of collision. This module estimates the potential dangers of the collisions between ships at a particular instant of time. For this purpose, a minimum authorized safety distance (D_{min}) must be considered. The navigation authority responsible for managing the french inland waterways networks [153]

indicates that "Powered ships traveling at speed more than 6 knots must keep a minimum distance of 30 meters from other ships and structures, and 60 meters from people in the water." The MEC host calculates distances between the ships using the data received. Based on D_{min} and calculated distances, we can determine whether there is an imminent collision or not. If one distance is less than D_{min} , the concerned ships are in a collision risk region, and warning messages are sent to the corresponding ships. The algorithm runs indefinitely in iteration within the edge node.

6.4.2 Cooperative Collision Detection System Training

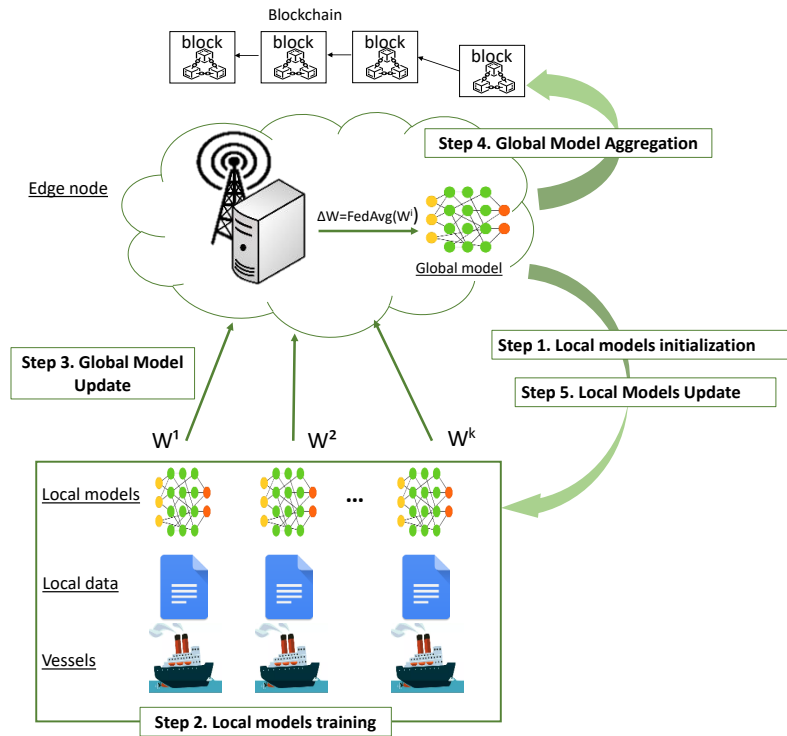


Figure 6.4: Collision detection: training phase steps.

As mentioned before, this system is based mainly on ships' mobility prediction module, which was trained in a federated way. Figure 6.4 illustrates the mobility prediction training process. It comprises five main steps:

- (i) **Step1: Local models initialization:** The federated learning (FL) server placed in the edge node generates an initial global model G_0 . Then, it broadcasts the G_0 training hyperparameters to the participating ships.
- (ii) **Step2: Local models training:** Each ship V_i starts to collect new data and update parameters of its local model L_i , based on the global model G_j , where j is the current iteration index. Then, each V_i tries to find optimal parameters minimizing the local loss function.

- (iii) **Step3: Global model update:** The local trained models are periodically uploaded to the FL server. Only the model parameters or gradients are transmitted instead of the raw data.
- (iv) **Step4: Global model aggregation and consensus:** When receiving the local models from ships, the FL server aggregates all verified local models and packages them into consensus blocks. Once consensus is reached, the block will be added to the end of the blockchain. Finally, the FL server sends the updated model parameters back to the ships. There are several algorithms to do the aggregation. We use here the Federated Averaging (FedAvg) algorithm, which is a baseline algorithm presented in [319]. It was shown that it is capable of achieving high accuracy, communication efficiency, and privacy preservation.
- (v) **Step5: Local models update:** G_j periodically pushes the updated local model parameters to connected ships.

The learning process is iterated for many communication rounds until the global model can converge to the global optimal, a threshold level is reached, or a desirable training accuracy is achieved.

6.4.3 Cooperative Collision Detection System Deployment

Once trained, the ships are now able to predict and share their positions, and the system can detect eventual collisions, as shown in Algorithm 2. Each ship executes the ship mobility prediction module, the obstacle detection module, and the MEC identification module. Then, it sends its local predicted position and local detected obstacles to the identified MEC. Each MEC performs the collision detection module and sends warning messages to the concerned ships when a collision is detected. It is assumed that all the MEC nodes have sufficient onboard resources to execute the given learning tasks at any instance of time t . It is worth noting that the collision detection application is instantiated in all the MEC nodes. In addition, each MEC node serves the ships located within a specific area. Thus, if a ship moves from one area to another, it will be served by the collision detection application instantiated in a new MEC node.

6.5 Performance Evaluation

The proposed cooperative collision detection system can detect unpleasant situations proactively to make inland transportation safe and reliable. In this section, we evaluate its performance in order to validate it. We first define the used simulation scenarios. Then, we focus on its first module, the mobility prediction module, by comparing the algorithms mentioned above to select the best one to use later. Next, we show through a comparative experiment the benefits of federated learning. Finally, we evaluate the effectiveness of the whole collision detection model under challenging situations.

Algorithm 2 Collision Detection: Deployment Phase

-
- 1: **Input** For each ship: local position, local detected obstacles
 - 2: **Output** For each MEC node: Detected collisions, warning messages
 - 3: Initialise the configuration of all instances of the collision detection application on the MEC nodes.
 - 4: **while** True **do**
 - 5: **for** each ship **do**
 - 6: *Mobility prediction*: Receive a batch of GPS coordinate from the local data and predict the future position of ship for epoch $t + 1$.
 - 7: *MEC identification*: Based on the future position, decide the future recipient MEC.
 - 8: *Obstacle detection*: Process camera data to detect obstacles (ships, persons, river-side, or infrastructure).*
 - 9: Send results data to concerned MEC nodes
 - 10: **end for**
 - 11: **for** each MEC node **do**
 - 12: Receive data information from ships
 - 13: Execute the *collision detection* module and send warning messages to the concerned ships if a collision is detected.
 - 14: **end for**
 - 15: **end while**
-

6.5.1 Simulation Scenarios

Due to the lack of real datasets from industrial systems, especially localization information in a fluvial environment, we produced our simulations scenarios based on accurate river maps. The settings of the main mobility-related parameters are reported in Table 6.2. We use Overpass turbo [291], an internet data mining tool for OpenStreetMap that allows executing different API requests and presents the result on an interactive map to select our simulation maps. The selected maps are composed of a segment of inland waterways of the Nord and Pas-de-Calais region, with the densest canals in France. Then, we use SUMO to generate mobility data with a ship as the vehicle type and adjust each simulation's circulating ship's number. SUMO is an open-source, highly portable, microscopic, and continuous multi-modal traffic simulation package, designed to handle large networks. The produced dataset is open-source, and it is available online ¹ for other researchers working in this area.

The ships can navigate in both directions except at intersections. The ship spatial density and speed vary in our scenarios, while other parameters, such as the maximum speed, are fixed. The duration of the simulation step in SUMO is the frequency at which the simulator updates the position of the simulated ships. In our case, we set it to 5 s, which allows us to have a maximum positioning error due to the discrete-events nature of the simulator. As a result, we have 4 instances of simulation as follows:

¹<https://widedhammedi.wixsite.com/phdproject/>

Table 6.2: Simulation parameters

| Parameter | Value |
|--------------------------------------|---|
| Simulator | SUMO |
| Total number of mobility simulations | 4 |
| Vehicle category | ships |
| Number of ships | 30, 50, 70, 100 |
| Maximum Speed | 15 km/h |
| Data Output type | speed, angle, x, y, z, lane, slope |
| Simulation time | 100000 seconds |
| Step length of simulation | 5 seconds |
| Simulation maps | 60*60 km ² 40*40 km ² 30*20 km ² |

- Instance 1: Small network scale with a small number of ships 30*20 km² 30 ships.
- Instance 2: Medium network scale with a high number of ships 40*40 km² 70 ships.
- Instance 3: Large network scale with a high number of ships 60*60 km² 70 ships.
- Instance 4: Large network scale with a very high number of ships 60*60 km² 100 ships.

6.5.2 Mobility Prediction Module Analysis

Unlike the movement of sea ships, river ships are forced to maneuver in confined spaces constantly (e.g., the presence of a large number of possible obstacles, frequent course changes, a large number of oncoming and passing ships). We compare in this section the effectiveness of the four different variants of LSTM algorithms mentioned in the ship mobility prediction module (see section 6.4.1) in terms of mean squared error (MSE) and in terms of stability. In statistics, an estimator's mean squared error (MSE) measures the average of the squares of the errors — that is, the average squared difference between the predicted and accurate values. The lower it is, the greater is the estimator. MSE is calculated as follow:

$$MSE = \frac{1}{n} \sum (y_i - \tilde{y}_i)^2,$$

where n is the size of data points, y are the accurate values, \tilde{y} and are the predicted values.

A sequence of driving data is utilized as inputs of the four algorithms. We note that the hyperparameters of each algorithm (number of epochs, batch size, and number of neurons) are tuned to determine the optimal mean squared error (MSE) between the predicted position and the actual position while avoiding over/under learning, using the Keras Tuner [320]. Moreover, we try different window sizes with variations in the window overlapping percentage to determine the optimum size with optimum MSE. Finally, the input window size is fixed to five, and the output sliding window is fixed to two sample predictions. To

compare the stability of the algorithms, we proceed as follows: (i) From each instance, we extract the position data history (HPbi) of each ship V_i separately, (ii) For each ship V_i , we split its HPbi into train/test sets, (iii) For each ship V_i , we train each model on the train set and calculate the MSE_i on the test set, and (iv) For each instance, we calculate the AvMSE. $AvMSE = \text{Average} \left(\sum_{V_i} MSE_i \right)$, where the V_i are the ships that belonged to the instance. We plot the distribution of the AvMSE for different instances in Figure 6.5.

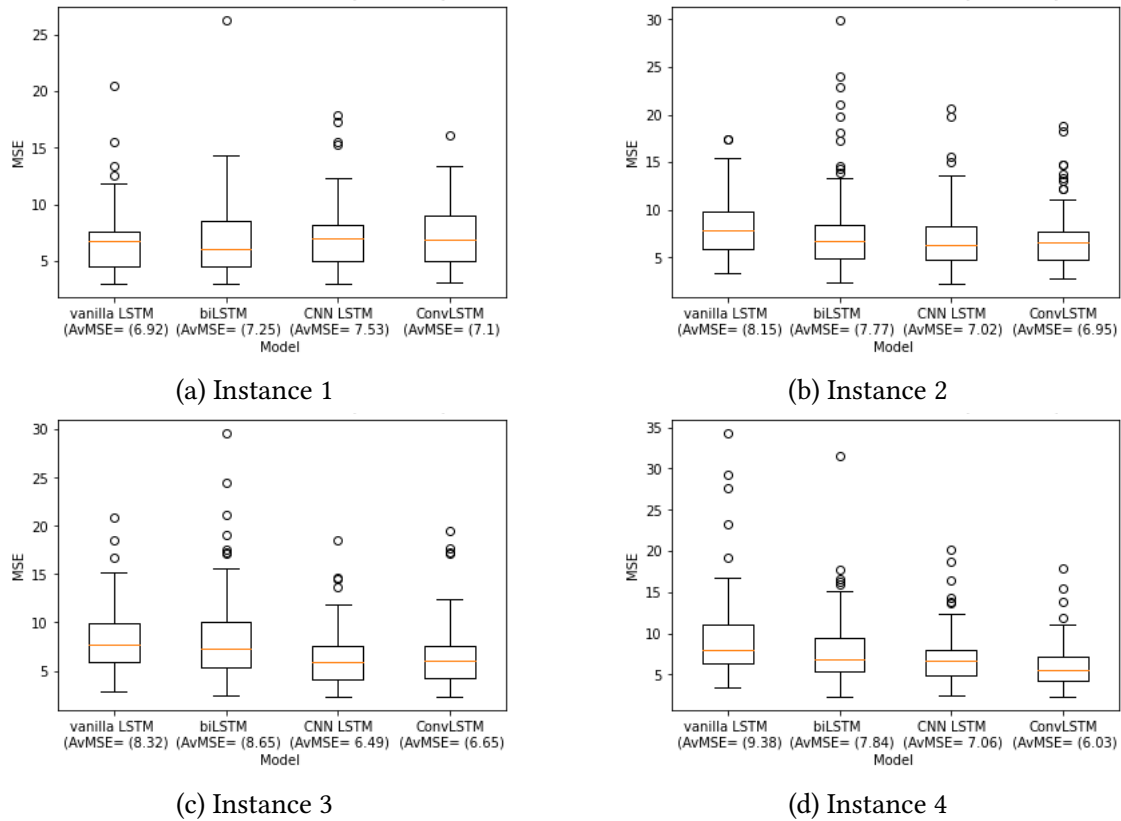


Figure 6.5: Boxplots of AvMSE value on the test set representing the stabilities of the models at various instances.

These results confirm that the four models can correctly predict the ships' mobility. We find that there is no one best model to find the smallest AvMSE. We notice that increasing the size of the network in itself does not affect the performance of the models. However, while assessing the boxplots of the four neural networks, two significant peculiarities appear: on the one hand, we find that the vanilla and bidirectional LSTMs often exhibit extreme outliers (i.e., a very high MSE), which means that they are not suitable for long-term use due to the lack of stable prediction quality, and therefore it is impossible to guarantee the safety of the ship's motion. On the other hand, the CNN LSTM and ConvLSTM represent minor variations, which implies more robustness. In addition, the ConvLSTM has more minor variations and optimal AvMSEs. Several reasons can explain the differences: 1) Our input data is ship mobility data characterized by simultaneously changing in time and space. 2) The LSTM input takes positions directly at each timestamp. However, CNN-LSTM and ConvLSTM take a fully connected layers-based observer network. The latter provides more data and relationships to the network. Convolution-based models also use

historical observations and their expected output to extract correlations between the observation and the output. The relationship is encoded as a condition tensor. This design allows these models to more easily extract the trend of trajectory changes and capture the spatiotemporal dependencies of the input data. 3) The characteristics extracted from the convolution layers are joined to form more robust features affected to a lesser degree by noise, spatiotemporal variations, and disturbances in the acquired mobility data. The relationships recognized between the CNN layers are exploited to consider the lost spatial information in the deeper layers. **In the following, we use the ConvLSTM as a ship mobility prediction model.**

6.5.3 Federated Learning Benefits

This section conducts comparative experiments among classical and federated learning in terms of convergence, least MSE, and execution time.

6.5.3.1 Definitions

We first briefly outline the difference between the three learning paradigms: centralized learning (CL), distributed learning (DL), and the federated learning (FL):

- ❖ **CL:** In centralized learning, the ships are connected with the central server to upload their data. In particular, the ships upload their local data to the edge/cloud server, and the central server performs all the computational tasks to train the data. The advantage of this type of learning is that the model can generalize based on data from all the ships. However, this is extremely hard to ensure. First, the bandwidth is often limited, and the sheer amount of data that may exceed the reasonable limits can create communication overhead. Moreover, latency due to communication to the cloud is often a barrier for an application operating in real-time. Finally, ships' private data is highly at risk as the cloud server can be malicious or infer through adversaries.

- ❖ **DL:** Distributed Learning avoids the evoked problems of CL. The distributed models are trained with the same methodology as centralized machine learning models, but they are trained separately on multiple ships without needing a central server. In practice, this simplifies the task since each model only needs to understand its local data and not how it varies compared to all other ships. Another benefit of DL is that the internet connection does not constrain Learning and that no confidential information needs to be transferred to the central server.

- ❖ **FL:** In federated learning, ships build their models collaboratively using their local datasets. In essence, FL trains algorithms across decentralized edge devices while holding data samples locally. In particular, local epochs are declared in the learning parameters, and each ship trains the data by running the local epochs. After specific epochs, the local update is computed, and the ships send the updates to the central

server. The central server receives the update from each participant, average them and aggregate the next global model. Based on this global model, the ships execute the training process for the next communication round.

6.5.3.2 Training Phase Convergence

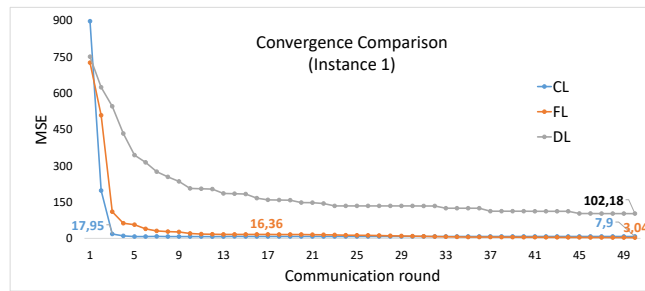
The training phase is performed each time a new ship comes, at the beginning of the journey, and then regularly (each week) to update the models with the new data. Therefore, the convergence speed of learning is an essential issue for our application. We compare federated learning with distributed learning and centralized learning with the same ship mobility prediction model. We assume that the server has unlimited bandwidth and that the ship nodes have identical upload and download bandwidths. To calculate the necessary time, we suppose that the ships execute their calculation in parallel. We use the theoretical speed of 5G: the commercial speeds announced today vary mainly from 100 Mbit/s to 1 Gbit/s in reception and from 50 Mbit/s to 250 Mbit/s in transmission. Figure 6.6 shows the training process of the three learning techniques for different network characteristics, and Table 6.3 reveals their training time.

We can first observe that all the methods converge towards an optimum but not at an identical speed. FL converges less fast than the CL but faster than the DL. The CL (5 rounds) and FL (15 rounds) manages to adjust their global models from the first rounds. This can be explained by the fact that the ships are independent of training their models in DL. The CL has all the data so that it can converge quickly. FL adjusts its parameters quickly due to the involvement of the central server and the parameters aggregation process.

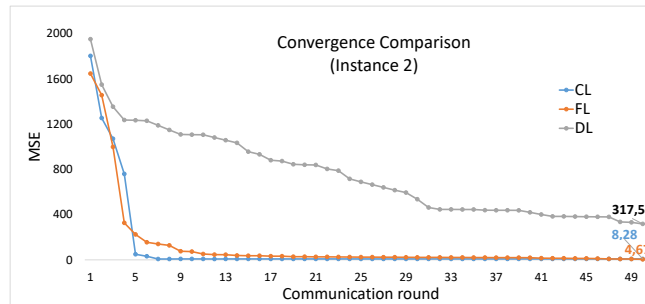
| Computing Method | Characteristic | Training time |
|------------------|--|---------------|
| FL | 15 communication rounds, sending local model weights, and receiving global weights | 2h13 |
| DL | 10 communication rounds, sending local data, and receiving global weights in the end | 4h2 |
| CL | 50 communication rounds, no sending nor receiving | 9h19 |

Table 6.3: Training time for the 3 different learning techniques

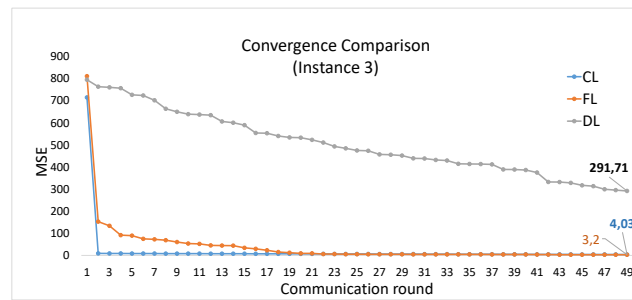
Furthermore, results prove that the efficiency of FL can approach the centralized computing method. As seen from Figure 6.6, the loss during the training of federated learning is lower than the centralized method, and the final MSE is less in most cases. FL performs the others because it initializes its models with aggregated weights of the overall model, and it has several models that run in parallel. However, DL, not taking advantage of the results of other models, fails to find an optimum MSE in a reasonable time, as shown in Table 6.3.



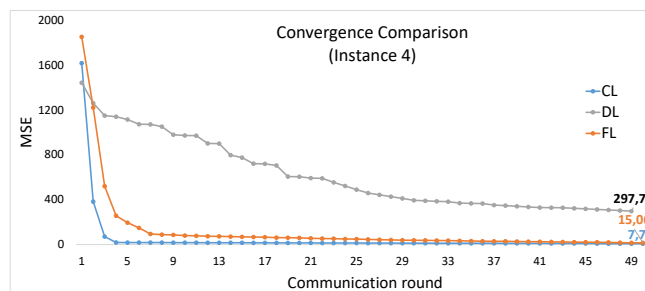
(a) Instance 1



(b) Instance 2



(c) Instance 3



(d) Instance 4

Figure 6.6: Training Phase Convergence of the three different learning techniques vs. different network characteristics.

6.5.3.3 Mean Squared Error (MSE) Comparison

In this series of experiments, we compare performance of the three learning methods on the test set. We train them with the same number of learning hyperparameters, till the convergence towards an optimal MSE (10, 15, and 50 communication rounds for the CL, FL,

and DL, respectively). All these experiments are carried on Google colab Pro T4 GPU from Google. The results are the average value of 5 simulation runs.

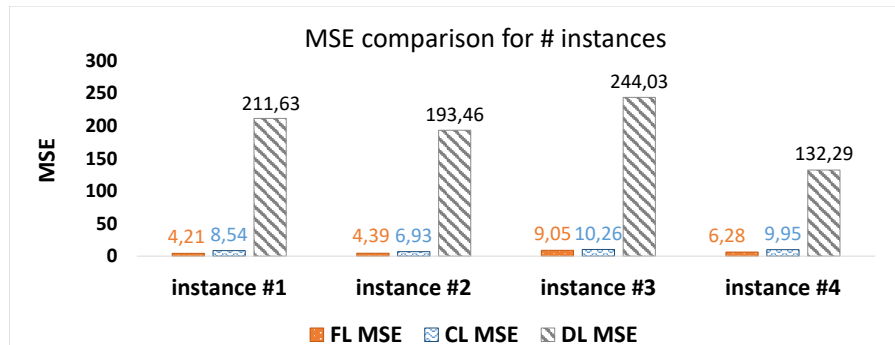


Figure 6.7: MSE comparison

As for evaluation by test datasets, Figure 6.7 shows the overall results. It indicates that federated and centralized learning perform almost similarly in terms of MSE. The federated learning performs slightly better than centralized learning, which proves the practical value of our method.

6.5.3.4 Main findings

To conclude about FL advantages, we can deduce the following main findings:

❖ Advantages of FL over DL:

- Better performance: MSE minimized.
- Fast training convergence: FL does not make several communication rounds due to its efficient aggregation of information.
- In DL, each ship generates its model based on its unique data history not seen by other ships. So, local data remains decentralized, unrepresentative, and not identically distributed from all data.
- If a new ship comes on the network, its integration in FL will be faster and more efficient.
- Ship data can vary in size due to the different environments they pass through. The non-interaction between them may cause widely varying volumes of training data. For instance, we take a scenario where a ship has data of just 1 hour of navigation and another three days. The 1st ship will take too much time to adjust its model with DL. Meanwhile, with the FL, it will be able to benefit from the experience of the 2nd ship and adjust its model easily.

- With FL, we can choose only a limited number of ships representative of the entire flow. Therefore, we can reduce the calculation time and the use of the communication channels while guaranteeing good performance.

❖ Advantages of FL over CL:

- Even though the CL can converge in a practically realistic time, it uses massive communications resources. The FL achieves the same convergence rate as the CL, but outperforms it by avoiding the saturation of communication channels and excessive energy use.
- In CL, the MEC node will be submerged with model computation, and thus, it will not be able to respond to real-time events for other latency-sensitive applications.
- CL technique implies more data security and privacy concerns.

6.5.4 Cooperative Collision Detection Efficiency

In this subsection, the effectiveness of the cooperative collision detection application in terms of collisions avoided is tested using two main verification scenarios: multi-ship encounters into open busy river waterways segment sailing and multi-ship encounters in busy intersections areas.

6.5.4.1 Performance Metrics

We consider True Positive (TP) as the correct positive predictions. False Positive (FP) are the incorrect positive predictions. True Negative (TN) are the correct negative predictions, and False Negative (FN) are the incorrect negative predictions. We use the following standard metrics to evaluate the performance of the collision detection method:

- **Precision (Pr)** = $TP / (TP + FP)$
- **Sensitivity (Sn)** = $TP / (TP + FN)$
- **Specificity (Sp)** = $TN / (TN + FN)$

As implied by the above equations, precision is the ability to identify only the relevant collisions. Sensitivity shows the ability to detect all collisions. Specificity is the proportion of the true negatives correctly identified. It measures the ability to identify normal navigation (no collision).

6.5.4.2 Open Busy River Waterways Scenario

This series of experiments consists of detecting collisions on open busy river waterways. In this case, the results are averaged over a single simulation campaign, 5 runs, speed fixed to 15 km/h, and different densities of ships per kilometer.

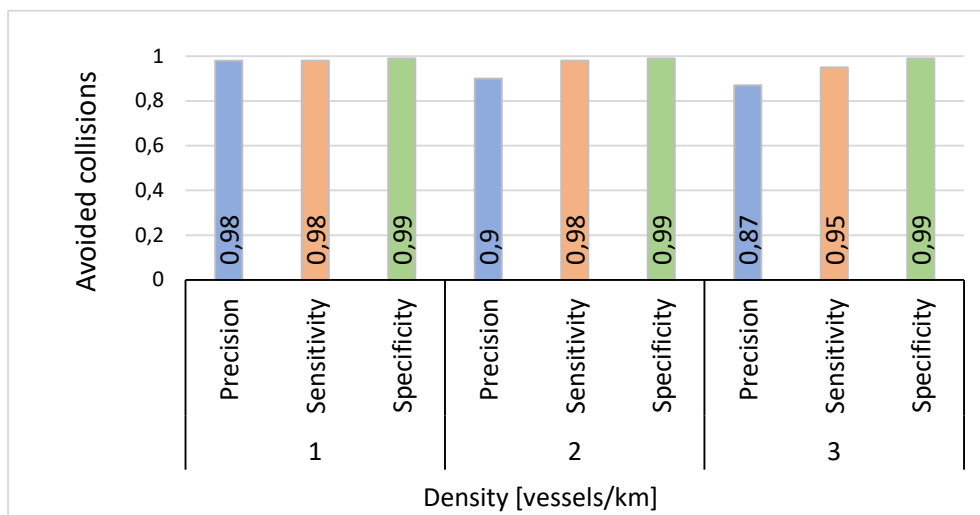


Figure 6.8: Percentage of collisions avoided. In the x-axis, the ship spatial density is plotted; each bar corresponds to a different performance metric.

In Figure 6.8, we show the different metrics to measure the avoided collisions. The x-axis represents the ship's spatial density, and each bar corresponds to a different performance metric. In the first experiment, where density equals 1 ship/km, we found 177 TP, 20 FP, 2 FN, and 4801 TN. In the second experiment with a density equal to 2 ships/km, we find 152 TP, 18 FP, 7 FN, and 4823 TN. The density in the third experiment is 4 ships/km, and it has 235 TP, 4 FP, 3 FN, and 6758 TN. We see that we reach almost 100% of collisions avoided for all densities. Moreover, we remark high precision, high sensitivity, and high specificity. This means that the system returns many results, with all results detected correctly. The ship spatial density directly impacts the final results: the higher it is, the lower the final number of collisions avoided due to the more challenging, complicated configurations that may arise in crowded waterways.

6.5.4.3 Busy Intersections Areas Scenario

Unlike the open river areas, the collision at the intersections has more constraints and higher risks since only one ship is allowed to pass most of the time. Thus, we call such zones dangerous zones, and they are usually in channel entrances or at intersections. An example of such an environment, where red circles surround danger zones, is plotted in Figure 6.9.

The results seen in Table 6.4 are obtained with 5 simulation runs, each 30000 seconds long, with a speed fixed to 15 km/h.

Results show a precision equal to 0.91, a sensitivity equal to 0.92, and a specificity equal to 0.99. These values prove that the model has a high precision, which means its ability to trigger an alarm only for an actual risk of collision, and a high sensitivity, which means its ability to detect all the risks of collision. However, there is still a low rate of FN (a collision not detected) and FP (a normal passage detected as a collision). Even if these two indicators



Figure 6.9: Danger zones are surrounded by red circles

| | Actual “collision” | Actual “no collision” | |
|--------------------------|--------------------|-----------------------|-----------|
| Predicted “collision” | 180 | 16 | Pr = 0.91 |
| Predicted “no collision” | 14 | 5790 | |
| | Sn = 0.92 | Sp = 0.99 | |

Table 6.4: Collision detection results on busy intersections areas

are expensive, scary, and even painful, the FN remains the most dangerous since it can be fatal.

6.6 Conclusion

This chapter proposes a collision detection system application based on federated deep learning to ensure safety in inland navigation. The proposed system is implemented on the Cooperative Inland Autonomous Ships (C-IAShops) architecture, previously defined in Chapter 5. Numerical results show that our system achieves a high accuracy collision detection rate above 92% in different circumstances. Furthermore, we demonstrated the trade-off between the low latency and accuracy due to the use of the MEC technology and the federated deep learning technique. These results demonstrate its effectiveness for fluvial navigation.

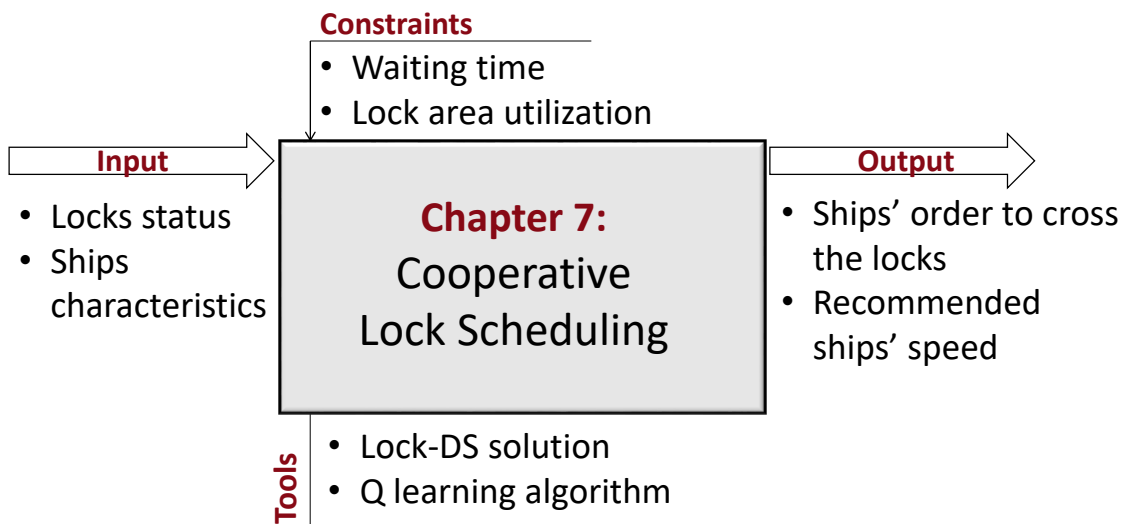
However, despite our proposed system’s outstanding results, the perfect performance is still not achieved, essentially due to some missed detection and position inaccuracy. To reduce the potential for erratic ship behavior, a larger vision should be set. Thus, in the future, we will focus on improving this part of the system by integrating other sources of information, such as Lidar, to ensure more comprehensive sensing capabilities of ships. Additionally, at this stage, we cannot omit the presence of a human operator onboard or remotely to intervene in emergencies and avoid crashes.

In the next chapter, we address another application of cooperative fleet management, cooperative lock scheduling. We propose a novel scheduling method based on the C-IAShops architecture to reduce the fleet’s waiting time.

COOPERATIVE LOCK SCHEDULING

The key is not to prioritize what's on your schedule, but to schedule your priorities.

– Stephen Covey



| | | |
|-------|--|-----|
| 7.1 | Introduction | 143 |
| 7.2 | Lock Scheduling Related Works | 144 |
| 7.3 | Problem Formulation | 145 |
| 7.3.1 | Proposed Solution | 146 |
| 7.3.2 | Problem Parameters and Assumptions | 147 |
| 7.4 | Dynamic Lock Scheduling (Lock-DS) Description | 148 |
| 7.4.1 | Lock-DS - Stage 1: Collect Ship's Characteristics | 148 |
| 7.4.2 | Lock-DS - Stage 2: Estimate Ship's arrival time and lockage duration | 149 |
| 7.4.3 | Lock-DS - Stage 3: Schedule Ships | 149 |
| 7.4.4 | Lock-DS - Stage 4: Optimize Ships' Speed | 154 |
| 7.5 | Performance Evaluation | 155 |
| 7.5.1 | Performance Evaluation Metrics | 155 |
| 7.5.2 | Numerical Experiments | 155 |
| 7.6 | Conclusion | 156 |

7.1 Introduction

Locks are considered vital inland navigation nodes. They maintain a water level suitable for navigation while allowing ships to overcome the resulting differences in water level. Locks also provide a way to detour obstacles such as waterfalls or dams with hydropower generation facilities. Occasionally, locks may also serve to control the water flow on a river or act as a barrier for flood protection. Figure 7.1 shows two locks: the first lock elevates three ships, whereas the second lock is empty.



Figure 7.1: Example of locks

Due to the time needed to operate these locks, they constitute a natural bottleneck along rivers. In particular, the arrangements made concerning the operating times of one or multiple locks determine to a large extent the total delay or waiting time incurred by ships traveling through a waterway. Therefore, the scheduling method of crossing the locks has a strong influence on the improvement of the channel and economic efficiency [321]. As briefly discussed in Section 3.2.3, the currently used scheduling method in France river locks is First Come, First Served (FCFS), prioritizing urgent ships such as the police. In practice, lock operators get the position of navigating vessels based on observations or Very High Frequency (VHF) radio communication, then decide for passage order by ranking the ships using the above simple rules. However, it is challenging to keep the scheduling safe and effective simultaneously with the inland traffic growth. Thus, the staffs usually make decisions, where they would sacrifice some efficiency for more safety [322]. Therefore, their method does not ensure good time optimization for the whole system.

Moreover, in order to avoid long waiting queues, some ships may speed up to arrive early and wait at the anchorage so they can get served first. This behavior leads to a longer waiting time before the locks. It increases the operational cost for these ships due to the higher fuel consumption caused by maintaining a higher speed [323]. In contrast, some ships may arrive late to avoid waiting for a long time at the lock area. However, they may pass through the lock later than their preferred departure time; hence, they may not meet their scheduled arrival time. Therefore, this chapter considers improving the scheduling method using an automatic cooperative lock scheduling based on the CIAShips architecture previously explained. We propose the Dynamic Lock Scheduling (Lock-DS) solution to efficiently manage ships scheduling at locks by minimizing their waiting time and optimizing their speed. Reducing the ships' waiting time at locks implies minimizing the waiting time of all ships in the system. We discern three main contributions of this work, summarized as follows:

- We propose the Lock-DS, a cooperative lock scheduling system, to sufficiently order

the ship’s passage through the locks. The process is deployed at the central cloud to ensure a global view of the system. The central node can make schedules according to the data received while keeping an eye on the state of the river, such as locks availability and ships’ estimated arrival times. In return, the scheduling order impacts ships’ decision-making on departure time and speed choices.

- We adjust the ships’ arrival times by adjusting their speeds according to their order. Hence, we can reduce their fuel consumption.
- We evaluate the effectiveness of the Lock-DS while considering near real-world conditions and suggest possible improvements.

The rest of this chapter is structured as follows: We start with studying existing research works on lock scheduling 7.2. Then, we formulate the problem by first showing how our solution addresses these limitations 7.3, before describing the problem parameters and assumptions. Afterward, we detail the proposed solution and its four stages in Section 7.4. We show and analyze the simulation results in Section 7.5. Conclusions and potential future works are discussed in Section 7.6.

7.2 Lock Scheduling Related Works

Scheduling consists in planning some tasks of a machine to achieve some goals subject to certain constraints. Lock scheduling has recently attracted increasing attention. We give in this section a general overview of the existing related works.

Since the efforts required for solving this problem depend mainly on the lock configurations, we divide the research of lock scheduling into four categories: on the one hand, a lock can be either *a single-chamber lock* or *a multi-chamber lock*. On the other hand, they may be *unidirectional*, allowing the transfer of ships in one way, or *bidirectional*, allowing ships to pass in both ways. We summarize in Table 7.1 the most well-known methods discussed in this chapter with their reference projects.

| | Unidirectional | Bidirectional |
|---------------------|----------------|---------------|
| Single chamber lock | [324] [325] | [322] [326] |
| Multi-chamber lock | [327] | [328] [143] |

Table 7.1: Overview of problem variants discussed in this chapter.

Nauss [326] considers a static lock scheduling problem where many ships are queued on both sides of a lock. They formulate nonlinear integer programs to deal with three potential objective functions: minimize total elapsed time to schedule all the ships, minimize unweighted cycle time and minimize weighted cycle time. However, this static method may be applicable for specific situations only, when a river lock has been unavailable for a considerable period, and so many ships arrive meanwhile.

Bugarski, Bačkalić, and Kuzmanov [325] and Liang et al. [322] use a Fuzzy logic-based method to minimize waiting time while minimizing the number of empty lockages. However, their solutions are restricted since they make many assumptions, including a constant

average time for the lockage duration and a fixed capacity of the locks, allowing it to hold only a single vessel.

A single-chamber unidirectional lock scheduling problem is investigated in [324]. The objective is to minimize the operation costs and other costs, e.g., water cost, by selecting an appropriate slot number during a planned period. To solve this problem, Wang et al. use an Ant Colony Optimization-based data-driven method. Minimizing the lock's water usage is often modeled as the total number of lockages required to transfer all ships. While applying this objective results in water-efficient scheduling, the resulting waiting time for the ships can be high, especially when the inter-arrival time between ships is long. However, the main disadvantage to (mixed) integer programming models is the rapid increase of the required computation time as the instance increases in size.

Passchyn et al. [327] consider a predefined arrival time of the ships and a constant lockage time to minimize the total waiting time of the ships. They prove that the lock scheduling problem with multiple chambers is NP-hard even after removing the restrictions of two-dimensional ship placement.

Ji et al. [143] develop a multi-order-best-fit (MOBF)-based Tabu search method, which is essentially a specific heuristic to minimize the waiting time of ships and place them into chambers. This hybrid method has been numerically proven to be highly efficient by comparing it with a MIP method.

A hybrid heuristic method, where a modified binary nondominated sorting genetic algorithm is proposed in [328] to solve a coordinated scheduling problem at the Three Gorges Dam lock.

In conclusion, previous works on lock scheduling consider deterministic frameworks that do not resemble reality. They either act after the ships arrive or admit that their arrival times are known. This leads to additional waiting time or high congestion at the locks. Moreover, most researchers suppose a constant lockage duration and a fixed dimension of the chambers. Thus, the present study considers bidirectional and multi-chamber locks with at least one but possibly multiple chambers with different characteristics. Each chamber has a specific capacity based on its dimensions and a different lockage duration. Additionally, we consider arranging the arrival time of ships by adjusting their speed. When ships have an Inquired Arrival Deadline (IAD), they may sail at unnecessarily high speeds to meet the deadline but consume much more fuel than necessary. Thus, adjusting ships' speeds can necessarily minimize fuel consumption.

7.3 Problem Formulation

A lockage consists of a group of ships entering a chamber from one side of the lock, upon which the water level inside the chamber changes and the ships can exit the chamber from the other side of the lock. Note that a lockage may also be empty, i.e., contain no ships and that such empty lockages cannot always be avoided. Indeed, after serving a ship, the water level in one of the lock's chambers changes. Hence, to let another ship waiting at the initial water level pass by the same chamber, an empty lockage is needed. The necessary steps to transfer a ship in a lock from downstream to upstream are: i) the ship enters the lock;

ii) the back door is closed; iii) the lock is filled with water until the water level rises to the upstream; iv) the front door is opened; v) the ship exits the lock. Inland locks have at least one chamber but may consist of multiple consecutive chambers of different dimensions. Each chamber has a limited capacity and lockage duration, as shown in Fig 7.2.



Figure 7.2: Different types of locks

Lock scheduling is non-preemptive since once a ship enters a chamber, the process must be completed before allowing another ship to enter. We can, therefore, consider scheduling a lock as an Unrelated Parallel Machines Scheduling Problem (UPMSP) [329] by considering the lockage as the task and the lock chambers as the machines. The UPMSP consists of scheduling a set of jobs without preemption, each available at time zero, on unrelated machines, regarding some objectives. The processing time depends on the machine where the job is assigned. In the next section, we formulate our proposed solution and give an overview of how it works.

7.3.1 Proposed Solution

In this study, we propose to schedule the ships before their arrivals to the lock by a lock-ship real-time and dynamic cooperation, ensured by the C-IAShips architecture. Figure 7.3 depicts an overview of the system model of the proposed Lock-DS.

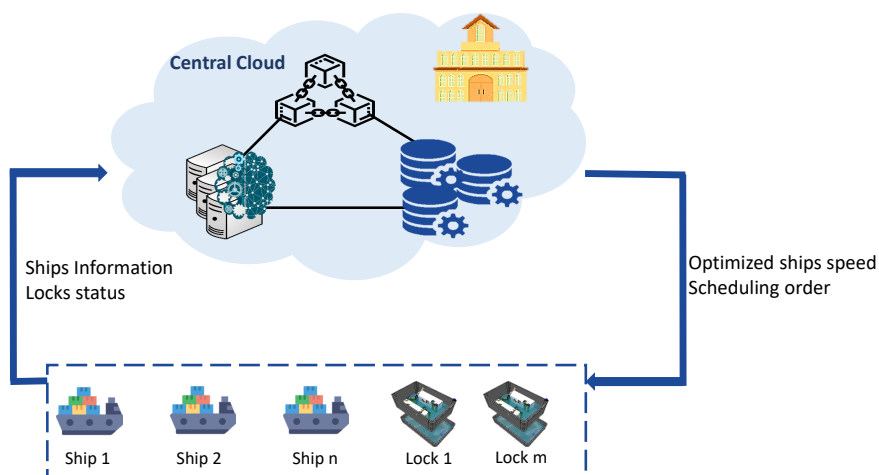


Figure 7.3: Lock-DS system model

More specifically, Lock-DS works as follow:

1. Cooperative ships send their characteristics, including their weight, dimensions, order of priority, the direction of travel, final destination, speed, and position to the cloud node.
2. The central cloud estimates their arrival time and lockage duration.
3. The central cloud arranges the ships to minimize the total waiting time and maximize the chamber area utilization while keeping an eye on the state of the locks, such as availability and waiting time.
4. The central cloud gives recommendations of optimized ships speed. The scheduling order impacts ships' decision-making on departure time and speed choices.

More details about the implementation are provided in the next sections.

7.3.2 Problem Parameters and Assumptions

The decision variables, parameters and assumptions are detailed in this section. We consider a set of ships passing a lock composed of C chambers. The symbols used in this section are detailed in Table 7.2.

| Notation | Meaning |
|-------------------|---|
| V | set of ships |
| C | set of different chambers in the lock |
| $t_{transition}$ | transition time required between two consecutive lockages performed on the same chamber both upstream (or downstream) |
| l_i, wd_i | length and width of ship i (meters) |
| L_0, Wd_0 | length and width of the smallest chamber of the lock (meters). |
| d_{safe} | safe distance to ensure at the lock (meters) |
| $start_{t_i}$ | starting time of a ship i (hour) |
| $estimated_{t_i}$ | estimated arrival time of a ship i (hour) |
| $waiting_{t_i}$ | waiting time of a ship $i = st_i - et_i$ (minutes) |
| end_{t_i} | completion time of a ship i (minutes) |
| T | end of the scheduling process (minutes) |
| fc_i | fuel consumption of ship i (tonnes per day) |
| s_i | speed of ship i (kilometer per hour) |
| a_i | weight of the ship i if empty (tonnes) |
| w_i | payload of the ship i (tonnes) |
| ld_i | lockage duration of the ship i (minutes) |

Table 7.2: Lock-DS Parameters

In this work, we ignore the influence of gravity on the movement of water through the locks. The effect of weather conditions is neglected, and we assume that the channel depth meets the requirement for ships to pass through the channel safely.

7.4 Dynamic Lock Scheduling (Lock-DS) Description

The Lock-DS process is illustrated in Figure 7.4. Details of the Lock-DS algorithm stages are given in this section.

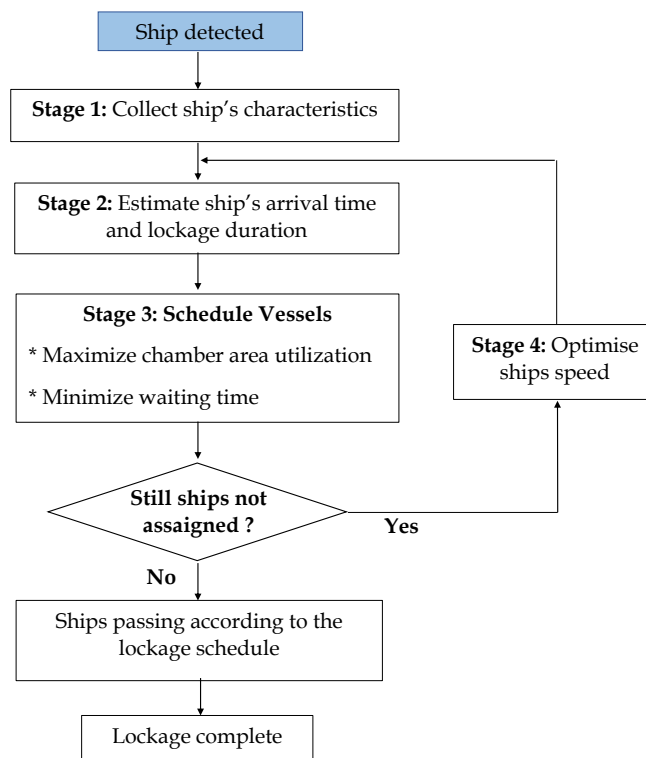


Figure 7.4: Lock-DS process

7.4.1 Lock-DS - Stage 1: Collect Ship's Characteristics

In the first step, a central entity collects the ships' characteristics including their identification, weight, dimensions, order of priority, direction of travel, final destination, speed and position. This central entity is located in the cloud and collect information about a specific geographical zone with a certain number of locks that it manages. The central cloud is managed by a central authority in each country, such as Voie Navigable de France (VNF) [153] in France, which is the navigation authority responsible for managing the french inland waterways networks, as explained in Chapter 5. Thanks to the advanced communication and locating methods, the ships' information can be collected hours before their arrival, increasing the time available to get a good schedule order later.

7.4.2 Lock-DS - Stage 2: Estimate Ship's arrival time and lockage duration

Based on the data received from the ships, we calculate for each one its estimated arrival time assuming that its speed remains constant until reaching the lock:

$$\text{estimated arrival time} = \frac{\text{actual ship's position}}{\text{actual ship's speed}} \quad (7.1)$$

The second information to calculate is the lockage duration which depends on:

- the weight of the ship (w_i) to be transferred; heavy ships require more time to fill the lock,
- the type and the capacity of the lock; the lock's capacity defines the number of ships that can pass together,
- the initial state of the chamber; the chamber's initial state defines its first available time and its state at that time. If a ship is coming from upstream and the lock is on downstream, an empty lockage may be required.

Taking into account all these parameters, the lockage duration can be calculated using the following formula [330]:

$$\left\{ \begin{array}{l} \text{lockage duration} = t_{\text{transition}} \times trans_i \\ \quad + 0.0038 \times L_0 \frac{(3 \times G + 1)}{1 - G/2} \\ \quad + 6.23 \times \exp\left(-\left(\frac{G/2 - 0.55}{0.124}\right)^2\right), \\ \text{where :} \\ G = \frac{N \times \bar{L}}{L_0} \\ \bar{L} = \frac{\sum_i (a_i + w_i)}{N} \\ N \text{ is the number of ships grouped together} \end{array} \right. \quad (7.2)$$

$$trans_i = \begin{cases} 0 & \text{if the level of water in the lock} \\ & \text{is the same as the coming ship} \\ 1 & \text{otherwise} \end{cases}$$

7.4.3 Lock-DS - Stage 3: Schedule Ships

This section explains the objectives of our problem (reduce waiting time and maximize chamber area utilization) and its constrained functions.

7.4.3.1 Objective 1: Maximize Chamber Area Utilization

each chamber has a limited surface area, which must not be exceeded by the total area of ships present in the chamber at any given time.

When several ships are assigned to the same chamber in the lockage process, the chamber capacity requirement should be met. We aim to find ships coming from the same direction:

$$\text{maximize } \sum_i v_i * \left(\frac{l_i}{L_0} + \frac{wd_i}{Wd_0} \right), \quad (7.3)$$

respect to:

$$\begin{cases} \sum_i v_i * l_i + d_{safe} \leq L_0 \\ \sum_i v_i * wd_i + d_{safe} \leq Wd_0 \end{cases}$$

where

$$v_i = \begin{cases} 1 & \text{ship } i \text{ is chosen to pass the lock} \\ 0 & \text{otherwise.} \end{cases}$$

These constraints indicate that the ships scheduled to pass together should keep a safe distance between them, and their size should be less than the size of the lock's chamber. In particular, the lock scheduling is non-preemptive, and the same group of ships must pass through all the chambers; hence, we should consider the dimensions of the smallest chamber of the lock. In other words, the size of ships in any given lockage should not exceed the smallest chamber capacity. We assume that the lockage process starts immediately when all the ships in the same group have reached the lock, i.e. we do not consider an extra waiting time to enter the locks.

7.4.3.2 Objective 2: Minimize the Waiting Time

We define the waiting time as the time spent by that ship waiting before entering a lock chamber. Note that this excludes the time spent inside lock chambers. The waiting time can thus be computed as the difference between the estimated arrival time of the ship and the lockage start time. We aim to minimize the waiting time of all the ships:

$$\text{minimize } \sum_i (start_t_i - estimated_t_i), \quad (7.4)$$

respect to:

- $estimated_t_i \leq start_t_i \leq T, \forall i \in V \implies$ The starting time of a lockage should be within its estimated arrival time and the end of scheduling period,

- $start_t_i \leq end_t_j \forall i > j \in V \implies$ the ship i can only start after the former one completes.

7.4.3.3 Problem Resolution

To solve the problem described above and find the best sequence of ships in a deterministic way, we need to try all the possible permutations to conclude the best order that minimizes the total waiting time and maximizes the chamber area utilization. We can consider scheduling a lock as a machine's scheduling problem by considering the lockage as the task and the lock chambers as the machines. [331] Garey, Johnson, and Stockmeyer prove that determining a minimum mean-flow-time schedule in an m -machine flow shop is NP-complete for every $m \geq 2$.

Flow shop problems assume that all tasks are identical so that all operations run in a uniform order through the machines. Our sequencing problem is more complicated since each task (lockage) has specific operations (lockage duration). Given the inherent complexity, no exact algorithm has been found to solve these problems in a reasonable amount of time. Therefore, the challenge is to find a balance between solution quality and the model's speed when creating a lock scheduling model. The traditional solution is to design efficiently by approximation or heuristic algorithms with performance guarantees under certain assumptions. Although these model-oriented algorithms can achieve good results, they are not adapted to the dynamic environment where the ships arriving rate and density are unknown in advance. As a model-free reinforcement learning (RL) method [332], Q-learning can be applied to address this dynamic scheduling problem with a low delay response. RL is a type of machine learning concerned with how agents should take actions in an environment to maximize future rewards. The RL main idea is to let agents learn by trial and error throughout the actions' execution. So, the algorithm learns about the search space and directs the search to find good-quality solutions. One of the strengths of this algorithm is that it can compare the expected utility of the available actions without requiring a model of the environment. Several authors use RL to solve scheduling problems [332, 333, 334]. The ability to cope with environmental uncertainty in a dynamic environment, the ability to self-learn the environment, the computational efficiency, and the high adaptivity are all grounds for reinforcement learning to become a promising technology in dynamic scheduling. We choose the Q-learning algorithm, in particular, because it has many qualities relevant to our problem [335, 336]: (i) it can find a variety of distinct solutions for the same issue thanks to its combination of exploitative and explorative policies (i.e., picking the best-known action vs. creating a new random one), (ii) the Q-table is highly reusable and easy to import and export into new executions, and (iii) it takes into account the effects of implementing an action to measure its suitability. The purpose is to find a suitable sequence of jobs that optimize a set of constraints required by the real-world processes. Conclusions show that the RL scheduler can find near-optimal solutions across different instances and optimization objectives with high stability and low computation cost. Our study's formalization of the reinforcement learning problem is inspired by Markov's Decision Process (MDP), a practical approach to model the sequential decision-making problem to achieve a long-term objective.

7.4.3.4 MDP Formulation & Model Design

We consider a lock with m chambers, each having L_0 as length and Wd_0 as width and a number of ships, with for each ship v : l_v as length and wd_v as width and having each a ld_v lockage duration. The scheduler picks ships and assigns them to chambers. Over time, scheduled ships are processed, and the scheduler assigns new ships as long as there are unassigned ships remaining or new ships arrive.

Formally, the basic MDP model consists of a triplet $\{S, A, R\}$ defined as:

- a set of environment states S ;
- a set of actions A ;
- a set of rewards R .

At each time t , the agent perceives its state s_t in S and the set of possible actions $A(s_t)$. It chooses an action a in $A(s_t)$ and moves to a new state s_{t+1} and receives from the environment a reward r_{t+1} (see Figure 7.5). This means that the agent implements a mapping from states to probabilities of selecting each possible action. This mapping is called the agent's policy and is denoted π_t , where $\pi_t(s, a)$ is the probability that $a_t = a$ if $s_t = s$, in other words, it is the probability of selecting action a in state s at time t . A RL agent's sole objective is to maximize the total reward it receives in the long run.

- ❖ **State Representation:** The state set S contains all possible states, and a state change is only possible when a new assignment is made. We define the state set as the ships scheduling processes that have been completed and the order in which they were passed, or more precisely, ships' precedence relations. We represent the state of the system as binary matrices. In Figure 7.5, we draw a simplified illustration of our state's representation.

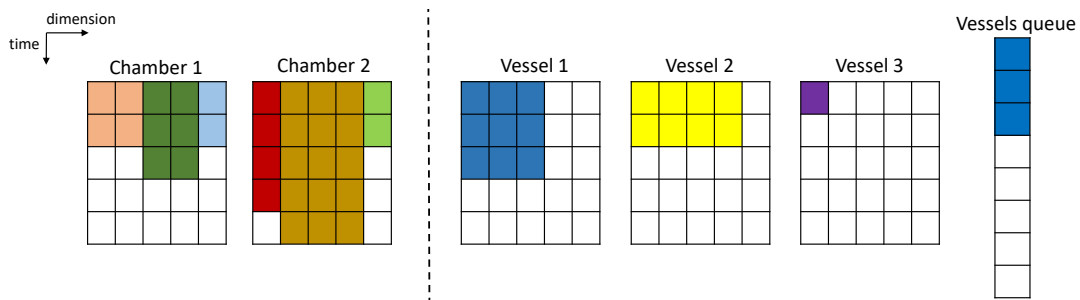


Figure 7.5: Illustration of the state representation with two chambers and 3 waiting ships.

One color represents one ship. The left matrices show the status of scheduled ships for different chambers. In contrast, the ship matrices show the profiles of the queuing ships. For example, the green dots in the first matrix indicate that a ship is scheduled for chamber 1, it has $2/5$ of chamber dimension, and its lockage lasts three time-units.

- ❖ **Action Space:** The action space \mathbf{A} of the task scheduling includes two actions: one is to determine the selection order among \mathbf{V} ships, and the other is to assign one of the \mathbf{C} chambers to each ship. Thus, the agent's action is equivalent to deciding which ship will be listed next in the sequence already constructed from the set of ships still waiting.

We define an action \mathbf{a} as assigning the ship \mathbf{v} to the chamber \mathbf{c} . The scheduler takes multiple actions until all the chambers are full or it takes an invalid action (attempting to schedule a ship in an inappropriate chamber). With each valid action, one ship is assigned to one chamber and the system state changes.

- ❖ **Rewards:** In our case, rewards are designed so that the system learns to minimize the waiting time and maximize the chamber area utilization. Therefore, we relate it with the inverse of total waiting time and chamber occupation percentage. As mentioned above, the objective is to minimize the waiting time (Equation 7.4) and maximize the lock's chamber area utilization (Equation. 7.3). The reward is designed to guide the scheduler toward the goal of the optimal policy $\pi = \mathbf{p}(\mathbf{a}|\mathbf{s})$. For a valid action, the reward is the sum of the total area utilization and the inverse of the total waiting time, and we give zero rewards if the invalid action is selected; thus, the reward function is designed as:

$$\text{reward} = \begin{cases} \text{Total Area Utilization}(Eq.7.3) + \frac{1}{\text{Total Waiting Time (Eq. 7.4)}} & \mathbf{a} \in A^{valide} \\ 0 & \mathbf{a} \in A^{invalide} \end{cases}$$

- ❖ **Q-Learning implementation:** The general architecture of reinforcement learning is presented in Figure 7.6, where the RL agent selects an action according to the Q-table and executes it. The environment moves to a new state and returns a reward to the agent [337].

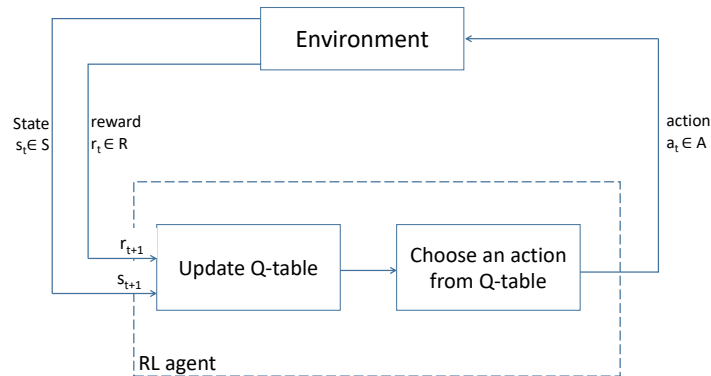


Figure 7.6: RL general architecture

Initially, the RL agent does not know which action will maximize its reward. It has to discover which actions are most profitable by applying them. So it performs random actions from a state and observes the incurred reward using the feedback of that action from the environment. Knowledge acquired is stored in the Q-Table. This table stores pairs

of states and actions together with a Q-value. The Q-value indicates how good each pair is. We calculate the Q-values for particular state action pairs based on these feedbacks, with repeated calculations using the Bellman Equation (Equation 7.5.)

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)), \text{ where:} \quad (7.5)$$

Where :

- α is the learning rate: The learning rate $0 < \alpha < 1$ determines what fraction of the old estimate will be updated with the new estimate. $\alpha = 0$ will stop the RL agent from learning anything while $\alpha = 1$ will completely change the previous values with the new one. Here, we use $\alpha = 0.1$.
- γ is the discount factor: The discount factor $0 < \gamma < 1$ determines the importance of future rewards. For $\gamma = 0$ all the upcoming rewards are ignored. For $\gamma = 1$, the RL agent will consider the current and upcoming rewards equal weightage. We use $\gamma = 0.6$.
- For action selection, we use the ϵ -greedy policy: it avoids local optima and maintains the balance between exploration (i.e., choosing a random action) with probability ϵ and exploiting (i.e., choosing the action with highest Q-value) the remainder of the time. We use an ϵ of 0.3.

Details of Q-Learning is provided in Algorithm 3, adapted from [335].

Algorithm 3 Q-Learning pseudo-code

```
Initialize Q-Table
for each training session do
   $s_t \leftarrow s_0$ 
  for each training session step do
    Choose  $a_t$  for  $s_t$  using the  $\epsilon$ -greedy policy
    Take action  $a_t$ , observe state  $s_{t+1}$  and  $r_{t+1}$ 
    Update Q-value, with the Equation 7.5
   $s \leftarrow s_{t+1}$ 
end for
end for
```

7.4.4 Lock-DS - Stage 4: Optimize Ships' Speed

Speed optimization can offer notable gains in terms of waiting time and fuel consumption. On the one hand, because of the non-linear relationship between speed and fuel consumption, it is evident that a ship that goes slower will emit much less than the same ship going faster. On the other hand, instead of reaching the lock and waiting for their turn, ships

can lower their speeds, for instance, so they arrive at their scheduled starting time. In that sense, the impact of a change in ship speed can be significant.

In this study, we recommend that ships adjust their speed in different ways:

- For ships grouped to pass the lock together, we recommend the last ones accelerate, while considering the maximum authorized speed, to reach the lock at approximately the same arrival time,
- For last scheduled ships, we recommend they decelerate to minimize their waiting time and thus their fuel consumption.

7.5 Performance Evaluation

This section presents the experiments carried out to validate the proposed solution. We start by generating problem instances and then, we perform multiple numerical experiments.

7.5.1 Performance Evaluation Metrics

To evaluate the performance of the scheduling process, we use two evaluation metrics. The first metric concerns the Total waiting time of all the ships, which is the sum of the difference between the lockage start time and the estimated arrival time for all the ship. The second metric concerns their Fuel consumption.

Most maritime literature considers that fuel consumption has a cubic relationship with sailing speed. However, it is proved that fuel consumption is influenced by many other factors, such as the ship's payload. Therefore, the formula $fc_i = \alpha s_i^3$ is not accurate, especially for inland ships because they sail at a low speed. A realistic closed-form approximation that takes both speed and payload into account is formulated by [338] :

$$fc_i = k(p + s_i^q)(a_i + w_i)^{2/3}, \quad (7.6)$$

Where $k > 0$ is a known ship specific constant related to its characteristics. $p \geq 0$ is a constant used to ensure obtaining a realistic fuel consumption when the ship speed is low. $q \geq 3$ is a constant related to ships' speed for generic use a value of $q = 3$ could be used. [338] suggests using $q \geq 4$ when ship's speed is greater than 20 knots (37 km/h). Since, the maximum authorized speed is 10 to 12 km/h on French canals [153], we will use $q = 3$ in this study.

7.5.2 Numerical Experiments

The problem instances are randomly generated within fixed ranges according to [153]: the lock's length is between 40.50 m and 120 m, and its width is between 6 m and 30 m. The ships' weight is between 3000 and 7000 tonnes, their speed is between 5 and 12km/h, their width is between 5m to 11.40m, their length is between 38m and 95m, and their arrival

times are randomly generated according to a Poisson distribution. Algorithm performance is tested on a single lock with a different number of ships in both directions. We compare in Table 7.3 the deterministic, the original navigation that uses First Come First Serve (FCFS), the Shortest Job First (SJF), and our optimized navigation scheduling, Lock-DS. In the deterministic scheduling, we generate all the possible permutations and conclude the best order to minimize the waiting time. In the original navigation scheduling, we implement the FCFS technique. The third scheduling technique is the SJF, where we first pass the ships with a minimum lockage duration. In addition, we compare our solution to [328] and [143] solutions.

| Number of ships | Algorithm | Deterministic | FCFS | SJF | [328] | [143] | Lock-DS |
|-----------------|-----------|---------------|--------------|--------------|---------------|---------------|----------------|
| | | waiting time | waiting time | waiting time | waiting time | waiting time | waiting time |
| 3 | | 3 | 14.85 | 3 | 3 | 3 | 3 |
| 4 | | 42 | 130.49 | 42 | 42 | 42 | 42 |
| 5 | | 24.65 | 163.79 | 183.79 | 24.65 | 24.65 | 24.65 |
| 6 | | 194.64 | 354.64 | 434.64 | 194.64 | 194.64 | 194.64 |
| 7 | | 174.53 | 474.13 | 494.13 | 434.13 | 434.13 | 174.53 |
| 10 | | - | 894.53 | 1154.53 | 915.4 | 987.46 | 646.91 |
| 15 | | - | 1709.45 | 1669.45 | 537.69 | 537.69 | 537.69 |
| 20 | | - | 3190.82 | 3190.82 | 2583.01 | 2320.93 | 1833.31 |
| 25 | | - | 6297.48 | 6237.48 | 3963.17 | 3670.64 | 3190.64 |
| 30 | | - | 6524.44 | 7784.44 | 3752.69 | 2189.7 | 2390.51 |
| 40 | | - | 14930.64 | 13410.64 | 9209.24 | 7266.37 | 6726.37 |
| 50 | | - | 10701.91 | 9041.91 | 7477.29 | 7991.76 | 6371.59 |

Table 7.3: Schedulers waiting time comparison

The numerical experiments show that our optimized algorithm produces a near-optimal solution. It is observed that traditional non-preemptive techniques SJF and FCFS are simple, fast, and deterministic. However, they are not efficient in understanding the optimality problem. FCFS does not prioritize ships, which means when there are ships with a long lockage duration, in the beginning, all small ships must wait a lengthy time. SJF does not always provide better solutions. For small instances, [328] and [143] usually obtain the optimal solution; however, for large instances, they fail to find it. Additionally, since we adjust the ships' speed with our solution, we always obtain the best values.

Figure 7.7 gives a comparison of the fuel consumption between the FCFS, SJF, [328] and [143] schedulers and the proposed optimized navigation scheduling. As shown, the fuel consumption is reduced for all ships in the optimized navigation scheduling. The total consumption of the ships can be reduced by 48.03 % on average compared to the current used method.

7.6 Conclusion

With the rapid growth of freight volume in inland waterways, there is an urgent need to manage the traffic efficiently. In particular, crossing the locks is one of the most important goals of traffic management. This chapter proposes an efficient cooperative scheduling algorithm, Lock-DS, based on reinforcement learning to answer which order is optimal for

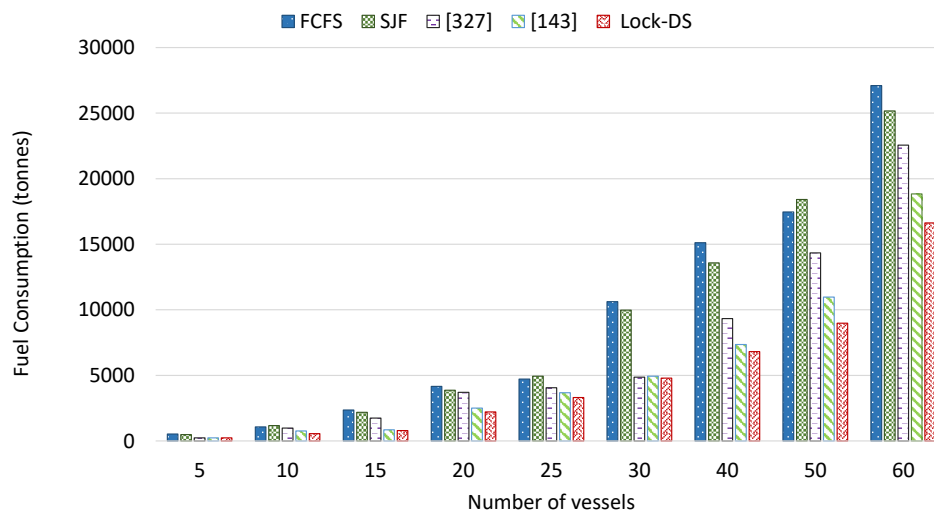


Figure 7.7: The fuel consumption (in tonnes) under five different schedulers

a given set of coming ships. The optimal order may be the one which: (1) minimizes the total waiting time, (2) maximizes the lock occupation, (3) reduces the total fuel consumption. We thus study the problem of optimal order selection from the three different angles mentioned above in a dynamic way. To ameliorate the performance of our scheduler, we adjust the ships' speed according to the need. Simulation results show that the proposed approach could be beneficial for both ships and locks. For the ships, they can spend less time waiting at the locks and leave the lock earlier with the optimized ship arrivals. Additionally, with optimal speeds, they can achieve lower fuel consumption. For the locks, fewer lockage operations and less time are required to handle the same number of ships with the optimized lockage plans. Numerical results prove that the Lock-DS reduces fuel consumption by 48.03 % on average compared to the current scheduling method used at locks, i.e., the FCFS.

In future work, we propose to extend this research in several directions. One direct extension is to manage the ship's positioning in the lock chamber. The ship positioning problem is a variant of the well-known two-dimensional rectangular Single Bin Size Bin Packing Problem (2D rectangular SBSBPP). A set of rectangular items (ships) has to be positioned inside as few rectangular bins (lock chambers) as possible, without allowing the rotation of these items. In this study, we consider simplified constraints to ensure. However, in future work, we can tackle this positioning problem. Another extension considers the ships' speed adjustment with a more sophisticated solution such as the solution proposed by Buchem, Golak, and Grigoriev [339]. The paper shows how to find a close-to-optimal velocity policy by introducing a mathematical optimization problem modeling the uncertainty at locks in inland waterways when considering the operation of a single ship. Additionally, we can consider other objectives for the lock scheduling problem, including the minimization of the lock's water usage.

To sum up, in this part, we studied the cooperation among the river entities, i.e. ships and locks. With the proposed C-IAShips architecture, we ensure low latency and efficient communication while protecting the privacy of the ships and the security of the exchanged data. Additionally, it enables a more powerful and efficient operation of the overall system. In particular, we have studied the feasibility of two cooperative applications: the first for collision detection and the second for scheduling the passage of ships at locks.

CONCLUSION

| | | |
|-----|--|-----|
| 8.1 | Reminder of Goals and Challenges | 160 |
| 8.2 | Summary of the Contributions | 160 |
| 8.3 | Future Research Directions | 162 |

8.1 Reminder of Goals and Challenges

Inland network is offering potential growth for river freight transportation in many world countries. The inland waterway transport mode may be a great alternative to other modes since it is environmentally friendly, reliable, secure, and less costly, especially when transporting vast bulk cargo volumes. Europe has over 30,000 kilometers of rivers that link together many crucial industrial concentration areas, which are already providing services to freight businesses, and a lesser extent, passengers and boaters. Recently, the European Union and the French government, in particular, have decided to progress more in this sector by doubling the share of inland waterways transport in the next few years. Thus, developing this sector is becoming urgent.

In this context, this thesis proposes to adopt the smart river concept, relying on cooperative and autonomous navigation, in order to improve inland transportation and accessibility, reduce inconvenience, and increase social and economic profits. Basically, we aim to introduce Cooperative Intelligent Transport Systems (C-ITS) techniques to the inland sector. However, deploying such a concept reveals some challenges, including information sensing, communication and navigation, traffic planning, status monitoring, fault diagnosis, distress warning and rescue, and autonomous navigation. Hence, to make the smart river become a reality, we discern three vital sub-problems:

- ❖ How to deal with the restricted maneuverability and lack of flexibility in inland infrastructure?
- ❖ How to develop a reliable autonomous ship regarding the fluvial environment-specific characteristics?
- ❖ How to manage the increasing volume of inland ships with reliable fleet management?

8.2 Summary of the Contributions

The manuscript began with a deep survey on cooperative and autonomous inland navigation, where we discussed its specific features, applications, requirements, and challenges. We realize that it is still in its early stages, although it will have great potentials in the future. Hence, throughout this thesis, we focused on designing a reliable cooperative and autonomous inland navigation while considering the aforementioned three sub-problems. We briefly conclude our contributions in what follows:

- ❖ **Providing a Lock Automation Decision Making (Lock-ADM)** Inland vessels often have to cross numerous locks before reaching their final destination, which leads to a significant delay and sometimes represents as much as half of the total travel time. The delay affects shipment costs and can affect other parts of the transport chain, adversely impacting this transportation mode's growth. Our first contribution focuses on making infrastructural modifications by proposing an efficient Lock Automation Decision Making (Lock-ADM) method. The problem modeling consists of

using a three-stage algorithm. Firstly, we calculate the optimal number of locks while minimizing the investment costs using the exact solver, CPLEX. Secondly, we measure the importance of locks in the network, and finally, we select the best locks to automate using the Genetic Algorithm (GA) metaheuristic. We achieved an average reduction of 33.7 % in overall lock waiting time at a low cost based on real data.

- ❖ **Designing a Reliable Awareness System for Autonomous River Ships** For autonomous ships, an intelligent awareness system is a fundamental task that provides crucial information on the sailing environment. Therefore, our second contribution addressed new challenges related to fluvial visual perception for autonomous navigation to propose a system that accurately delimits safe navigation areas by simultaneously locating and mapping the environment. First, we constructed the first open-source dataset; the InlandAutoDetect dataset that comprises 3,377 images comprehensively labeled for object detection in a fluvial domain with almost 30,000 objects annotated. Second, we analyzed and compared the results of nine deep learning perception models adapted to the inland environment in accuracy and run-time speed. The best one, namely Retinanet, was selected to be integrated into the complete system to delimit the safe sailing area. The performance evaluations showed the proposed system's robustness and effectiveness to give accurate results and fulfill real-time operation requirements.
- ❖ **Implementing a Functional Architecture for Cooperative Inland Fleet Management** The Cooperative Inland Autonomous Ships (CIAShips) architecture leverages the MEC concept, blockchain technology, and federated deep learning for the effective cooperative deployment of autonomous inland-based ships. The architecture can achieve high communication efficiency and protect the privacy of ships from being leaked while realizing the different proposed applications.
- ❖ **Improving Traffic Safety with the Cooperative Collision Detection System** The inland shipping industry is growing continuously, and its traffic is becoming denser in many navigable waterways. Therefore, we designed a new cooperative collision detection system based on the CIAShips architecture. It continuously gathers localization data from ships and processes them at the MEC level to predict potential collisions. Then, alerts will be sent to ships to avoid collisions between them. Numerical results showed that our system achieves a high accuracy collision detection rate while maintaining a trade-off between the low latency and accuracy due to the MEC technology and the federated deep learning technique. These results demonstrated its effectiveness for fluvial navigation.
- ❖ **Ensuring a Dynamic Lock Scheduling (Lock-DS)** Our last contribution introduced a Dynamic Lock Scheduling (Lock-DS) to efficiently manage vessels scheduling at locks by minimizing their waiting time and optimizing their speed. We achieved an average reduction of 69.9 % in vessel waiting time and a reduction of 48.03 % in total fuel consumption compared to existing scheduling methods.

To sum up, the smart river concept proposed in this thesis focus on improving inland navigation. Our contributions revolve around developing cooperative and autonomous

navigation. To automate the navigation, we proposed two solutions. The first contribution, Lock-ADM, aims to automate the inland infrastructure, and the second contribution aims to automate the inland ships. On the other hand, to ensure reliable cooperation between river components, we first propose the CIAShips architecture to deal with the cooperation challenges and requirements. Then, we focus on improving traffic management with two cooperative applications: improving traffic safety with the cooperative collision detection system and managing ships crossing through locks with the Lock-DS. Our work is open to possible extensions to enlarge the treated challenges and face new emerging ones. Therefore, we enumerate the possible enhancements, which is the aim of our future directions described in the next section.

8.3 Future Research Directions

Regardless of the presented contributions provided during this thesis to enhance cooperative and autonomous navigation, some aspects can be additionally explored and extended. Therefore, we devote this section to identify and study some perspectives and possible future research directions.

- ❖ **Improving the Perception Part** The fluvial environment is complex, which makes autonomous shipping in such an environment a complicated task. In this thesis, we developed an autonomous reliable perception system based on visual analysis only. Nevertheless, using a single camera only is inefficient for guiding an autonomous vessel. Thus, the integration of other sources of information should be considered. In general, for an autonomous vehicle, both lidar sensors and cameras play essential roles in an autonomous stack. Cameras bring high resolution to the table, where lidar sensors bring depth information. That said, in situations where one of the two sensors may experience degraded performance, such as in the rain, the other sensor can play a role in picking up the slack for a perception system. When it comes to the utilization of radar in autonomous vehicles, LiDAR and camera both benefit from this antecedent technology: the operation of a camera sensor can be impaired by snow, rain, or fog. Such weather conditions also change the refractive index of the transmission medium and reduce the range of a LiDAR sensor. Resistance to weather conditions is one reason why radar is also incorporated in the design of most automotive sensor suites.
- ❖ **Implementation on Real Ships** Embedding our cooperative approach into real inland ships systems would offer other technical challenges. For instance, to evaluate our dynamic lock scheduling algorithm, we used the simulated arrival flow of ships. In the real environment, some external factors may influence their arrivals, such as the weather conditions and the seasonal characters. Thus, real-world deployment of our Lock-DS in a non-controlled environment would definitely be a plus.
- ❖ **Environmental Considerations** Future works would also be concerned with integrating new methods and disciplines, including those related to environmental protection enhancement. For instance, we can minimize the water used at locks as an additional objective of our lock scheduling solution. In addition, we can measure and thus reduce the ecological footprint [340, 341] at the port using installed sensors.

REFERENCES

- [1] Zoran Radmilović and Branislav Dragović. “The inland navigation in europe: basic facts, advantages and disadvantages”. In: *Journal of maritime research* 4 (2007), pp. 31–46.
- [2] European Commission 2020. *MOBILITY AND TRANSPORT*. Accessed: 2021-09-19. URL: https://ec.europa.eu/transport/modes/inland_en.
- [3] French ministry for ecological transition. *Généralités sur le transport et le réseau fluvial en France*. Accessed: 2021-08-01. URL: <https://www.ecologie.gouv.fr/generalites-sur-transport-et-reseau-fluvial-en-france>.
- [4] French waterways team. *French Waterways in Detail-Dunkirk port and canals*. Accessed: 2021-09-19. URL: <https://www.french-waterways.com/waterways/north/dunkerque-canal/>.
- [5] Omprakash Kaiwartya et al. “Internet of Vehicles: Motivation, Layered Architecture, Network Model, Challenges, and Future Aspects”. In: *IEEE Access* 4 (2016), pp. 5356–5373. DOI: [10.1109/ACCESS.2016.2603219](https://doi.org/10.1109/ACCESS.2016.2603219).
- [6] Sheraz Aslam, Michalis P. Michaelides, and Herodotos Herodotou. “Internet of Ships: A Survey on Architectures, Emerging Applications, and Challenges”. In: *IEEE Internet of Things Journal* 7.10 (2020), pp. 9714–9727. DOI: [10.1109/JIOT.2020.2993411](https://doi.org/10.1109/JIOT.2020.2993411).
- [7] Aleksandra Ivanova et al. “Autonomous Shipping Means: the Main Areas of Patenting Research and Development Results”. In: *Transportation Research Procedia* 54 (2021). International Scientific Siberian Transport Forum - TransSiberia 2020, pp. 793–801. ISSN: 2352-1465. DOI: <https://doi.org/10.1016/j.trpro.2021.02.132>. URL: <https://www.sciencedirect.com/science/article/pii/S2352146521003082>.
- [8] Adrienne LaFrance. *Our grandmother’s driverless car*. Accessed: 2021-09-14. URL: <https://www.theatlantic.com/technology/archive/2016/06/beep-beep/489029/>.
- [9] Amit Kumar Tyagi and S U Aswathy. “Autonomous Intelligent Vehicles (AIV): Research statements, open issues, challenges and road for future”. In: *International Journal of Intelligent Networks* 2 (2021), pp. 83–102. ISSN: 2666-6030. DOI: <https://doi.org/10.1016/j.ijin.2021.07.002>. URL: <https://www.sciencedirect.com/science/article/pii/S2666603021000142>.
- [10] Omprakash Kaiwartya et al. “Internet of Vehicles: Motivation, Layered Architecture, Network Model, Challenges, and Future Aspects”. In: *IEEE Access* 4 (2016), pp. 5356–5373. DOI: [10.1109/ACCESS.2016.2603219](https://doi.org/10.1109/ACCESS.2016.2603219).

- [11] S.O.R.A.V.S. Committee. “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles”. In: 2014, 01–16.
- [12] The Paris Urbanism Agency. “Impacts and potential benefits of autonomous vehicles : from an international context to Grand Paris”. In: *Atelier parisien d’urbanisme* (octobre 2018), pp. 4–48.
- [13] Roman Zakharenko. “Self-driving cars will change cities”. In: *Regional Science and Urban Economics* 61 (2016), pp. 26–37. ISSN: 0166-0462. DOI: [10.1016/j.regsciurbeco.2016.09.003](https://doi.org/10.1016/j.regsciurbeco.2016.09.003).
- [14] Shengbo Eben Li et al. “Kalman filter-based tracking of moving objects using linear ultrasonic sensor array for road vehicles”. In: *Mechanical Systems and Signal Processing* 98 (2018), pp. 173–189. ISSN: 0888-3270. DOI: <https://doi.org/10.1016/j.ymsp.2017.04.041>. URL: <https://www.sciencedirect.com/science/article/pii/S0888327017302364>.
- [15] Robert Bosch GmbH. *Bosch MRR Data Sheet*. Accessed: 2021-10-01. URL: <https://www.bosch-mobility-solutions.com>.
- [16] Rita Spinneker et al. “Fast fog detection for camera based Advanced Driver Assistance Systems”. In: *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. 2014, pp. 1369–1374. DOI: [10.1109/ITSC.2014.6957878](https://doi.org/10.1109/ITSC.2014.6957878).
- [17] N. Bulusu, J. Heidemann, and D. Estrin. “GPS-less low-cost outdoor localization for very small devices”. In: *IEEE Personal Communications* 7.5 (2000), pp. 28–34. DOI: [10.1109/98.878533](https://doi.org/10.1109/98.878533).
- [18] Julius Ziegler et al. “Making Bertha Drive—An Autonomous Journey on a Historic Route”. In: *IEEE Intelligent Transportation Systems Magazine* 6.2 (2014), pp. 8–20. DOI: [10.1109/MITS.2014.2306552](https://doi.org/10.1109/MITS.2014.2306552).
- [19] Alberto Broggi et al. “Extensive Tests of Autonomous Driving Technologies”. In: *IEEE Transactions on Intelligent Transportation Systems* 14.3 (2013), pp. 1403–1415. DOI: [10.1109/TITS.2013.2262331](https://doi.org/10.1109/TITS.2013.2262331).
- [20] Joydeep Dey, Wes Taylor, and Sudeep Pasricha. “VESPA: A Framework for Optimizing Heterogeneous Sensor Placement and Orientation for Autonomous Vehicles”. In: *IEEE Consumer Electronics Magazine* 10.2 (2021), pp. 16–26. DOI: [10.1109/MCE.2020.3002489](https://doi.org/10.1109/MCE.2020.3002489).
- [21] P.J. Besl and Neil D. McKay. “A method for registration of 3-D shapes”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (1992), pp. 239–256. DOI: [10.1109/34.121791](https://doi.org/10.1109/34.121791).
- [22] Jacopo Serafin and Giorgio Grisetti. “NICP: Dense normal based point cloud registration”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 742–749. DOI: [10.1109/IROS.2015.7353455](https://doi.org/10.1109/IROS.2015.7353455).
- [23] “Voxelized GICP for Fast and Accurate 3D Point Cloud Registration”. In:

- [24] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. 2005, 886–893 vol. 1. DOI: [10.1109/CVPR.2005.177](https://doi.org/10.1109/CVPR.2005.177).
- [25] Yu-Ho Tseng and Shau-Shiun Jan. “Combination of computer vision detection and segmentation for autonomous driving”. In: *2018 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. 2018, pp. 1047–1052. DOI: [10.1109/PLANS.2018.8373485](https://doi.org/10.1109/PLANS.2018.8373485).
- [26] Atmane Khellal, Hongbin Ma, and Qing Fei. “Convolutional Neural Network Based on Extreme Learning Machine for Maritime Ships Recognition in Infrared Images”. In: *Sensors* 18.5 (2018). ISSN: 1424-8220. DOI: [10.3390/s18051490](https://doi.org/10.3390/s18051490). URL: <https://www.mdpi.com/1424-8220/18/5/1490>.
- [27] Andreas Geiger, Philip Lenz, and Raquel Urtasun. “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.
- [28] Marius Cordts et al. “The Cityscapes Dataset for Semantic Urban Scene Understanding”. In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.
- [29] Hunjae Yoo, Ukil Yang, and Kwanghoon Sohn. “Gradient-Enhancing Conversion for Illumination-Robust Lane Detection”. In: *IEEE Transactions on Intelligent Transportation Systems* 14.3 (2013), pp. 1083–1094. DOI: [10.1109/TITS.2013.2252427](https://doi.org/10.1109/TITS.2013.2252427).
- [30] Gwangsoo Park, Byungjin Lee, and Sangkyung Sung. “Integrated Pose Estimation Using 2D Lidar and INS Based on Hybrid Scan Matching”. In: *Sensors* 21.16 (2021). ISSN: 1424-8220. DOI: [10.3390/s21165670](https://doi.org/10.3390/s21165670). URL: <https://www.mdpi.com/1424-8220/21/16/5670>.
- [31] Xin Yang and Kwang-Ting Tim Cheng. “Local Difference Binary for Ultrafast and Distinctive Feature Description”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.1 (2014), pp. 188–194. DOI: [10.1109/TPAMI.2013.150](https://doi.org/10.1109/TPAMI.2013.150).
- [32] Cesar Cadena et al. “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age”. In: *IEEE Transactions on Robotics* 32.6 (2016), pp. 1309–1332. DOI: [10.1109/TRO.2016.2624754](https://doi.org/10.1109/TRO.2016.2624754).
- [33] Raúl Mur-Artal and Juan D. Tardós. “ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras”. In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262. DOI: [10.1109/TRO.2017.2705103](https://doi.org/10.1109/TRO.2017.2705103).
- [34] Keisuke Tateno et al. “CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

- [35] Niko Sünderhauf et al. “Place Recognition With ConvNet Landmarks: Viewpoint-Robust, Condition-Robust, Training-Free”. In: July 2015. DOI: [10.15607/RSS.2015.XI.022](https://doi.org/10.15607/RSS.2015.XI.022).
- [36] Reza Moghdani et al. “The green vehicle routing problem: A systematic literature review”. In: *Journal of Cleaner Production* 279 (2021), p. 123691. ISSN: 0959-6526. DOI: <https://doi.org/10.1016/j.jclepro.2020.123691>. URL: <https://www.sciencedirect.com/science/article/pii/S0959652620337367>.
- [37] Vincent A. Laurence and J. Christian Gerdes. “Long-Horizon Vehicle Motion Planning and Control Through Serially Cascaded Model Complexity”. In: *IEEE Transactions on Control Systems Technology* (2021), pp. 1–14. DOI: [10.1109/TCST.2021.3056315](https://doi.org/10.1109/TCST.2021.3056315).
- [38] Paolo Toth and Daniele Vigo. *Vehicle Routing*. Ed. by Daniele Vigo and Paolo Toth. Society for Industrial and Applied Mathematics, 2014. DOI: [10.1137/1.9781611973594](https://doi.org/10.1137/1.9781611973594). eprint: <https://epubs.siam.org/doi/pdf/10.1137/1.9781611973594>. URL: <https://epubs.siam.org/doi/abs/10.1137/1.9781611973594>.
- [39] Ulrich Derigs, Markus Pullmann, and Ulrich Vogel. “Truck and trailer routing—Problems, heuristics and computational experience”. In: *Computers & Operations Research* 40.2 (2013), pp. 536–546. ISSN: 0305-0548. DOI: <https://doi.org/10.1016/j.cor.2012.08.007>. URL: <https://www.sciencedirect.com/science/article/pii/S0305054812001724>.
- [40] C.K.Y. Lin. “A vehicle routing problem with pickup and delivery time windows, and coordination of transportable resources”. In: *Computers & Operations Research* 38.11 (2011), pp. 1596–1609. ISSN: 0305-0548. DOI: <https://doi.org/10.1016/j.cor.2011.01.021>. URL: <https://www.sciencedirect.com/science/article/pii/S0305054811000335>.
- [41] Yuxiang Zhang et al. “A Novel Trajectory Planning Method for Automated Vehicles Under Parameter Decision Framework”. In: *IEEE Access* 7 (2019), pp. 88264–88274. DOI: [10.1109/ACCESS.2019.2925417](https://doi.org/10.1109/ACCESS.2019.2925417).
- [42] Peng Liu and Ümit Özgüner. “Predictive control of a vehicle convoy considering lane change behavior of the preceding vehicle”. In: *2015 American Control Conference (ACC)*. 2015, pp. 4374–4379. DOI: [10.1109/ACC.2015.7172017](https://doi.org/10.1109/ACC.2015.7172017).
- [43] A. Chebly, R. Talj, and A. Charara. “Coupled Longitudinal and Lateral Control for an Autonomous Vehicle Dynamics Modeled Using a Robotics Formalism”. In: *IFAC-PapersOnLine* 50.1 (2017). 20th IFAC World Congress, pp. 12526–12532. ISSN: 2405-8963. DOI: <https://doi.org/10.1016/j.ifacol.2017.08.2190>. URL: <https://www.sciencedirect.com/science/article/pii/S2405896317328604>.

- [44] Rachid ATTIA, Rodolfo ORJUELA, and Michel BASSET. “Longitudinal Control for Automated Vehicle Guidance”. In: *IFAC Proceedings Volumes* 45.30 (2012). 3rd IFAC Workshop on Engine and Powertrain Control, Simulation and Modeling, pp. 65–71. ISSN: 1474-6670. DOI: <https://doi.org/10.3182/20121023-3-FR-4025.00049>. URL: <https://www.sciencedirect.com/science/article/pii/S1474667015351405>.
- [45] Carlos Massera Filho et al. “Longitudinal and lateral control for autonomous ground vehicles”. In: *2014 IEEE Intelligent Vehicles Symposium Proceedings*. 2014, pp. 588–593. DOI: [10.1109/IVS.2014.6856431](https://doi.org/10.1109/IVS.2014.6856431).
- [46] Fernando auat cheein et al. “SLAM-based Cross-a-Door Solution Approach for a Robotic Wheelchair”. In: *International Journal of Advanced Robotic Systems* 6 (Sept. 2009). DOI: [10.5772/7230](https://doi.org/10.5772/7230).
- [47] R. Lenain et al. “Model Predictive Control for Vehicle Guidance in Presence of Sliding: Application to Farm Vehicles Path Tracking”. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. 2005, pp. 885–890. DOI: [10.1109/ROBOT.2005.1570229](https://doi.org/10.1109/ROBOT.2005.1570229).
- [48] Zhijun Li, Weidong Chen, and Hong Liu. “Robust Control of Wheeled Mobile Manipulators Using Hybrid Joints”. In: *International Journal of Advanced Robotic Systems* 5.1 (2008), p. 9. DOI: [10.5772/5656](https://doi.org/10.5772/5656). eprint: <https://doi.org/10.5772/5656>. URL: <https://doi.org/10.5772/5656>.
- [49] T. Hessburg and M. Tomizuka. “Fuzzy logic control for lateral vehicle guidance”. In: *IEEE Control Systems Magazine* 14.4 (1994), pp. 55–63. DOI: [10.1109/37.295971](https://doi.org/10.1109/37.295971).
- [50] Jose Calvo-Rolle et al. “A hybrid intelligent system for PID controller using in a steel rolling process”. In: *Expert Systems with Applications* 40 (Oct. 2013), 5188–5196. DOI: [10.1016/j.eswa.2013.03.013](https://doi.org/10.1016/j.eswa.2013.03.013).
- [51] S. Baluja. “Evolution of an artificial neural network based autonomous land vehicle controller”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 26.3 (1996), pp. 450–463. DOI: [10.1109/3477.499795](https://doi.org/10.1109/3477.499795).
- [52] Dong Li et al. “Reinforcement Learning and Deep Learning Based Lateral Control for Autonomous Driving [Application Notes]”. In: *IEEE Comput. Intell. Mag.* 14 (2019), pp. 83–98.
- [53] Hans-Christoph Burmeister et al. “Autonomous Unmanned Merchant Vessel and its Contribution towards the e-Navigation Implementation: The MUNIN Perspective”. In: *International Journal of e-Navigation and Maritime Economy* 1 (2014), pp. 1–13. ISSN: 2405-5352. DOI: <https://doi.org/10.1016/j.enavi.2014.12.002>. URL: <https://www.sciencedirect.com/science/article/pii/S2405535214000035>.

- [54] Fatma Outay, Hanan Abdullah Mengash, and Muhammad Adnan. “Applications of unmanned aerial vehicle (UAV) in road safety, traffic and highway infrastructure management: Recent advances and challenges”. In: *Transportation Research Part A: Policy and Practice* 141 (2020), pp. 116–129. ISSN: 0965-8564. DOI: <https://doi.org/10.1016/j.tra.2020.09.018>. URL: <https://www.sciencedirect.com/science/article/pii/S096585642030728X>.
- [55] S. Sudhakar et al. “Unmanned Aerial Vehicle (UAV) based Forest Fire Detection and monitoring for reducing false alarms in forest-fires”. In: *Computer Communications* 149 (2020), pp. 1–16. ISSN: 0140-3664. DOI: <https://doi.org/10.1016/j.comcom.2019.10.007>. URL: <https://www.sciencedirect.com/science/article/pii/S0140366419308655>.
- [56] Ahmed Alioua et al. “UAVs for traffic monitoring: A sequential game-based computation offloading/sharing approach”. In: *Computer Networks* 177 (2020), p. 107273. ISSN: 1389-1286. DOI: <https://doi.org/10.1016/j.comnet.2020.107273>. URL: <https://www.sciencedirect.com/science/article/pii/S1389128619315798>.
- [57] Razieh Nadafianshamabadi, Mohammad Tayarani, and Gregory Rowangould. “A closer look at urban development under the emergence of autonomous vehicles: Traffic, land use and air quality impacts”. In: *Journal of Transport Geography* 94 (2021), p. 103113. ISSN: 0966-6923. DOI: <https://doi.org/10.1016/j.jtrangeo.2021.103113>. URL: <https://www.sciencedirect.com/science/article/pii/S0966692321001666>.
- [58] MBARI. *Autonomous underwater vehicles*. Accessed: 2021-10-10. URL: <https://www.mbari.org/at-sea/vehicles/autonomous-underwater-vehicles/>.
- [59] Zorana Milosevic et al. “Guidance for Autonomous Underwater Vehicles in Confined Semistructured Environments”. In: *Sensors* 20.24 (2020). ISSN: 1424-8220. DOI: [10.3390/s20247237](https://doi.org/10.3390/s20247237). URL: <https://www.mdpi.com/1424-8220/20/24/7237>.
- [60] Chia-Hsun Chang et al. “Risk assessment of the operations of maritime autonomous surface ships”. In: *Reliability Engineering & System Safety* 207 (2021), p. 107324. ISSN: 0951-8320. DOI: <https://doi.org/10.1016/j.res.2020.107324>. URL: <https://www.sciencedirect.com/science/article/pii/S0951832020308176>.
- [61] M.A. Ramos et al. “Human-system concurrent task analysis for maritime autonomous surface ship operation and safety”. In: *Reliability Engineering & System Safety* 195 (2020), p. 106697. ISSN: 0951-8320. DOI: <https://doi.org/10.1016/j.res.2019.106697>. URL: <https://www.sciencedirect.com/science/article/pii/S0951832018313085>.

- [62] Jacoby Larson et al. “Advances in Autonomous Obstacle Avoidance for Unmanned Surface Vehicles”. In: (Jan. 2007).
- [63] Ørnulf Rødseth and Håvard Nordahl. “Definition of autonomy levels for merchant ships, Report from NFAS, Norwegian Forum for Autonomous Ships, 2017-08-04.” In: (Aug. 2018). DOI: [10.13140/RG.2.2.21069.08163](https://doi.org/10.13140/RG.2.2.21069.08163).
- [64] Namireddy Praveen Reddy et al. “Zero-Emission Autonomous Ferries for Urban Water Transport: Cheaper, Cleaner Alternative to Bridges and Manned Vessels”. In: *IEEE Electrification Magazine* 7.4 (2019), pp. 32–45. DOI: [10.1109/MELE.2019.2943954](https://doi.org/10.1109/MELE.2019.2943954).
- [65] AMS research team. *Transforming Amsterdam’s canals with a fleet of autonomous boats*. Accessed: 2021-10-10. URL: <https://roboat.org/#roboat>.
- [66] Gerben Peeters et al. “An Inland Shore Control Centre for Monitoring or Controlling Unmanned Inland Cargo Vessels”. In: *Journal of Marine Science and Engineering* 8.10 (2020). ISSN: 2077-1312. DOI: [10.3390/jmse8100758](https://doi.org/10.3390/jmse8100758). URL: <https://www.mdpi.com/2077-1312/8/10/758>.
- [67] Mingyu Kim et al. “Autonomous shipping and its impact on regulations, technologies, and industries”. In: *Journal of International Maritime Safety, Environmental Affairs, and Shipping* 4.2 (2020), pp. 17–25. DOI: [10.1080/25725084.2020.1779427](https://doi.org/10.1080/25725084.2020.1779427). URL: <https://doi.org/10.1080/25725084.2020.1779427>.
- [68] NTNU. *SFI AutoShip*. Accessed: 2021-10-10. URL: <https://www.ntnu.edu/sfi-autoship>.
- [69] Mina Sorial et al. “Towards a Real Time Obstacle Detection System for Unmanned Surface Vehicles”. In: *OCEANS 2019 MTS/IEEE SEATTLE*. 2019, pp. 1–8. DOI: [10.23919/OCEANS40490.2019.8962685](https://doi.org/10.23919/OCEANS40490.2019.8962685).
- [70] Wa Nzengu et al. “Regulatory framework analysis for the unmanned inland waterway vessel”. In: *WMU Journal of Maritime Affairs* 20 (Sept. 2021), pp. 357–375. DOI: [10.1007/s13437-021-00237-z](https://doi.org/10.1007/s13437-021-00237-z).
- [71] Igor Bačkalov. “Safety of autonomous inland vessels: An analysis of regulatory barriers in the present technical standards in Europe”. In: *Safety Science* 128 (2020), p. 104763. ISSN: 0925-7535. DOI: <https://doi.org/10.1016/j.ssci.2020.104763>. URL: <https://www.sciencedirect.com/science/article/pii/S0925753520301600>.
- [72] Wei Wang et al. “Design, Modeling, and Nonlinear Model Predictive Tracking Control of a Novel Autonomous Surface Vehicle”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 6189–6196. DOI: [10.1109/ICRA.2018.8460632](https://doi.org/10.1109/ICRA.2018.8460632).

- [73] International Maritime Organization. *SOLAS: International Convention for the Safety of Life at Sea, 1974 : 1997/1998 amendments*. English. International Maritime Organization London, 1999, v, 18 p. : ISBN: 9280160931.
- [74] International Maritime Organization. *COLREG: Convention on the International Regulations for Preventing Collisions at Sea, 1972*. International Maritime Organization, 2003. DOI: [92801416789789280141672](https://doi.org/10.1016/j.oceaneng.2020.107056). URL: <http://worldcat.org>.
- [75] Hull to Hull. *Using EGNOS and Galileo to support Autonomous Maritime Operations*. Accessed: 2021-09-14. URL: <https://www.sintef.no/projectweb/hull-to-hull/>.
- [76] Gerben Peeters et al. “An unmanned inland cargo vessel: Design, build, and experiments”. In: *Ocean Engineering* 201 (2020), p. 107056. ISSN: 0029-8018. DOI: <https://doi.org/10.1016/j.oceaneng.2020.107056>. URL: <https://www.sciencedirect.com/science/article/pii/S0029801820301293>.
- [77] Watertruck+. *The future of inland navigation*. Accessed: 2021-10-10. URL: <https://watertruckplus.eu/>.
- [78] Natalia Wawrzyniak and Andrzej Stateczny. “Automatic Watercraft Recognition and Identification on Water Areas Covered by Video Monitoring as Extension for Sea and River Traffic Supervision Systems”. In: *Polish Maritime Research* 25.s1 (2018), pp. 5–13. DOI: [doi:10.2478/pomr-2018-0016](https://doi.org/10.2478/pomr-2018-0016). URL: <https://doi.org/10.2478/pomr-2018-0016>.
- [79] Michael D. Henschel et al. “Comparison of probability statistics for automated ship detection in SAR imagery”. In: *Other Conferences*. 1998.
- [80] Xue Yang et al. “Automatic Ship Detection in Remote Sensing Images from Google Earth of Complex Scenes Based on Multiscale Rotation Dense Feature Pyramid Networks”. In: *Remote Sensing* 10.1 (2018), p. 132. ISSN: 2072-4292. DOI: [10.3390/rs10010132](https://doi.org/10.3390/rs10010132). URL: <http://dx.doi.org/10.3390/rs10010132>.
- [81] Marta Włodarczyk-Sielicka and Dawid Polap. “Automatic Classification Using Machine Learning for Non-Conventional Vessels on Inland Waters”. In: *Sensors* 19.14 (2019). ISSN: 1424-8220. DOI: [10.3390/s19143051](https://doi.org/10.3390/s19143051). URL: <https://www.mdpi.com/1424-8220/19/14/3051>.
- [82] Fábio Duarte, Lenna Johnsen, and Carlo Ratti. “Reimagining urban infrastructure through design and experimentation”. In: Mar. 2020, pp. 395–410. ISBN: 9781315178387. DOI: [10.4324/9781315178387-27](https://doi.org/10.4324/9781315178387-27).
- [83] Thanh-Hai Tran and Thi Le. “Vision based boat detection for maritime surveillance”. In: Jan. 2016. DOI: [10.1109/ELINFOCOM.2016.7563033](https://doi.org/10.1109/ELINFOCOM.2016.7563033).
- [84] Yuan Yao et al. “Ship detection in optical remote sensing images based on deep convolutional neural networks”. In: *Journal of Applied Remote Sensing* 11.4 (2017), pp. 1–12. DOI: [10.1117/1.JRS.11.042611](https://doi.org/10.1117/1.JRS.11.042611). URL: <https://doi.org/10.1117/1.JRS.11.042611>.

- [85] Hongwei Zhao et al. “Embedded Deep Learning for Ship Detection and Recognition”. In: *Future Internet* 11.2 (2019). ISSN: 1999-5903. DOI: [10.3390/fi11020053](https://doi.org/10.3390/fi11020053). URL: <https://www.mdpi.com/1999-5903/11/2/53>.
- [86] Yohei Matsumoto. “Ship Image Recognition using HOG”. In: *The Journal of Japan Institute of Navigation* 129 (Jan. 2013), pp. 105–112. DOI: [10.9749/jin.129.105](https://doi.org/10.9749/jin.129.105).
- [87] Zhijun Chen et al. “Deep learning for autonomous ship-oriented small ship detection”. In: *Safety Science* 130 (2020), p. 104812. ISSN: 0925-7535. DOI: <https://doi.org/10.1016/j.ssci.2020.104812>. URL: <https://www.sciencedirect.com/science/article/pii/S0925753520302095>.
- [88] Ian J. Goodfellow et al. “Generative Adversarial Nets”. In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. NIPS’14. Montreal, Canada: MIT Press, 2014, 2672–2680.
- [89] František Duchoň et al. “Path Planning with Modified a Star Algorithm for a Mobile Robot”. In: *Procedia Engineering* 96 (2014). Modelling of Mechanical and Mechatronic Systems, pp. 59–69. ISSN: 1877-7058. DOI: <https://doi.org/10.1016/j.proeng.2014.12.098>. URL: <https://www.sciencedirect.com/science/article/pii/S187770581403149X>.
- [90] Yogang Singh et al. “Towards use of Dijkstra Algorithm for Optimal Navigation of an Unmanned Surface Vehicle in a Real-Time Marine Environment with results from Artificial Potential Field”. In: *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation* 12.1 (2018), pp. 125–131. ISSN: 2083-6473. DOI: [10.12716/1001.12.01.14](https://doi.org/10.12716/1001.12.01.14). URL: [./Article_Towards_use_of_Dijkstra_Algorithm_for_Optimal_Navigation_of_an_Unmanned_Surface_Vehicle_in_a_Real-Time_Marine_Environment_with_results_from_Artificial_Potential_Field_Singh,45,795.html](https://www.transnav.com/Article_Towards_use_of_Dijkstra_Algorithm_for_Optimal_Navigation_of_an_Unmanned_Surface_Vehicle_in_a_Real-Time_Marine_Environment_with_results_from_Artificial_Potential_Field_Singh,45,795.html).
- [91] Heesu Kim et al. “A study on path optimization method of an unmanned surface vehicle under environmental loads using genetic algorithm”. In: *Ocean Engineering* 142 (2017), pp. 616–624. ISSN: 0029-8018. DOI: <https://doi.org/10.1016/j.oceaneng.2017.07.040>. URL: <https://www.sciencedirect.com/science/article/pii/S0029801817304122>.
- [92] Jia Song, Ce Hao, and Su Jiangcheng. “Path planning for unmanned surface vehicle based on predictive artificial potential field”. In: *International Journal of Advanced Robotic Systems* 17 (Mar. 2020), p. 172988142091846. DOI: [10.1177/1729881420918461](https://doi.org/10.1177/1729881420918461).
- [93] Chang Song. “Global Path Planning Method for USV System Based on Improved Ant Colony Algorithm”. In: *Applied Mechanics and Materials* 568-570 (June 2014), pp. 785–788. DOI: [10.4028/www.scientific.net/AMM.568-570.785](https://doi.org/10.4028/www.scientific.net/AMM.568-570.785).

- [94] Wang Yuan-hui and Chi Cen. “Research on optimal planning method of USV for complex obstacles”. In: *2016 IEEE International Conference on Mechatronics and Automation*. 2016, pp. 2507–2511. DOI: [10.1109/ICMA.2016.7558960](https://doi.org/10.1109/ICMA.2016.7558960).
- [95] Yongquan Zhou and Rui Wang. “An Improved Flower Pollination Algorithm for Optimal Unmanned Undersea Vehicle Path Planning Problem”. In: *International Journal of Pattern Recognition and Artificial Intelligence* 30.04 (2016), p. 1659010. DOI: [10.1142/S0218001416590102](https://doi.org/10.1142/S0218001416590102). eprint: <https://doi.org/10.1142/S0218001416590102>. URL: <https://doi.org/10.1142/S0218001416590102>.
- [96] J. Zhuang et al. “Improved rapidly-exploring random tree algorithm application in unmanned surface vehicle local path planning”. In: *Harbin Gongye Daxue Xuebao/Journal of Harbin Institute of Technology* 47 (Jan. 2015), pp. 112–117. DOI: [10.11918/j.issn.0367-6234.2015.01.017](https://doi.org/10.11918/j.issn.0367-6234.2015.01.017).
- [97] Dongdong Mu et al. “Course control of USV based on fuzzy adaptive guide control”. In: *2016 Chinese Control and Decision Conference (CCDC)*. 2016, pp. 6433–6437. DOI: [10.1109/CCDC.2016.7532156](https://doi.org/10.1109/CCDC.2016.7532156).
- [98] Sang-Min Lee, Kyung-Yub Kwon, and Joongseon Joh. “A fuzzy logic for autonomous navigation of marine vehicle satisfying COLREG guidelines”. In: *International Journal of Control, Automation, and Systems* 2 (June 2004).
- [99] Roy Glasius, Andrzej Komoda, and Stan C.A.M. Gielen. “Neural Network Dynamics for Path Planning and Obstacle Avoidance”. In: *Neural Networks* 8.1 (1995), pp. 125–133. ISSN: 0893-6080. DOI: [https://doi.org/10.1016/0893-6080\(94\)E0045-M](https://doi.org/10.1016/0893-6080(94)E0045-M). URL: <https://www.sciencedirect.com/science/article/pii/0893608094E0045M>.
- [100] G. Wang et al. “Obstacle identification method based on laser radar for inland unmanned vessel”. In: *Guangxue Jishu/Optical Technique* 44 (Sept. 2018), pp. 602–608.
- [101] Kai Zheng et al. “A SVM based ship collision risk assessment algorithm”. In: *Ocean Engineering* 202 (2020), p. 107062. ISSN: 0029-8018. DOI: <https://doi.org/10.1016/j.oceaneng.2020.107062>. URL: <https://www.sciencedirect.com/science/article/pii/S0029801820301359>.
- [102] Josh Redding et al. “Collaborative Mission Planning, Autonomy and Control Technology (CoMPACT) for Unmanned Surface Vehicles”. In: Aug. 2009. ISBN: 978-1-60086-978-5. DOI: [10.2514/6.2009-5774](https://doi.org/10.2514/6.2009-5774).
- [103] Signe Moe and Kristin Y. Pettersen. “Set-Based line-of-sight (LOS) path following with collision avoidance for underactuated unmanned surface vessels under the influence of ocean currents”. In: *2017 IEEE Conference on Control Technology and Applications (CCTA)*. 2017, pp. 241–248. DOI: [10.1109/CCTA.2017.8062470](https://doi.org/10.1109/CCTA.2017.8062470).
- [104] P. A. Wilson, C. J. Harris, and X. Hong. “A Line of Sight Counteraction Navigation Algorithm for Ship Encounter Collision Avoidance”. In: *Journal of Navigation* 56.1 (2003), 111–121. DOI: [10.1017/S0373463302002163](https://doi.org/10.1017/S0373463302002163).

- [105] S. Campbell, W. Naeem, and G.W. Irwin. “A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres”. In: *Annual Reviews in Control* 36.2 (2012), pp. 267–283. ISSN: 1367-5788. DOI: <https://doi.org/10.1016/j.arcontrol.2012.09.008>. URL: <https://www.sciencedirect.com/science/article/pii/S1367578812000430>.
- [106] T Xu, Rhondasutton Sutton, and Sanjay Sharma. “A multi-sensor data fusion navigation system for an unmanned surface vehicle”. In: *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment* 221 (Dec. 2007), pp. 167–182. DOI: [10.1243/14750902JEME72](https://doi.org/10.1243/14750902JEME72).
- [107] Jonggu Kang et al. “A study on application of sensor fusion to collision avoidance system for ships”. In: *ICCAS 2010*. 2010, pp. 1741–1744. DOI: [10.1109/ICCAS.2010.5669788](https://doi.org/10.1109/ICCAS.2010.5669788).
- [108] Center for Advanced Automotive Technology. *Connected and Automated Vehicles*. Accessed: 2021-09-18. URL: http://autocaat.org/Technologies/Automated_and_Connected_Vehicles/.
- [109] Toyota. *Connected Services*. Accessed: 2021-10-08. URL: <https://www.toyota.com/connected-services/>.
- [110] Furqan Jameel, Zheng Chang, and Jun Huang. *Internet of Autonomous Vehicles: Architecture, Features, and Socio-Technological Challenges*. May 2019.
- [111] Klaus Bengler et al. “Three Decades of Driver Assistance Systems: Review and Future Perspectives”. In: *IEEE Intelligent Transportation Systems Magazine* 6.4 (2014), pp. 6–22. DOI: [10.1109/MITS.2014.2336271](https://doi.org/10.1109/MITS.2014.2336271).
- [112] DSRC Technical Committee. *On-Board System Requirements for V2V Safety Communications*. 2016. DOI: https://doi.org/10.4271/J2945/1_201603. URL: https://doi.org/10.4271/J2945/1_201603.
- [113] Technical Committee : ISO/TS 19091 Intelligent transport systems. *Intelligent transport systems — Cooperative ITS — Using V2I and I2V communications for applications related to signalized intersections*. 2019, p. 233.
- [114] Kristian Nybom et al. “IoT at Sea”. Odefinierat/okänt. In: *2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*. null ; Conference date: 06-06-2018 Through 08-06-2018. IEEE, 2018, 1–7. ISBN: 978-1-5386-4730-1. DOI: [10.1109/BMSB.2018.8436741](https://doi.org/10.1109/BMSB.2018.8436741).
- [115] Yongjae Kim, Yujae Song, and Sung Hoon Lim. “Hierarchical Maritime Radio Networks for Internet of Maritime Things”. In: *IEEE Access* PP (Apr. 2019), pp. 1–1. DOI: [10.1109/ACCESS.2019.2911703](https://doi.org/10.1109/ACCESS.2019.2911703).
- [116] 3GPP TR 22.819 V16.2.0 (2018-12). *Technical Report. Feasibility Study on Maritime Communication Services over 3GPP System*. Accessed: 2021-09-16. URL: http://www.3gpp.org/ftp/Specs/archive/22_series/22.819/22819-g20.zip.

- [117] Zhe Tian et al. “The Development of Key Technologies in Applications of Vessels Connected to the Internet”. In: *Symmetry* 9.10 (2017). ISSN: 2073-8994. DOI: [10.3390/sym9100211](https://doi.org/10.3390/sym9100211). URL: <https://www.mdpi.com/2073-8994/9/10/211>.
- [118] Sheraz Aslam, Michalis Michaelides, and Herodotos Herodotou. “Internet of Ships: A Survey on Architectures, Emerging Applications, and Challenges”. In: *IEEE Internet of Things Journal* (May 2020). DOI: [10.1109/JIOT.2020.2993411](https://doi.org/10.1109/JIOT.2020.2993411).
- [119] Moritz von Stietencron et al. “Utilising the Internet of Things for the Management of Through-life Engineering Services on Marine Auxiliaries”. In: *Procedia CIRP* 59 (2017). Proceedings of the 5th International Conference in Through-life Engineering Services Cranfield University, 1st and 2nd November 2016, pp. 233–239. ISSN: 2212-8271. DOI: <https://doi.org/10.1016/j.procir.2016.09.003>. URL: <https://www.sciencedirect.com/science/article/pii/S2212827116309362>.
- [120] Powell Gregory et al. “Vehicle-to-vehicle communications : readiness of V2V technology for application”. In: *United States. National Highway Traffic Safety Administration*. Rapport technique, Washington, DC : National Highway Traffic Safety Administration, 2014, p. 27999.
- [121] H. M. Taher. “Sea Ad hoc Network (SANET) Challenges and Development”. In: 2018.
- [122] Maritime VSAT Company. *Maritime VSAT Communications*. Accessed: 2021-09-16. URL: <https://marlink.com/solutions/maritime-vsats/>.
- [123] DanelecConnect company. *Danelec value your Maritime data*. Accessed: 2021-09-16. URL: <https://www.danelec-marine.com/danelecconnect/>.
- [124] Hongzhi Zhou, Gan Yu, and Linguo Li. “Cloud Communication based Ship Communication Network Security Risk Assessment Model”. In: *Journal of Coastal Research* 95.sp1 (2020), pp. 991 –995. DOI: [10.2112/SI95-193.1](https://doi.org/10.2112/SI95-193.1). URL: <https://doi.org/10.2112/SI95-193.1>.
- [125] Xubin Yang et al. “A Heterogeneous Ship Formation Network Selection Algorithm Based on Service Level and Load Balance”. In: *Proceedings of the 2015 International Conference on Communications, Signal Processing, and Systems*. Ed. by Qilian Liang et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 13–23. ISBN: 978-3-662-49831-6.
- [126] Junyi Yu et al. “AIS and VTS Information Fusion in the Internet of Inland Ships”. In: *ICTIS 2013*, pp. 2409–2416. DOI: [10.1061/9780784413036.325](https://doi.org/10.1061/9780784413036.325). eprint: <https://ascelibrary.org/doi/pdf/10.1061/9780784413036.325>. URL: <https://ascelibrary.org/doi/abs/10.1061/9780784413036.325>.

- [127] Yuan Zhuang and Shaoqiao Song. “Use of Internet of Things for Ship Management of Inland Rivers”. In: *ICTIS 2013*, pp. 2425–2431. DOI: [10.1061/9780784413036.327](https://doi.org/10.1061/9780784413036.327). eprint: <https://ascelibrary.org/doi/pdf/10.1061/9780784413036.327>. URL: <https://ascelibrary.org/doi/abs/10.1061/9780784413036.327>.
- [128] Francisco de la Vega et al. “Implementation of a Fiware-based Integration Platform and a Web Portal as Aids to Improve the Control of Ships Navigation in a River”. In: *2020 IEEE International Symposium on Systems Engineering (ISSE)*. 2020, pp. 1–3. DOI: [10.1109/ISSE49799.2020.9272242](https://doi.org/10.1109/ISSE49799.2020.9272242).
- [129] Huafeng Wu et al. “Novel Design of Inland Shipping Management Information System Based on WSN and Internet-of-things”. In: June 2011, pp. 199–205. ISBN: 978-0-415-69120-8. DOI: [10.1201/b11347-36](https://doi.org/10.1201/b11347-36).
- [130] Kevin Tierney, Stefan Voß, and Robert Stahlbock. “A mathematical model of inter-terminal transportation”. In: *European Journal of Operational Research* 235.2 (2014). Maritime Logistics, pp. 448–460. ISSN: 0377-2217. DOI: <https://doi.org/10.1016/j.ejor.2013.07.007>. URL: <https://www.sciencedirect.com/science/article/pii/S0377221713005778>.
- [131] Huarong Zheng, Rudy R. Negenborn, and Gabriël Lodewijks. “Closed-loop scheduling and control of waterborne AGVs for energy-efficient Inter Terminal Transport”. In: *Transportation Research Part E: Logistics and Transportation Review* 105 (2017), pp. 261–278. ISSN: 1366-5545. DOI: <https://doi.org/10.1016/j.tre.2016.07.010>. URL: <https://www.sciencedirect.com/science/article/pii/S1366554516300321>.
- [132] A.M. Douma. “Aligning the Operations of Barges and Terminals through Distributed Planning”. Undefined. PhD thesis. Netherlands: University of Twente, Dec. 2008. ISBN: 9789036527354. DOI: [10.3990/1.9789036527354](https://doi.org/10.3990/1.9789036527354).
- [133] Shijie Li, Rudy R. Negenborn, and Gabriel Lodewijks. “Distributed constraint optimization for addressing vessel rotation planning problems”. In: *Engineering Applications of Artificial Intelligence* 48 (2016), pp. 159–172. ISSN: 0952-1976. DOI: <https://doi.org/10.1016/j.engappai.2015.11.001>. URL: <https://www.sciencedirect.com/science/article/pii/S0952197615002481>.
- [134] Linying Chen et al. “Cooperative Multi-Vessel Systems in Urban Waterway Networks”. In: *IEEE Transactions on Intelligent Transportation Systems* 21.8 (2020), pp. 3294–3307. DOI: [10.1109/TITS.2019.2925536](https://doi.org/10.1109/TITS.2019.2925536).
- [135] Rafal Szlapczynski and Joanna Szlapczynska. “A method of determining and visualizing safe motion parameters of a ship navigating in restricted waters”. In: *Ocean Engineering* 129 (2017), pp. 363–373. ISSN: 0029-8018. DOI: <https://doi.org/10.1016/j.oceaneng.2016.11.044>. URL: <https://www.sciencedirect.com/science/article/pii/S0029801816305492>.

- [136] Mauro Candeloro, Anastasios M. Lekkas, and Asgeir J. Sørensen. “A Voronoi-diagram-based dynamic path-planning system for underactuated marine vessels”. In: *Control Engineering Practice* 61 (2017), pp. 41–54. ISSN: 0967-0661. DOI: <https://doi.org/10.1016/j.conengprac.2017.01.007>. URL: <https://www.sciencedirect.com/science/article/pii/S0967066117300072>.
- [137] Agnieszka Nowy et al. “Modeling of Vessel Traffic Flow for Waterway Design—Port of Świnoujście Case Study”. In: *Applied Sciences* 11.17 (2021). ISSN: 2076-3417. DOI: [10.3390/app11178126](https://doi.org/10.3390/app11178126). URL: <https://www.mdpi.com/2076-3417/11/17/8126>.
- [138] Longhui Gang et al. “Estimation of vessel collision risk index based on support vector machine”. In: *Advances in Mechanical Engineering* 8.11 (2016), p. 1687814016671250. DOI: [10.1177/1687814016671250](https://doi.org/10.1177/1687814016671250). URL: <https://doi.org/10.1177/1687814016671250>.
- [139] Rong Zhen, Maria Riveiro, and Yongxing Jin. “A novel analytic framework of real-time multi-vessel collision risk assessment for maritime traffic surveillance”. In: *Ocean Engineering* 145 (2017), pp. 492–501. ISSN: 0029-8018. DOI: <https://doi.org/10.1016/j.oceaneng.2017.09.015>. URL: <https://www.sciencedirect.com/science/article/pii/S0029801817305346>.
- [140] J. Needham. “Civil Engineering and Nautics”. In: *Science and Civilisation in China, Cambridge: Cambridge University Press* 4:3 (1971).
- [141] Z. Kanovic, V. Bugarski, and T. Backalic. “Ship lock control system optimization using GA, PSO and ABC: A comparative review”. In: *Promet-Traffic & Transportation* 26.1 (2014).
- [142] X. Yuan, B. Ji, and Y. Yuan. “Co-scheduling of lock and water–land transshipment for ships passing the dam”. In: *Applied Soft Computing* 45 (2016).
- [143] Bin Ji et al. “Exact and heuristic methods for optimizing lock-quay system in inland waterway”. In: *European Journal of Operational Research* 277.2 (2019), pp. 740–755. ISSN: 0377-2217. DOI: <https://doi.org/10.1016/j.ejor.2019.03.010>. URL: <https://www.sciencedirect.com/science/article/pii/S0377221719302322>.
- [144] William W. Wilson, Bruce L. Dahl, and Richard D. Taylor. “Impacts of Lock Capacity Expansion on Delay Costs for Grain Shipped on the Mississippi River”. In: *Journal of Transport Economics and Policy* 45.1 (2011), pp. 129–154. ISSN: 00225258. URL: <http://www.jstor.org/stable/25801417>.
- [145] Ahmadreza Mahmoudzadeh et al. “Waterway maintenance budget allocation in a multimodal network”. In: *Transportation Research Part E: Logistics and Transportation Review* 146 (2021), p. 102215. ISSN: 1366-5545. DOI: <https://doi.org/10.1016/j.tre.2020.102215>. URL: <https://www.sciencedirect.com/science/article/pii/S1366554520308577>.

- [146] Bin Ji, Xiaohui Yuan, and Yanbin Yuan. “A Hybrid Intelligent Approach for Co-Scheduling of Cascaded Locks With Multiple Chambers”. In: *IEEE Transactions on Cybernetics* 49.4 (2019), pp. 1236–1248. DOI: [10.1109/TCYB.2018.2799303](https://doi.org/10.1109/TCYB.2018.2799303).
- [147] Ward Passchyn, Dirk Briskorn, and Frits C.R. Spieksma. “Mathematical programming models for lock scheduling with an emission objective,” in: *European Journal of Operational Research* 248.3 (2016), pp. 802–814. DOI: [10.1016/j.ejor.2015.09.012](https://doi.org/10.1016/j.ejor.2015.09.012).
- [148] Verstichel Jannes and Vanden Berghe Greet. “Scheduling Serial Locks: A Green Wave for Waterbound Logistics”. In: *Sustainable Logistics and Supply Chains: Innovations and Integral Approaches*. Cham: Springer International Publishing, 2016, pp. 91–109. ISBN: 978-3-319-17419-8. DOI: [10.1007/978-3-319-17419-8_5](https://doi.org/10.1007/978-3-319-17419-8_5). URL: https://doi.org/10.1007/978-3-319-17419-8_5.
- [149] Elisabeth Lübbecke, Marco E. Lübbecke, and Rolf H. Möhring. “Ship Traffic Optimization for the Kiel Canal”. In: *Operations Research* 67.3 (2019), pp. 791–812. DOI: [10.1287/opre.2018.1814](https://doi.org/10.1287/opre.2018.1814). eprint: <https://doi.org/10.1287/opre.2018.1814>. URL: <https://doi.org/10.1287/opre.2018.1814>.
- [150] E. R. Petersen and A. J. Taylor. “An Optimal Scheduling System for the Welland Canal”. In: *Transportation Science* 22.3 (1988), pp. 173–185. DOI: [10.1287/trsc.22.3.173](https://doi.org/10.1287/trsc.22.3.173). eprint: <https://doi.org/10.1287/trsc.22.3.173>. URL: <https://doi.org/10.1287/trsc.22.3.173>.
- [151] L D Smith, D C Sweeney II, and J F Campbell. “Simulation of alternative approaches to relieving congestion at locks in a river transportation system”. In: *Journal of the Operational Research Society* 60.4 (2009), pp. 519–533. DOI: [10.1057/palgrave.jors.2602587](https://doi.org/10.1057/palgrave.jors.2602587). URL: <https://doi.org/10.1057/palgrave.jors.2602587>.
- [152] Masnawi Mustaffa et al. “Preliminary Study on Air Traffic Density of Peninsular Malaysia using Visual Flight Path Trajectories from Automatic Dependent Surveillance-Broadcast (ADS-B) Data”. In: *International Journal of Engineering & Technology (IJET)* 7.1 (2018), pp. 4–25. DOI: [10.14419/ijet.v7i4.25.22238](https://doi.org/10.14419/ijet.v7i4.25.22238).
- [153] VNF. *Voies navigables de France*. 2020. URL: <http://www.vnf.fr/vnf/>.
- [154] VNF. *Voies navigables de France*. 2020. URL: <http://www.vnf.fr/passereune-ecluse-r140.html>.
- [155] Xinyu Zhang et al. “Vessel transportation scheduling optimization based on channel–berth coordination,” in: *Ocean Engineering* 112 (2016), pp. 145–152. DOI: [10.1016/j.oceaneng.2015.12.011](https://doi.org/10.1016/j.oceaneng.2015.12.011).
- [156] Xingwei Pan et al. “Managing appointments with waiting time targets and random walk-ins,” in: *Omega* 95 (2020). DOI: [10.1016/j.omega.2019.04.005](https://doi.org/10.1016/j.omega.2019.04.005).

- [157] Bin Zhang and Zhongyi Zheng. “Model and Algorithm for Vessel Scheduling through a One-Way Tidal Channel”. In: *Journal of Waterway, Port, Coastal, and Ocean Engineering* 146.1 (2020). DOI: [10.1061/\(ASCE\)WW.1943-5460.0000545](https://doi.org/10.1061/(ASCE)WW.1943-5460.0000545).
- [158] Asef-Vaziri Ardavan. “Incorporation of formal safety assessment and Bayesian network in navigational risk estimation of the Yangtze River”. In: *Transportation Research Board 98th Annual Meeting* (2019). ISSN: 19-00580.
- [159] Jean-Philippe Gervais et al. “Evaluating the Logistic and Economic Impacts of Extending 600-Foot Locks on the Upper Mississippi River: A Linear Programming Approach”. In: *Journal of the Transportation Research Forum* 118 (2000), pp. 93–105. ISSN: 0278-9434. DOI: [10.1016/j.ress.2013.04.006](https://doi.org/10.1016/j.ress.2013.04.006).
- [160] Elisa S. Menendez et al. “Novel Node Importance Measures to Improve Keyword Search over RDF Graphs”. In: *Database and Expert Systems Applications*. Ed. by Sven Hartmann et al. Springer International Publishing, 2019, pp. 143–158. ISBN: 978-3-030-27618-8.
- [161] Jianxin Li et al. “Most Influential Community Search over Large Social Networks”. In: *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. 2017, pp. 871–882. DOI: [10.1109/ICDE.2017.136](https://doi.org/10.1109/ICDE.2017.136).
- [162] Jianxin Li et al. “Most Influential Community Search over Large Social Networks”. In: *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. 2017, pp. 871–882. DOI: [10.1109/ICDE.2017.136](https://doi.org/10.1109/ICDE.2017.136).
- [163] J. Kennedy and R. Eberhart. “Particle swarm optimization”. In: *Proceedings of ICNN’95 - International Conference on Neural Networks*. Vol. 4. 1995, 1942–1948 vol.4. DOI: [10.1109/ICNN.1995.488968](https://doi.org/10.1109/ICNN.1995.488968).
- [164] Hedar Abdel-Rahman and Fukushima Masao. “Derivative-Free Filter Simulated Annealing Method for Constrained Continuous Global Optimization”. In: *Journal of Global Optimization* 35.4 (2006), 521–549. DOI: [10.1007/s10898-005-3693-z](https://doi.org/10.1007/s10898-005-3693-z).
- [165] Matheus R. D. Ullmann et al. “Comparison of PSO variants applied to large scale optimization problems”. In: *2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. 2017, pp. 1–6. DOI: [10.1109/LA-CCI.2017.8285728](https://doi.org/10.1109/LA-CCI.2017.8285728).
- [166] Hongming Mo and Yong Deng. “Identifying node importance based on evidence theory in complex networks”. In: *Physica A: Statistical Mechanics and its Applications* 529 (2019), p. 121538. ISSN: 0378-4371. DOI: [10.1016/j.physa.2019.121538](https://doi.org/10.1016/j.physa.2019.121538).
- [167] M.E. J. Newman. “A measure of betweenness centrality based on random walks”. In: *Social Networks* 27.1 (2005), pp. 39–54. ISSN: 0378-8733. DOI: [10.1016/j.socnet.2004.11.009](https://doi.org/10.1016/j.socnet.2004.11.009).

- [168] Santos Matilde, Dai Cai, and Lei Xiujuan. “A Multiobjective Brain Storm Optimization Algorithm Based on Decomposition”. In: *Hindawi Memet. Comput.* pp. 1-14 (2019), p. 100719. DOI: <https://doi.org/10.1155/2019/5301284>.
- [169] K.Z. Gao et al. “A survey on meta-heuristics for solving disassembly line balancing, planning and scheduling problems in remanufacturing”. In: *Swarm and Evolutionary Computation* 57 (2020), p. 100719. ISSN: 2210-6502. DOI: [10.1016/j.swevo.2020.100719](https://doi.org/10.1016/j.swevo.2020.100719).
- [170] Dervis Karaboga and Bahriye Basturk. “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm”. In: *Journal of Global Optimization* 2007 (2007), p. 6. DOI: [10.1007/s10898-007-9149-x](https://doi.org/10.1007/s10898-007-9149-x).
- [171] Yong Deng, Yang Liu, and Deyun Zhou. “An Improved Genetic Algorithm with Initial Population Strategy for Symmetric TSP”. In: *Mathematical Problems in Engineering* 2015 (2015), p. 6. DOI: [10.1155/2015/212794](https://doi.org/10.1155/2015/212794).
- [172] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. “Optimization by Simulated Annealing”. In: *Science* 220.4598 (1983), pp. 671–680. ISSN: 0036-8075. DOI: [10.1126/science.220.4598.671](https://doi.org/10.1126/science.220.4598.671). eprint: <https://science.sciencemag.org/content/220/4598/671.full.pdf>. URL: <https://science.sciencemag.org/content/220/4598/671>.
- [173] Silvester Handy Permana et al. “Comparative Analysis of Pathfinding Algorithms A *, Dijkstra, and BFS on Maze Runner Game”. In: *IJISTECH (International Journal of Information System & Technology)* 1.2 (2018), pp. 1–8. ISSN: 2580-7250. DOI: [10.30645/ijistech.v1i2.7](https://doi.org/10.30645/ijistech.v1i2.7).
- [174] Christian Zammit and Erik-Jan Van Kampen. “Comparison of A* and RRT in real-time 3D path planning of UAVs”. In: *AIAA Scitech 2020 Forum*. 2020. DOI: [10.2514/6.2020-0861](https://doi.org/10.2514/6.2020-0861). eprint: <https://arc.aiaa.org/doi/pdf/10.2514/6.2020-0861>. URL: <https://arc.aiaa.org/doi/abs/10.2514/6.2020-0861>.
- [175] W. Zeng and R. L. Church. “Finding shortest paths on real road networks: the case for A*”. In: *International Journal of Geographical Information Science* 23.4 (2009), pp. 531–543. DOI: [10.1080/13658810801949850](https://doi.org/10.1080/13658810801949850).
- [176] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. “A Formal Basis for the Heuristic Determination of Minimum Cost Paths”. In: *IEEE Transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107. DOI: [10.1109/TSSC.1968.300136](https://doi.org/10.1109/TSSC.1968.300136).
- [177] M. Sniedovich. “Dijkstra’s algorithm revisited: the dynamic programming connexion”. In: *Control and Cybernetics* 35 (2006), pp. 599–620.
- [178] networkx. *networkx*. Accessed: 2020-10-14. URL: <https://networkx.org/>.

- [179] Vedat Toğan and Ayşe T. Daloğlu. “An improved genetic algorithm with initial population strategy and self-adaptive member grouping”. In: *Computers & Structures* 86.11 (2008), pp. 1204–1218. ISSN: 0045-7949. DOI: [10.1016/j.compstruc.2007.11.006](https://doi.org/10.1016/j.compstruc.2007.11.006).
- [180] Xiwang Guo et al. “Lexicographic Multiobjective Scatter Search for the Optimization of Sequence-Dependent Selective Disassembly Subject to Multiresource Constraints”. In: *IEEE Transactions on Cybernetics* 50.7 (2020), pp. 3307–3317. DOI: [10.1109/TCYB.2019.2901834](https://doi.org/10.1109/TCYB.2019.2901834).
- [181] Jian Xu et al. “Automatic Detection of Inshore Ships in High-Resolution Remote Sensing Images Using Robust Invariant Generalized Hough Transform”. In: *IEEE Geoscience and Remote Sensing Letters* 11.12 (2014), pp. 2070–2074. DOI: [10.1109/LGRS.2014.2319082](https://doi.org/10.1109/LGRS.2014.2319082).
- [182] R. Zhang et al. “S-CNN-BASED SHIP DETECTION FROM HIGH-RESOLUTION REMOTE SENSING IMAGES”. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLI-B7 (2016), pp. 423–430. DOI: [10.5194/isprs-archives-XLI-B7-423-2016](https://doi.org/10.5194/isprs-archives-XLI-B7-423-2016).
- [183] M. Tello, C. Lopez-Martinez, and J.J. Mallorqui. “A novel algorithm for ship detection in SAR imagery based on the wavelet transform”. In: *IEEE Geoscience and Remote Sensing Letters* 2.2 (2005), pp. 201–205. DOI: [10.1109/LGRS.2005.845033](https://doi.org/10.1109/LGRS.2005.845033).
- [184] Xing Xiang-wei et al. “Review of Ship Surveillance Technologies Based on High-Resolution Wide-Swath Synthetic Aperture Radar Imaging”. In: *Journal of Radars* 4 (Feb. 2015), 107–121. DOI: [10.12000/JR14144](https://doi.org/10.12000/JR14144).
- [185] Shlomo Greenberg et al. “Region-of-interest-based algorithm for automatic target detection in infrared images”. In: *Optical Engineering* 44 (2005), p. 077002.
- [186] Wenbing Tao, Hai Jin, and Jin Liu. “Unified mean shift segmentation and graph region merging algorithm for infrared ship target segmentation”. In: *Optical Engineering* 46.12 (2007), pp. 1–7. DOI: [10.1117/1.2823159](https://doi.org/10.1117/1.2823159). URL: <https://doi.org/10.1117/1.2823159>.
- [187] Zhenfeng Shao et al. “SeaShips: A Large-Scale Precisely Annotated Dataset for Ship Detection”. In: *IEEE Transactions on Multimedia* 20.10 (2018), pp. 2593–2604. DOI: [10.1109/TMM.2018.2865686](https://doi.org/10.1109/TMM.2018.2865686).
- [188] Bogdan Iancu et al. “ABOships - An Inshore and Offshore Maritime Vessel Detection Dataset with Precise Annotations”. English. In: *Remote Sensing* 13.5 (Feb. 2021), pp. 1–17. ISSN: 2072-4292. DOI: <https://doi.org/10.3390/rs13050988>.
- [189] Yan Song, Bo He, and Peng Liu. “Real-Time Object Detection for AUVs Using Self-Cascaded Convolutional Neural Networks”. In: *IEEE Journal of Oceanic Engineering* 46.1 (2021), pp. 56–67. DOI: [10.1109/JOE.2019.2950974](https://doi.org/10.1109/JOE.2019.2950974).

- [190] David G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: 60.2 (2004), pp. 91–110. DOI: [10.1023/b:visi.0000029664.99615.94](https://doi.org/10.1023/b:visi.0000029664.99615.94). URL: <https://doi.org/10.1023%2Fb%3Avisi.0000029664.99615.94>.
- [191] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. 2005, 886–893 vol. 1. DOI: [10.1109/CVPR.2005.177](https://doi.org/10.1109/CVPR.2005.177).
- [192] R. Lienhart and J. Maydt. “An extended set of Haar-like features for rapid object detection”. In: *Proceedings. International Conference on Image Processing*. Vol. 1. 2002, pp. I–I. DOI: [10.1109/ICIP.2002.1038171](https://doi.org/10.1109/ICIP.2002.1038171).
- [193] Corinna Cortes and Vladimir Naumovich Vapnik. “Support-Vector Networks”. In: *Machine Learning* 20 (2004), pp. 273–297.
- [194] Ji Zhu et al. “Multi-class AdaBoost”. In: *Statistics and its interface* 2 (Feb. 2006). DOI: [10.4310/SII.2009.v2.n3.a8](https://doi.org/10.4310/SII.2009.v2.n3.a8).
- [195] Pedro F. Felzenszwalb et al. “Object Detection with Discriminatively Trained Part-Based Models”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.9 (2010), pp. 1627–1645. DOI: [10.1109/TPAMI.2009.167](https://doi.org/10.1109/TPAMI.2009.167).
- [196] Zhong-Qiu Zhao et al. “Object Detection With Deep Learning: A Review”. In: *IEEE Transactions on Neural Networks and Learning Systems* 30.11 (2019), pp. 3212–3232. DOI: [10.1109/TNNLS.2018.2876865](https://doi.org/10.1109/TNNLS.2018.2876865).
- [197] Ahmad Arinaldi, Jaka Arya Pradana, and Arlan Arventa Gurusinga. “Detection and classification of vehicles for traffic video analytics”. In: *Procedia Computer Science* 144 (2018). INNS Conference on Big Data and Deep Learning, pp. 259–268. ISSN: 1877-0509. DOI: <https://doi.org/10.1016/j.procs.2018.10.527>.
- [198] Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015.
- [199] Ross B. Girshick. “Fast R-CNN”. In: *2015 IEEE International Conference on Computer Vision (ICCV)* (2015), pp. 1440–1448.
- [200] Wei Liu et al. “SSD: Single Shot MultiBox Detector”. In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 21–37. ISBN: 978-3-319-46448-0.
- [201] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788. DOI: [10.1109/CVPR.2016.91](https://doi.org/10.1109/CVPR.2016.91).

- [202] Mabel M. Zhang et al. “VAIS: A dataset for recognizing maritime imagery in the visible and infrared spectrums”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2015, pp. 10–16. DOI: [10.1109/CVPRW.2015.7301291](https://doi.org/10.1109/CVPRW.2015.7301291).
- [203] Erhan Gundogdu et al. “MARVEL: A large-scale image dataset for maritime vessels”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 13th Asian Conference on Computer Vision, ACCV 2016 ; Conference date: 20-11-2016 Through 24-11-2016. Springer Verlag, 2017, pp. 165–180. ISBN: 9783319541921. DOI: [10.1007/978-3-319-54193-8_11](https://doi.org/10.1007/978-3-319-54193-8_11).
- [204] Fleetmon. *Vessel Photo Database*. Accessed: 2021-10-08. URL: <https://www.fleetmon.com/>.
- [205] Marinetraffic. *Vessel Photo*. Accessed: 2021-10-08. URL: <https://www.marinetraffic.com/en/photos/of/ships>.
- [206] Several photographers. *User contributed ship photos and vessels gallery*. Accessed: 2021-10-08. URL: <https://www.vesselfinder.com/>.
- [207] Dilip K. Prasad. *Singapore Maritime dataset(SMD)*. Accessed: 2021-10-08. URL: <https://www.sites.google.com/site/dilipprasad/home/singapore-maritime-dataset>.
- [208] Domenico D. Bloisi et al. “ARGOS-Venice Boat Classification”. In: *Advanced Video and Signal Based Surveillance (AVSS), 2015 12th IEEE International Conference on*. 2015, pp. 1–6. DOI: [10.1109/AVSS.2015.7301727](https://doi.org/10.1109/AVSS.2015.7301727).
- [209] Aurélio Campilho, Fakhri Karray, and Bart M. ter Haar Romeny, eds. *Image Analysis and Recognition - 15th International Conference, ICIAR 2018, Póvoa de Varzim, Portugal, June 27-29, 2018, Proceedings*. Vol. 10882. Lecture Notes in Computer Science. Springer, 2018. ISBN: 978-3-319-93000-8. DOI: [10.1007/978-3-319-93000-8](https://doi.org/10.1007/978-3-319-93000-8). URL: <https://doi.org/10.1007/978-3-319-93000-8>.
- [210] Domenico D Bloisi et al. “ARGOS-Venice Boat Classification”. In: *2015 12th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2015, pp. 1–6. ISBN: 9781467376327.
- [211] Xingang Pan et al. “Spatial As Deep: Spatial CNN for Traffic Scene Understanding”. In: *CoRR abs/1712.06080* (2017).
- [212] Zhe Chen and Zijing Chen. “RBNet: A Deep Neural Network for Unified Road and Road Boundary Detection”. In: *Neural Information Processing*. Ed. by Derong Liu et al. Cham: Springer International Publishing, 2017, pp. 677–687. ISBN: 978-3-319-70087-8.

- [213] Gabriel L. Oliveira, Wolfram Burgard, and Thomas Brox. “Efficient deep models for monocular road segmentation”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 4885–4891. DOI: [10.1109/IROS.2016.7759717](https://doi.org/10.1109/IROS.2016.7759717).
- [214] Yecheng Lyu and Xinming Huang. “RoadNet-v2: A 10 ms Road Segmentation Using Spatial Sequence Layer”. In: *ArXiv abs/1808.04450* (2018).
- [215] Junyu Gao, Qi Wang, and Yuan Yuan. “Embedding structured contour and location prior in siamesed fully convolutional networks for road detection”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 219–224. DOI: [10.1109/ICRA.2017.7989027](https://doi.org/10.1109/ICRA.2017.7989027).
- [216] Farnoush Zohourian et al. “Superpixel-based Road Segmentation for Real-time Systems using CNN”. In: *VISIGRAPP*. 2018.
- [217] Cano Juan C. et al. “Safety Risk Analysis of Unmanned Ships in Inland Rivers Based on a Fuzzy Bayesian Network”. In: *Journal of Advanced Transportation* (2019), pp. 259–268. ISSN: 0197-6729. DOI: [10.1155/2019/4057195](https://doi.org/10.1155/2019/4057195).
- [218] tzutalin. *Labelimg*. Accessed: 2021-10-14. URL: <https://github.com/tzutalin/labelImg>.
- [219] Mark Everingham et al. “The Pascal Visual Object Classes Challenge: A Retrospective”. In: *International Journal of Computer Vision* 111 (2014), pp. 98–136.
- [220] Martín Abadi et al. “TensorFlow: A System for Large-Scale Machine Learning”. In: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. Savannah, GA: USENIX Association, 2016, pp. 265–283. ISBN: 978-1-931971-33-1.
- [221] NVIDIA. *RTX. IT’S ON. GEFORCE RTX 2080 Ti*. <https://www.nvidia.com/fr-fr/geforce/graphics-cards/rtx-2080-ti/>. Accessed: 2021-10-19.
- [222] NVIDIA DEVELOPER. *NVIDIA cuDNN*. <https://developer.nvidia.com/cudnn>. Accessed: 2021-10-19.
- [223] Barret Zoph et al. “Learning Data Augmentation Strategies for Object Detection”. In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi et al. Cham: Springer International Publishing, 2020, pp. 566–583. ISBN: 978-3-030-58583-9.
- [224] Tong Yang et al. “MetaAnchor: Learning to Detect Objects with Customized Anchors”. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems. NIPS’18*. Montréal, Canada: Curran Associates Inc., 2018, 318–328.
- [225] Matlab mathworks. *Estimate Anchor Boxes From Training Data*. Accessed: 2021-09-14. URL: <https://fr.mathworks.com/help/vision/ug/estimate-anchor-boxes-from-training-data.html>.
- [226] Ross Girshick et al. “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Nov. 2014). DOI: [10.1109/CVPR.2014.81](https://doi.org/10.1109/CVPR.2014.81).

- [227] Huaizu Jiang and Erik Learned-Miller. “Face Detection with the Faster R-CNN”. In: *2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)*. 2017, pp. 650–657. DOI: [10.1109/FG.2017.82](https://doi.org/10.1109/FG.2017.82).
- [228] Jane Hung and Anne Carpenter. “Applying Faster R-CNN for Object Detection on Malaria Images”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2018.
- [229] Yuhua Chen et al. “Domain Adaptive Faster R-CNN for Object Detection in the Wild”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3339–3348. DOI: [10.1109/CVPR.2018.00352](https://doi.org/10.1109/CVPR.2018.00352).
- [230] Yundong Li et al. “Building Damage Detection from Post-Event Aerial Imagery Using Single Shot Multibox Detector”. In: *Applied Sciences* 9.6 (2019). ISSN: 2076-3417. DOI: [10.3390/app9061128](https://doi.org/10.3390/app9061128).
- [231] Wen Ma, Xiao Wang, and Jiong Yu. “A Lightweight Feature Fusion Single Shot Multibox Detector for Garbage Detection”. In: *IEEE Access* 8 (2020), pp. 188577–188586. DOI: [10.1109/ACCESS.2020.3031990](https://doi.org/10.1109/ACCESS.2020.3031990).
- [232] Lan Du et al. “Saliency-Guided Single Shot Multibox Detector for Target Detection in SAR Images”. In: *IEEE Transactions on Geoscience and Remote Sensing* 58.5 (2020), pp. 3366–3376. DOI: [10.1109/TGRS.2019.2953936](https://doi.org/10.1109/TGRS.2019.2953936).
- [233] Yuanyuan Wang, Chao Wang, and Hong Zhang. “Combining a single shot multibox detector with transfer learning for ship detection using sentinel-1 SAR images”. In: *Remote Sensing Letters* 9.8 (2018), pp. 780–788. DOI: [10.1080/2150704X.2018.1475770](https://doi.org/10.1080/2150704X.2018.1475770).
- [234] Joseph Redmon and Ali Farhadi. “YOLO9000: Better, Faster, Stronger”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 6517–6525.
- [235] Joseph Redmon and Ali Farhadi. “YOLOv3: An Incremental Improvement”. In: *CoRR* abs/1804.02767 (2018). arXiv: [1804.02767](https://arxiv.org/abs/1804.02767). URL: <http://arxiv.org/abs/1804.02767>.
- [236] J. Redmon and A. Farhadi. *yolo*. <https://pjreddie.com/darknet/yolo>. Accessed: 2021-09-30.
- [237] Rayson Laroca et al. “A Robust Real-Time Automatic License Plate Recognition Based on the YOLO Detector”. In: *2018 International Joint Conference on Neural Networks (IJCNN)*. 2018, pp. 1–10. DOI: [10.1109/IJCNN.2018.8489629](https://doi.org/10.1109/IJCNN.2018.8489629).
- [238] Mohammed A. Al-masni et al. “Simultaneous detection and classification of breast masses in digital mammograms via a deep learning YOLO-based CAD system”. In: *Computer Methods and Programs in Biomedicine* 157 (2018), pp. 85–94. ISSN: 0169-2607. DOI: <https://doi.org/10.1016/j.cmpb.2018.01.017>. URL: <https://www.sciencedirect.com/science/article/pii/S0169260717314980>.

- [239] Wenbo Lan et al. “Pedestrian Detection Based on YOLO Network Model”. In: *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*. 2018, pp. 1547–1551. DOI: [10.1109/ICMA.2018.8484698](https://doi.org/10.1109/ICMA.2018.8484698).
- [240] Tsung-Yi Lin et al. “Focal Loss for Dense Object Detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.2 (2020), pp. 318–327. DOI: [10.1109/TPAMI.2018.2858826](https://doi.org/10.1109/TPAMI.2018.2858826).
- [241] T. Lin et al. “Feature Pyramid Networks for Object Detection”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, 2017, pp. 936–944. DOI: [10.1109/CVPR.2017.106](https://doi.org/10.1109/CVPR.2017.106).
- [242] Tsung-Yi Lin et al. “Focal Loss for Dense Object Detection”. In: *CoRR* abs/1708.02002 (2018).
- [243] Laha Ale, Ning Zhang, and Longzhuang Li. “Road Damage Detection Using RetinaNet”. In: *2018 IEEE International Conference on Big Data (Big Data)*. 2018, pp. 5197–5200. DOI: [10.1109/BigData.2018.8622025](https://doi.org/10.1109/BigData.2018.8622025).
- [244] Mobeen Ahmad, Muhammad Abdullah, and Dongil Han. “Small Object Detection in Aerial Imagery using RetinaNet with Anchor Optimization”. In: *2020 International Conference on Electronics, Information, and Communication (ICEIC)*. 2020, pp. 1–3. DOI: [10.1109/ICEIC49074.2020.9051269](https://doi.org/10.1109/ICEIC49074.2020.9051269).
- [245] Md Ashraful Alam Milton. “Towards Pedestrian Detection Using RetinaNet in ECCV 2018 Wider Pedestrian Detection Challenge”. In: *CoRR* abs/1902.01031 (2019). URL: <http://arxiv.org/abs/1902.01031>.
- [246] Mingming Zhu et al. “Arbitrary-Oriented Ship Detection Based on RetinaNet for Remote Sensing Images”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 14 (2021), pp. 6694–6706. DOI: [10.1109/JSTARS.2021.3082526](https://doi.org/10.1109/JSTARS.2021.3082526).
- [247] Luyang Zhang et al. In: 1757.1 (2021), p. 012070. DOI: [10.1088/1742-6596/1757/1/012070](https://doi.org/10.1088/1742-6596/1757/1/012070). URL: <https://doi.org/10.1088/1742-6596/1757/1/012070>.
- [248] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *CoRR* abs/1409.1556 (2015).
- [249] Olga Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: [10.1007/s11263-015-0816-y](https://doi.org/10.1007/s11263-015-0816-y).
- [250] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [251] Jean Sommet and François Parthiot. “Distances d’arrêt des grands navires en canal”. In: *La Houille Blanche* (2009).

- [252] Zhaowei Cai and Nuno Vasconcelos. “Cascade R-CNN: Delving Into High Quality Object Detection”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 6154–6162.
- [253] Furkan E. Sahin. “Long-Range, High-Resolution Camera Optical Design for Assisted and Autonomous Driving”. In: *Photonics* 6.2 (2019). ISSN: 2304-6732. DOI: [10.3390/photonics6020073](https://doi.org/10.3390/photonics6020073).
- [254] Touqeer Ahmad et al. “An Edge-Less Approach to Horizon Line Detection”. In: *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. 2015, pp. 1095–1102. DOI: [10.1109/ICMLA.2015.67](https://doi.org/10.1109/ICMLA.2015.67).
- [255] Jur van den Berg, Ming Lin, and Dinesh Manocha. “Reciprocal Velocity Obstacles for real-time multi-agent navigation”. In: *2008 IEEE International Conference on Robotics and Automation*. 2008, pp. 1928–1935. DOI: [10.1109/ROBOT.2008.4543489](https://doi.org/10.1109/ROBOT.2008.4543489).
- [256] BEA-TT Bureau d’Enquêtes sur les Accidents de Transport Terrestre. *Les Enquêtes techniques*. Accessed: 2021-08-30. URL: <https://www.bea-tt.developpement-durable.gouv.fr/les-transports-fluviaux-r13.html>.
- [257] Željko Kanović, Vladimir Bugarski, and Todor Backalic. “Ship lock control system optimization using GA, PSO and ABC: A comparative review”. In: *PROMET - Traffic & Transportation* 26 (Feb. 2014). DOI: [10.7307/ptt.v26i1.1475](https://doi.org/10.7307/ptt.v26i1.1475).
- [258] MD. Abdur Rahman et al. “Blockchain-Based Mobile Edge Computing Framework for Secure Therapy Applications”. In: *IEEE Access* 6 (2018), pp. 72469–72478. DOI: [10.1109/ACCESS.2018.2881246](https://doi.org/10.1109/ACCESS.2018.2881246).
- [259] Bouziane Brik, Adlen Ksentini, and Maha Bouaziz. “Federated Learning for UAVs-Enabled Wireless Networks: Use Cases, Challenges, and Open Problems”. In: *IEEE Access* 8 (2020), pp. 53841–53849. DOI: [10.1109/ACCESS.2020.2981430](https://doi.org/10.1109/ACCESS.2020.2981430).
- [260] Yuchuan Fu et al. “An Autonomous Lane-Changing System With Knowledge Accumulation and Transfer Assisted by Vehicular Blockchain”. In: *IEEE Internet of Things Journal* 7.11 (2020), pp. 11123–11136. DOI: [10.1109/JIOT.2020.2994975](https://doi.org/10.1109/JIOT.2020.2994975).
- [261] Muhammad Firdaus and Kyung-Hyune Rhee. “On Blockchain-Enhanced Secure Data Storage and Sharing in Vehicular Edge Computing Networks”. In: *Applied Sciences* 11.1 (2021). ISSN: 2076-3417. DOI: [10.3390/app11010414](https://doi.org/10.3390/app11010414). URL: <https://www.mdpi.com/2076-3417/11/1/414>.
- [262] Rodrigo Roman, Javier Lopez, and Masahiro Mambo. “Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges”. In: *Future Generation Computer Systems* 78 (2018), pp. 680–698. ISSN: 0167-739X. DOI: <https://doi.org/10.1016/j.future.2016.11.009>.
- [263] Peter M. Mell and Timothy Grance. *SP 800-145. The NIST Definition of Cloud Computing*. Tech. rep. Gaithersburg, MD, USA, 2011.

- [264] Mahadev Satyanarayanan. “A Brief History of Cloud Offload: A Personal Journey from Odyssey Through Cyber Foraging to Cloudlets”. In: *GetMobile: Mobile Comp. and Comm.* 18.4 (Jan. 2015), 19–23. ISSN: 2375-0529. DOI: [10.1145/2721914.2721921](https://doi.org/10.1145/2721914.2721921).
- [265] Luis M. Vaquero and Luis Rodero-Merino. “Finding Your Way in the Fog: Towards a Comprehensive Definition of Fog Computing”. In: *SIGCOMM Comput. Commun. Rev.* 44.5 (Oct. 2014), 27–32. ISSN: 0146-4833. DOI: [10.1145/2677046.2677052](https://doi.org/10.1145/2677046.2677052).
- [266] Michael Till Beck and Marco Maier. “Mobile Edge Computing: Challenges for Future Virtual Network Embedding Algorithms”. In: 2014.
- [267] Yating Wang, Ing-Ray Chen, and Ding-Chau Wang. “A Survey of Mobile Cloud Computing Applications: Perspectives and Challenges”. In: *Wireless Personal Communications* 80 (Feb. 2014), pp. 1607–1623. DOI: [10.1007/s11277-014-2102-7](https://doi.org/10.1007/s11277-014-2102-7).
- [268] Zohreh Sanaei et al. “Heterogeneity in Mobile Cloud Computing: Taxonomy and Open Challenges”. In: *IEEE Communications Surveys Tutorials* 16.1 (2014), pp. 369–392. DOI: [10.1109/SURV.2013.050113.00090](https://doi.org/10.1109/SURV.2013.050113.00090).
- [269] Anand Paul et al. “Chapter 6 - Theory and application of vehicular networks”. In: *Intelligent Vehicular Networks and Communications*. Ed. by Anand Paul et al. Elsevier, 2017, pp. 131–159. ISBN: 978-0-12-809266-8. DOI: [10.1016/B978-0-12-809266-8.00006-5](https://doi.org/10.1016/B978-0-12-809266-8.00006-5).
- [270] Bonomi Flavio et al. “Fog Computing and Its Role in the Internet of Things”. In: *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*. MCC '12. Helsinki, Finland: Association for Computing Machinery, 2012, 13–16. ISBN: 9781450315197. DOI: [10.1145/2342509.2342513](https://doi.org/10.1145/2342509.2342513).
- [271] Flavio Bonomi et al. “Fog Computing: A Platform for Internet of Things and Analytics”. In: *Big Data and Internet of Things: A Roadmap for Smart Environments*. Ed. by Nik Bessis and Ciprian Dobre. Cham: Springer International Publishing, 2014, pp. 169–186. ISBN: 978-3-319-05029-4. DOI: [10.1007/978-3-319-05029-4_7](https://doi.org/10.1007/978-3-319-05029-4_7).
- [272] Sabireen H. and Neelanarayanan V. “A Review on Fog Computing: Architecture, Fog with IoT, Algorithms and Research Challenges”. In: *ICT Express* 7.2 (2021), pp. 162–176. ISSN: 2405-9595. DOI: <https://doi.org/10.1016/j.ictex.2021.05.004>.
- [273] Yun Chao Hu et al. “Mobile Edge Computing: A key technology towards 5G”. In: *ETSI white paper* (2015).
- [274] Miranda McClellan, Cristina Cervelló-Pastor, and Sebastià Sallent. “Deep Learning at the Mobile Edge: Opportunities for 5G Networks”. In: *Applied Sciences* 10.14 (2020). ISSN: 2076-3417. DOI: [10.3390/app10144735](https://doi.org/10.3390/app10144735). URL: <https://www.mdpi.com/2076-3417/10/14/4735>.

- [275] Satoshi Nakamoto. “Bitcoin: A Peer-to-Peer Electronic Cash System”. In: *Cryptography Mailing list at <https://metzdowd.com>* (Mar. 2009).
- [276] Xiaofei Wang et al. “In-Edge AI: Intelligentizing Mobile Edge Computing, Caching and Communication by Federated Learning”. In: *IEEE Network* 33.5 (2019), pp. 156–165. DOI: [10.1109/MNET.2019.1800286](https://doi.org/10.1109/MNET.2019.1800286).
- [277] Xi Jiang, Haijun Mao, and Hao Zhang. “Simultaneous Optimization of the Liner Shipping Route and Ship Schedule Designs with Time Windows”. In: *Mathematical Problems in Engineering* 2020 (Dec. 2020), pp. 1–11. DOI: [10.1155/2020/3287973](https://doi.org/10.1155/2020/3287973).
- [278] Helong Wang, Xiao Lang, and Wengang Mao. “Voyage optimization combining genetic algorithm and dynamic programming for fuel/emissions reduction”. In: *Transportation Research Part D: Transport and Environment* 90 (2021), p. 102670. ISSN: 1361-9209. DOI: <https://doi.org/10.1016/j.trd.2020.102670>.
- [279] Laura Walther et al. “Modeling and Optimization Algorithms in Ship Weather Routing”. In: *International Journal of e-Navigation and Maritime Economy* 4 (2016), pp. 31–45. ISSN: 2405-5352. DOI: <https://doi.org/10.1016/j.enavi.2016.06.004>. URL: <https://www.sciencedirect.com/science/article/pii/S2405535216300043>.
- [280] “Byzantine Attacks and its Security Measures in Mobile Adhoc Networks”. In: *IJC-CIE 2016*. 2016.
- [281] Luca Melis et al. “Exploiting Unintended Feature Leakage in Collaborative Learning”. In: *2019 IEEE Symposium on Security and Privacy (SP)* (2019), pp. 691–706.
- [282] Peter Kairouz et al. “Advances and Open Problems in Federated Learning”. In: *CoRR* abs/1912.04977 (2021). arXiv: [1912.04977](https://arxiv.org/abs/1912.04977). URL: <http://arxiv.org/abs/1912.04977>.
- [283] Eugene Bagdasaryan et al. “How To Backdoor Federated Learning”. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, 2020, pp. 2938–2948. URL: <https://proceedings.mlr.press/v108/bagdasaryan20a.html>.
- [284] Arjun Bhagoji et al. “Analyzing Federated Learning through an Adversarial Lens”. In: *Proceedings of the 36th International Conference on Machine Learning, PMLR* 97:634–643 (Nov. 2019).
- [285] Robin C. Geyer, Tassilo Klein, and Moin Nabi. “Differentially Private Federated Learning: A Client Level Perspective”. In: *CoRR* abs/1712.07557 (2018). arXiv: [1712.07557](https://arxiv.org/abs/1712.07557). URL: <http://arxiv.org/abs/1712.07557>.
- [286] Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. “Mitigating Sybils in Federated Learning Poisoning”. In: *ArXiv* abs/1808.04866 (2018).

- [287] M B Zaman et al. “Fuzzy FMEA model for risk evaluation of ship collisions in the Malacca Strait: based on AIS data”. In: *Journal of Simulation* 8.1 (2014), pp. 91–104. DOI: [10.1057/jos.2013.9](https://doi.org/10.1057/jos.2013.9).
- [288] Gunasekaran Raja et al. “Collisionless Fast Pattern Formation Mechanism for Dynamic Number of UAVs”. In: *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*. 2020, pp. 1–6. DOI: [10.1109/GLOBECOM42002.2020.9322085](https://doi.org/10.1109/GLOBECOM42002.2020.9322085).
- [289] D. Zhang et al. “Incorporation of formal safety assessment and Bayesian network in navigational risk estimation of the Yangtze River”. In: *Reliability Engineering & System Safety* 118 (2013), pp. 93–105. ISSN: 0951-8320. DOI: [10.1016/j.res.2013.04.006](https://doi.org/10.1016/j.res.2013.04.006).
- [290] International Hydrographic Organization. *e-Navigation Strategy Implementation Plan MSC.1/Circ.1595,2018*. Accessed: 2021-08-30. URL: <https://iho.int/en/imo-e-navigation-documents>.
- [291] Martin Raifer. *Overpass Turbo*. Accessed: 2021-06-10. URL: <https://overpass-turbo.eu/>.
- [292] Yamin Huang et al. “Ship collision avoidance methods: State-of-the-art”. In: *Safety Science* 121 (2020), pp. 451–473. ISSN: 0925-7535. DOI: [10.1016/j.ssci.2019.09.018](https://doi.org/10.1016/j.ssci.2019.09.018).
- [293] James Paulos et al. “Automated Self-Assembly of Large Maritime Structures by a Team of Robotic Boats”. In: *IEEE Transactions on Automation Science and Engineering* 12.3 (2015), pp. 958–968. DOI: [10.1109/TASE.2015.2416678](https://doi.org/10.1109/TASE.2015.2416678).
- [294] Robert Daily and David M. Bevly. “Harmonic potential field path planning for high speed vehicles”. In: *2008 American Control Conference*. 2008, pp. 4609–4614. DOI: [10.1109/ACC.2008.4587222](https://doi.org/10.1109/ACC.2008.4587222).
- [295] Yamin Huang, Linying Chen, and P.H.A.J.M. van Gelder. “Generalized velocity obstacle algorithm for preventing ship collisions at sea”. In: *Ocean Engineering* 173 (2019), pp. 142–156. ISSN: 0029-8018. DOI: [10.1016/j.oceaneng.2018.12.053](https://doi.org/10.1016/j.oceaneng.2018.12.053).
- [296] Agnieszka Lazarowska. “A new deterministic approach in a decision support system for ship’s trajectory planning”. In: *Expert Systems with Applications* 71 (2017), pp. 469–478. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2016.11.005>. URL: <https://www.sciencedirect.com/science/article/pii/S0957417416306261>.
- [297] CheeKuang Tam and Richard Bucknall. “Cooperative path planning algorithm for marine surface vessels”. In: *Ocean Engineering* 57 (2013), pp. 25–33. ISSN: 0029-8018. DOI: [10.1016/j.oceaneng.2012.09.003](https://doi.org/10.1016/j.oceaneng.2012.09.003).
- [298] Ming-Chung Fang, Kun-Yuan Tsai, and Chih-Chung Fang. “A Simplified Simulation Model of Ship Navigation for Safety and Collision Avoidance in Heavy Traffic Areas”. In: *Journal of Navigation* 71.4 (2018), 837–860. DOI: [10.1017/S0373463317000923](https://doi.org/10.1017/S0373463317000923).

- [299] Mazen Salous, Axel Hahn, and Christian Denker. “COLREGs-Coverage in Collision Avoidance Approaches: Review and Identification of Solutions”. In: Aug. 2016.
- [300] Yonghoon Cho, Jungwook Han, and Jinwhan Kim. “Intent inference of ship maneuvering for automatic ship collision avoidance”. In: *IFAC-PapersOnLine* 51.29 (2018). 11th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles CAMS 2018, pp. 384–388. ISSN: 2405-8963. DOI: [10.1016/j.ifacol.2018.09.457](https://doi.org/10.1016/j.ifacol.2018.09.457).
- [301] Touvia Miloh. “THE GAME OF TWO ELLIPTICAL SHIPS”. In: *Optimal Control Applications & Methods* 4 (1983), pp. 13–29.
- [302] S. P. Hoogendoorn et al. “Modeling Human Behavior in Vessel Maneuver Simulation by Optimal Control and Game Theory”. In: *Transportation Research Record* 2326.1 (2013), pp. 45–53. DOI: [10.3141/2326-07](https://doi.org/10.3141/2326-07).
- [303] Jozef Lisowski. “The Sensitivity of State Differential Game Vessel Traffic Model”. In: *Polish Maritime Research* 23.2 (2016), pp. 14–18. DOI: [10.1515/pomr-2016-0015](https://doi.org/10.1515/pomr-2016-0015).
- [304] Rina MIYAKE et al. “A user test of Automatic Navigational Intention Exchange Support System using an intelligent ship-handling simulator”. In: *IFAC Proceedings Volumes* 46.33 (2013). 9th IFAC Conference on Control Applications in Marine Systems, pp. 97–102. ISSN: 1474-6670. DOI: [10.3182/20130918-4-JP-3022.00045](https://doi.org/10.3182/20130918-4-JP-3022.00045).
- [305] Jinfen Zhang et al. “A distributed anti-collision decision support formulation in multi-ship encounter situations under COLREGs”. In: *Ocean Engineering* 105 (2015), pp. 336–348. ISSN: 0029-8018. DOI: <https://doi.org/10.1016/j.oceaneng.2015.06.054>. URL: <https://www.sciencedirect.com/science/article/pii/S0029801815002978>.
- [306] R.R. Negenborn and J.M. Maestre. “Distributed Model Predictive Control: An overview of features and research opportunities”. In: *Proceedings of the 11th IEEE International Conference on Networking, Sensing and Control*. 2014, pp. 530–535. DOI: [10.1109/ICNSC.2014.6819682](https://doi.org/10.1109/ICNSC.2014.6819682).
- [307] Shijie Li, Jialun Liu, and Rudy R. Negenborn. “Distributed coordination for collision avoidance of multiple ships considering ship maneuverability”. In: *Ocean Engineering* 181 (2019), pp. 212–226. ISSN: 0029-8018. DOI: [10.1016/j.oceaneng.2019.03.054](https://doi.org/10.1016/j.oceaneng.2019.03.054).
- [308] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: [10.1162/neco.1997.9.8.1735](https://doi.org/10.1162/neco.1997.9.8.1735). URL: <https://doi.org/10.1162/neco.1997.9.8.1735>.

- [309] Ahmed Tealab. “Time series forecasting using artificial neural networks methodologies: A systematic review”. In: *Future Computing and Informatics Journal* 3.2 (2018), pp. 334–340. ISSN: 2314-7288. DOI: doi.org/10.1016/j.fcij.2018.10.003. URL: www.sciencedirect.com/science/article/pii/S2314728817300715.
- [310] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Comput.* 9.8 (Nov. 1997), 1735–1780. ISSN: 0899-7667. DOI: [10.1162/neco.1997.9.8.1735](https://doi.org/10.1162/neco.1997.9.8.1735). URL: <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [311] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL: www.deeplearningbook.org.
- [312] Md. Saiful Islam and Emam Hossain. “Foreign Exchange Currency Rate Prediction using a GRU-LSTM Hybrid Network”. In: *Soft Computing Letters* (2020), p. 100009. ISSN: 2666-2221. DOI: doi.org/10.1016/j.socl.2020.100009.
- [313] Ryohei Sawada, Keiji Sato, and Takahiro Majima. “Automatic ship collision avoidance using deep reinforcement learning with LSTM in continuous action spaces”. In: *Journal of Marine Science and Technology* 26 (Aug. 2020). DOI: [10.1007/s00773-020-00755-0](https://doi.org/10.1007/s00773-020-00755-0).
- [314] Klaus Greff et al. “LSTM: A Search Space Odyssey”. In: *IEEE Transactions on Neural Networks and Learning Systems* 28.10 (2017), pp. 2222–2232. DOI: [10.1109/TNNLS.2016.2582924](https://doi.org/10.1109/TNNLS.2016.2582924).
- [315] Hojjat Salehinejad et al. “Recent Advances in Recurrent Neural Networks”. In: (2018). arXiv: [1801.01078 \[cs.NE\]](https://arxiv.org/abs/1801.01078).
- [316] M.V. Valueva et al. “Application of the residue number system to reduce hardware costs of the convolutional neural network implementation”. In: *Mathematics and Computers in Simulation* 177 (2020), pp. 232–243. ISSN: 0378-4754. DOI: doi.org/10.1016/j.matcom.2020.04.031.
- [317] Swapna G, Soman Kp, and Vinayakumar R. “Automated detection of diabetes using CNN and CNN-LSTM network and heart rate signals”. In: *Procedia Computer Science* 132 (2018). International Conference on Computational Intelligence and Data Science, pp. 1253–1262. ISSN: 1877-0509. DOI: <https://doi.org/10.1016/j.procs.2018.05.041>. URL: <https://www.sciencedirect.com/science/article/pii/S1877050918307737>.
- [318] Xingjian Shi et al. “Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting”. In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1. NIPS’15*. Montreal, Canada: MIT Press, 2015, 802–810.
- [319] H. Brendan McMahan et al. “Communication-Efficient Learning of Deep Networks from Decentralized Data”. In: *Artificial Intelligence and Statistics*. 2017, pp. 1273–1282. URL: academic.microsoft.com/paper/2541884796.

- [320] Tom O'Malley et al. *KerasTuner*. 2019. URL: <https://github.com/keras-team/keras-tuner>.
- [321] Tong Di. "The way to give full play to navigation capability of Three Gorges Project ship-lock". In: *Yangtze River* (2013).
- [322] Shan Liang et al. "Vessel traffic scheduling method for the controlled waterways in the upper Yangtze River". In: *Ocean Engineering* 172 (2019), pp. 96–104. ISSN: 0029-8018. DOI: [10.1016/j.oceaneng.2018.11.025](https://doi.org/10.1016/j.oceaneng.2018.11.025).
- [323] Yao Deng, Dian Sheng, and Baoli Liu. "Managing ship lock congestion in an inland waterway: A bottleneck model with a service time window". In: *Transport Policy* 112 (2021), pp. 142–161. ISSN: 0967-070X. DOI: [10.1016/j.tranpol.2021.08.017](https://doi.org/10.1016/j.tranpol.2021.08.017).
- [324] Xiaoping Wang et al. "An analysis on convergence of data-driven approach to ship lock scheduling". In: *Mathematics and Computers in Simulation* 88 (2013), pp. 31–38. ISSN: 0378-4754. DOI: <https://doi.org/10.1016/j.matcom.2013.03.005>. URL: <https://www.sciencedirect.com/science/article/pii/S0378475413000463>.
- [325] Vladimir Bugarski, Todor Bačkalić, and Uroš Kuzmanov. "Fuzzy decision support system for ship lock control". In: *Expert Systems with Applications* 40.10 (2013), pp. 3953–3960. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2012.12.101>. URL: <https://www.sciencedirect.com/science/article/pii/S095741741300002X>.
- [326] Robert M. Naus. "Optimal sequencing in the presence of setup times for tow/barge traffic through a river lock". In: *European Journal of Operational Research* 187.3 (2008), pp. 1268–1281. ISSN: 0377-2217. DOI: [10.1016/j.ejor.2006.06.071](https://doi.org/10.1016/j.ejor.2006.06.071).
- [327] Ward Passchyn et al. "The lockmaster's problem". In: *European Journal of Operational Research* 251.2 (2016), pp. 432–441. ISSN: 0377-2217. DOI: [10.1016/j.ejor.2015.12.007](https://doi.org/10.1016/j.ejor.2015.12.007).
- [328] Ji Bin et al. "Coordinated optimized scheduling of locks and transshipment in inland waterway transportation using binary NSGA-II". In: *International Transactions in Operational Research* 27.3 (2020), pp. 1501–1525. DOI: [10.1111/itor.12720](https://doi.org/10.1111/itor.12720). URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/itor.12720>.
- [329] Luis Fanjul-Peyro. "Models and an exact method for the Unrelated Parallel Machine scheduling problem with setups and resources". In: *Expert Systems with Applications: X* 5 (2020), p. 100022. ISSN: 2590-1885.
- [330] Zhang Yu et al. "Statistical analysis of vessel waiting time and lockage time on the Upper Mississippi River". In: *Maritime Economics & Logistics* 17.4 (2015), pp. 416–439. DOI: [10.1057/me1.2015.30](https://doi.org/10.1057/me1.2015.30).

- [331] M.R. Garey, D.S. Johnson, and L. Stockmeyer. “Some simplified NP-complete graph problems”. In: *Theoretical Computer Science* 1.3 (1976), pp. 237–267. ISSN: 0304-3975. DOI: [https://doi.org/10.1016/0304-3975\(76\)90059-1](https://doi.org/10.1016/0304-3975(76)90059-1). URL: <https://www.sciencedirect.com/science/article/pii/S0304397576900591>.
- [332] Ribal Atallah, Chadi Assi, and Maurice Khabbaz. “Deep reinforcement learning-based scheduling for roadside communication networks”. In: *2017 15th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*. 2017, pp. 1–8. DOI: [10.23919/WIOPT.2017.7959912](https://doi.org/10.23919/WIOPT.2017.7959912).
- [333] B. Peng et al. “Decentralized Scheduling for Cooperative Localization With Deep Reinforcement Learning”. In: *IEEE Transactions on Vehicular Technology* 68.5 (2019), pp. 4295–4305. DOI: [10.1109/TVT.2019.2913695](https://doi.org/10.1109/TVT.2019.2913695).
- [334] Di Zhang et al. *RLScheduler: An Automated HPC Batch Job Scheduler Using Reinforcement Learning*. 2020. arXiv: [1910.08925 \[cs.DC\]](https://arxiv.org/abs/1910.08925).
- [335] Angela Barriga et al. “A Comparative Study of Reinforcement Learning Techniques to Repair Models”. In: *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings. MODELS '20*. New York, NY, USA: Association for Computing Machinery, 2020. ISBN: 9781450381352. DOI: [10.1145/3417990.3421395](https://doi.org/10.1145/3417990.3421395).
- [336] Angela Barriga, Adrian Rutle, and Rogardt Heldal. “Improving Model Repair through Experience Sharing”. In: *Journal of Object Technology* 19.2 (July 2020). Ed. by Richard Paige and Antonio Vallecillo. The 16th European Conference on Modelling Foundations and Applications (ECMFA 2020), 13:1–21. ISSN: 1660-1769. DOI: [10.5381/jot.2020.19.2.a13](https://doi.org/10.5381/jot.2020.19.2.a13).
- [337] Lan Jiang, Hongyun Huang, and Zuohua Ding. “Path planning for intelligent robots based on deep Q-learning with experience replay and heuristic knowledge”. In: *IEEE/CAA Journal of Automatica Sinica* 7.4 (2020), pp. 1179–1189. DOI: [10.1109/JAS.2019.1911732](https://doi.org/10.1109/JAS.2019.1911732).
- [338] Christos A. Kontovas. “The Green Ship Routing and Scheduling Problem (GSRSP): A conceptual approach”. In: *Transportation Research Part D: Transport and Environment* 31 (2014), pp. 61–69. ISSN: 1361-9209. DOI: [10.1016/j.trd.2014.05.014](https://doi.org/10.1016/j.trd.2014.05.014).
- [339] Moritz Buchem, Julian Arthur Pawel Golak, and Alexander Grigoriev. “Vessel velocity decisions in inland waterway transportation under uncertainty”. In: *European Journal of Operational Research* 296.2 (2022), pp. 669–678. ISSN: 0377-2217. DOI: <https://doi.org/10.1016/j.ejor.2021.04.026>. URL: <https://www.sciencedirect.com/science/article/pii/S0377221721003404>.

- [340] Abhra Chanda et al. “CO₂ effluxes from an urban tidal river flowing through two of the most populated and polluted cities of India”. In: *Environmental Science and Pollution Research* 27 (2020), pp. 30093–30107.
- [341] Tao Zhang et al. “Rainfall possibly disturbs the diurnal pattern of CO₂ degassing in the Lijiang River, SW China”. In: *Journal of Hydrology* 590 (2020), p. 125540. ISSN: 0022-1694. DOI: [10.1016/j.jhydrol.2020.125540](https://doi.org/10.1016/j.jhydrol.2020.125540).