Doctoral theses at NTNU, 2019:230

Bjørn-Olav Holtung Eriksen

Collision Avoidance and Motion Control for Autonomous Surface Vehicles

Doctoral Thesis

Bjørn-Olav Holtung Eriksen

ISBN 978-82-326-4056-0 (printed version)
ISBN 978-82-326-4057-7 (electronic version)
ISSN 1503-8181

Doctoral theses at NTNU, 2019:230

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

NTNU

NTNU
Norwegian University of
Science and Technology

Bjørn-Olav Holtung Eriksen

Collision Avoidance and Motion Control for Autonomous Surface Vehicles

Thesis for the degree of Philosophiae Doctor

Trondheim, August 2019

Norwegian University of Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology
and Electrical Engineering
Department of Engineering Cybernetics

ISBN 978-82-326-4056-0 (printed version)
ISBN 978-82-326-4057-7 (electronic version)
ISSN 1503-8181

ITK-report: 2019-6-W

Doctoral theses at NTNU, 2019:230

MILJØMERKET 241 TRYKKSAK 731

Printed by Skipnes Kommunikasjon as

SUMMARY

This thesis presents topics related to collision avoidance (COLAV) and motion control for autonomous surface vehicles (ASVs). The thesis contains a collection of nine publications, of which six are peer-reviewed conference articles, two are journal articles and one is a book chapter. In addition, it contains an introduction providing context to the topic of COLAV at sea as well as explaining the relationship between the publications.

When designing a control system for an ASV, one can approach the COLAV problem in two distinct ways: using a single algorithm for handling all aspects of the problem, or employing a hybrid architecture with multiple algorithms to exploit their complementary strengths at different layers in the architecture. We have chosen the latter approach, resulting in a modular system that can be tailored and extended with new functionalities and algorithms. In particular, our COLAV system has three layers, distributing the task of COLAV, and its inherent motion planning tasks, at three distinct responsiveness and completeness layers.

The top layer is responsible for performing energy-optimized path planning in a global setting, finding a trajectory throughout an environment populated by static obstacles such as land, islands, shallow waters and navigational marks. The planning algorithm uses a vessel model together with ocean current information to produce an energy-optimized trajectory, which minimizes the required energy to reach the goal.

The middle layer follows the trajectory specified by the top layer, while avoiding moving obstacles in a predictable fashion in accordance with the International Regulations for Preventing Collisions at Sea (COLREGs). The COLREGs contains a set of rules on how vessels shall maneuver in situations where there is a risk of collision. Specifically, the developed algorithm used at this layer considers COLREGs rules 8, 13–16 and parts of Rule 17.

The bottom layer follows the trajectory specified by the middle layer, while ensuring that commands specified for the vessel controller are feasible with respect to the vessel's capabilities. This layer also handles emergency situations, such as obstacles detected late or behaving dangerously, and situations where the middle layer fails to find a solution. In particular, this thesis includes two algorithms for the bottom layer, which are tested in several full-scale experiments using a radar-based system for detection and tracking of obstacles. The first algorithm, based on the dynamic window (DW) algorithm, have problems with the amount of noise present in the obstacle estimates provided by the tracking system, as well as having other limitations. The second algorithm, named the branching-course model predictive control (BC-MPC) algorithm, addresses the weaknesses of the DW algorithm, and is the algorithm found to be most suitable for the bottom layer. The BC-MPC algorithm ensures compliance to the remaining parts of COLREGs Rule 17, while also considering rules 8 and 13–15, making our COLAV system compliant with COLREGs rules 8 and 13–17.

A COLAV system is highly dependent on a well-performing vessel controller. Relevant work to that end in this thesis has been directed towards high-speed ASVs, which commonly operate both in the displacement, semi-displacement and planing regions. Existing methods for modeling such vessels are complex, and not well suited for controller design. Therefore, a method for modeling and identification of high-speed ASVs is also proposed. This method is used to create a model of a high-speed ASV, shown to be valid for the displacement, semi-displacement and planing regions. This model is used to design two vessel controllers that utilize model-based feedforward terms, shown to outperform traditional controllers in full-scale experiments.

Summing up, this thesis contributes at all the three described COLAV layers, in addition to modeling, identification and control of high-speed ASVs, and represents a step on the road to autonomy at sea.

PREFACE

This thesis consists of research I conducted during the period 2015–2019, and is submitted as a partial fulfillment of the requirements for the degree of philosophiae doctor (PhD) at the Norwegian University of Science and Technology (NTNU). The work has been carried out at the Department of Engineering Cybernetics (ITK), with Dr. Morten Breivik as my main supervisor and Dr. Edmund F. Brekke as my co-supervisor. I have been working as part of the Autosea project, lead by Dr. Brekke, which is focusing on collision avoidance and sensor fusion for autonomous surface vehicles. The Autosea project has been collaboratively funded by the Norwegian Research Council (project number 244116), Kongsberg Maritime, DNV GL and Maritime Robotics (MR). My work has also been supported by the Centre for Autonomous Marine Operations and Systems (NTNU AMOS), funded by the Norwegian Research Council (project number 223254). Without the support of them all, the thesis in your hand would not have existed – thank you!

I have always been interested in engineering, technology and practical work, and I thank my family which encouraged this throughout my youth. In particular, my father Tor and my grandfather Bjarne (nicknamed “Bibbi”) nurtured this by helping me build things from birdhouses and shelves, to audio amplifiers and a combined doorbell and alarm system for my room. With this in mind, vocational studies was a natural choice for me when approaching high school, which led me to a profession certificate as an automation engineer in 2009. After a couple of years, I realized that I wanted to further my education. I started a BSc in computer science and industrial automation at the Telemark University College in 2010, with the thought that I would not pursue any more than this degree. Finishing my BSc in 2013, I obviously had forgotten all about this and started a MSc in engineering cybernetics at NTNU. Finishing my MSc in 2015, I expected to return to the industry, but instead I started on a PhD in engineering cybernetics which I am now finishing. Looking back, it has been an interesting journey, and I am very glad that I pursued a vocational career in high school – I do not think I would have managed to end up where I am today without it.

Like many other PhD candidates, my motivation during the studies has been riding a never-ending roller-coaster. The work has had periods with high stress levels and thoughts questioning the usefulness of my work, but I have been lucky to be able to angle my work experimentally – successful experiments have always put a smile on my face. I would like to thank my main supervisor Dr. Morten Breivik who gave me the opportunity to take my PhD at the department. He has guided me throughout my research, supported my ideas and restored my faith in myself when I have been in doubt. I'm also grateful for my co-supervisor Dr. Edmund Brekke, who has provided valuable inputs during my PhD studies, complementing Morten and myself with a more theoretical and analytical approach to things. I would also like to thank my fellow PhD candidates at ITK, who I have shared considerable amounts of coffee, tea and beer with, having good discussions and laughs together. In particular, I would like to thank Dr. Mathias H. Arbo, Erik F. Wilthil, Andreas L. Flåten, Kristoffer Gryte and Håkon H. Helgesen, who started their PhD studies at the same time as me. As part of the Autosea project, I have had the pleasure of working closely with Andreas and Erik, a collaboration I have enjoyed to a great extent. Mathias, thank you for putting up with me as your office mate – I am forever thankful for our friendship and all the fruitful discussions we have had, it has helped keeping me reasonably sane. I would like to express a special thanks to Arild Hepsø and Kenan Trnka at MR for providing assistance in performing experiments, and also to Glenn Bitar and Mikkel E. N. Sørensen who I have had the pleasure of collaborating with. Glenn, our discussions have been somewhat heated at times, which I believe have brought out the best in us both – I have really enjoyed working with you.

I would also like to thank my mother and father, Synnøve and Tor, and my brother Tor-Christian for supporting me throughout the years. Marie, thank you for lightening up my days with cheer, joy, adventures and love. I'm looking forward to more of this together with you.

CONTENTS

ABBREVIATIONS

AIS automatic identification system. 17, 18, 23, 37

AMV autonomous marine vehicle. 4

ARPA automatic radar plotting aid. 18, 23

ASV autonomous surface vehicle. iii, iv, 2–6, 12, 23, 25, 27–33, 35, 39–41

AUV autonomous underwater vehicle. 31

BC-MPC branching-course model predictive control. iv, 7, 32–34, 41

CARACaS control architecture for robotic agent command and sensing. 23, 24

COLAV collision avoidance. iii, iv, 4, 6, 7, 9–12, 14, 17, 18, 20–25, 27–29, 31, 32, 34–37, 39–41

COLREGs International Regulations for Preventing Collisions at Sea. iii, iv, 4, 7, 11, 13, 14, 22–25, 27–29, 32–34, 36, 37, 39–41

CPA closest point of approach. 19, 37

CV cross validation. 213

CVM constant velocity model. 37

DOF degree of freedom. 30, 31

DP dynamic positioning. 1, 2

DW dynamic window. iv, 7, 11, 23, 31, 32, 40

FB feedback. 30

FBL feedback-linearizing. 30

FF feedforward. 30

FF-FB feedforward feedback. 30, 31

FF-FB-C feedforward-feedback course. 31

IMO International Maritime Organization. 14, 17

MASS maritime autonomous surface ship. 4, 5

MPC model predictive control. 12, 34, 38, 40, 41

NCDM neighbor course distribution method. 37

NLP nonlinear program. 21, 34, 35, 40

PDF probability density function. 37

PI proportional-integral. 30, 31

RRT rapidly exploring random tree. 12, 23

SPNS single point neighbor search. 37

USV unmanned surface vehicle. 4, 5

VHF very high frequency. 17, 21

VO velocity obstacle. 11, 20, 21, 24

Chapter 1

INTRODUCTION

This chapter contains a brief motivation for the topics covered in this thesis, a summary of the main contributions, and finally an outline of the rest of the thesis.

1.1 Motivation

Automation has for a long time been an everyday part of human society. It is commonly associated with automatic manufacturing with minimum human effort, but encompasses everything from the float regulator in a toilet to numerically controlled milling machines. In general, the term is often defined as using machines or computers instead of humans to perform a certain task. The most apparent historic event marking the introduction of automation might be the Industrial Revolution, when manufacturing processes transitioned from hand production methods to using machines. Since then, the amount and complexity of automation in society has steadily increased. When automation reaches a certain complexity level, people tend to rather use the term *autonomy*. This can be exemplified by the automotive industry, where technologies such as cruise control and anti-lock braking system are regarded as automation, while the complete control of the vehicle is considered as autonomy.

Automation is not a new concept for ship control. In fact, automatic control of ships appeared as early as 1922, marked by the installation of a mechanical *gyropilot* on board the cargo vessel *Munargo* [1]. This gyropilot, commonly named the Metal-Mike and developed by Sperry Gyroscope Company, was the world's first maritime autopilot, steering the ship's rudder to control the heading. This was tightly coupled with the introduction of the gyrocompass, enabling reliable measurement of a ship's heading [2]. In the 1960s, dynamic positioning (DP) systems controlling a ship's position with high accuracy started to appear. The first DP systems were relatively simple, but a development spearheaded by the Norwegian professor Jens G. Balchen[†] was soon to revolutionize the DP systems by using Kalman-filtering techniques for improving the performance and reliability. This work resulted

[†]Jens Glad Balchen (1926-2009) was a Norwegian professor considered to be the father of Engineering Cybernetics in Norway. In 1954, he founded what was to become

in Kongsberg Weapons Factory, today the Kongsberg Group, releasing the Albatross DP system in 1977 [4], which soon became the market-leading DP system.

Autonomy has for a long time been researched for underwater and space operations, where limited communication bandwidths or delays often prohibit direct human control of the vehicles. Such operations depend on autonomy simply in order to be realizable, and are typically performed by a human specifying a mission which the vehicle executes autonomously. In this fashion, the human is still in control of the operation, but relieved of from moment to moment control decisions. In addition to being an enabling technology, autonomy has several other benefits. This includes increased convenience by automating dull tasks, increased safety by reducing the number of human errors, and possibilities for reduced environmental impact by optimizing vehicle motion. This have led to autonomous technology entering domains other than space and underwater applications. The technology has also shown promise in domains that do not have severe communication constraints, as exemplified by the research efforts on autonomous cars made by the automotive industry [5].

Increased safety of personnel and reduced operational costs have since the mid 2000s motivated research on autonomous surface vehicles (ASVs) for defense-related use [6]. The U.S. Navy recently demonstrated their technology when the *Sea Hunter*, depicted in Fig. 1.1, made a round-trip journey from San Diego, California to Pearl Harbor, Hawaii without a crew on board [7].

Research on ASVs is also going on in the commercial sector, where ASVs present attractive possibilities in reducing the cost of e.g. seabed survey operations. This have traditionally been performed by large manned vessels, but small ASVs like the Telemetron ASV, shown in Fig. 1.2, have potential for reducing the cost of these operations. Currently, focus is also picking up on transportation of people by autonomous ferries, which can take advantage of waterways for urban transportation and reduce travel time [9].

In later years, the larger maritime companies have also come to realize the advantages of autonomy at sea, directed towards applications such as autonomous passenger and cargo transport. An example is the Yara Birkeland project in Norway, where an autonomous, electrically-powered cargo vessel, depicted in Fig. 1.3, will replace approximately 40000 diesel-powered truckloads of fertilizer per year by 2022 [10]. Another example is the world's first autonomous car ferry *Falco*, developed by Rolls-Royce Commercial Marine

---

the Department of Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU) [3].

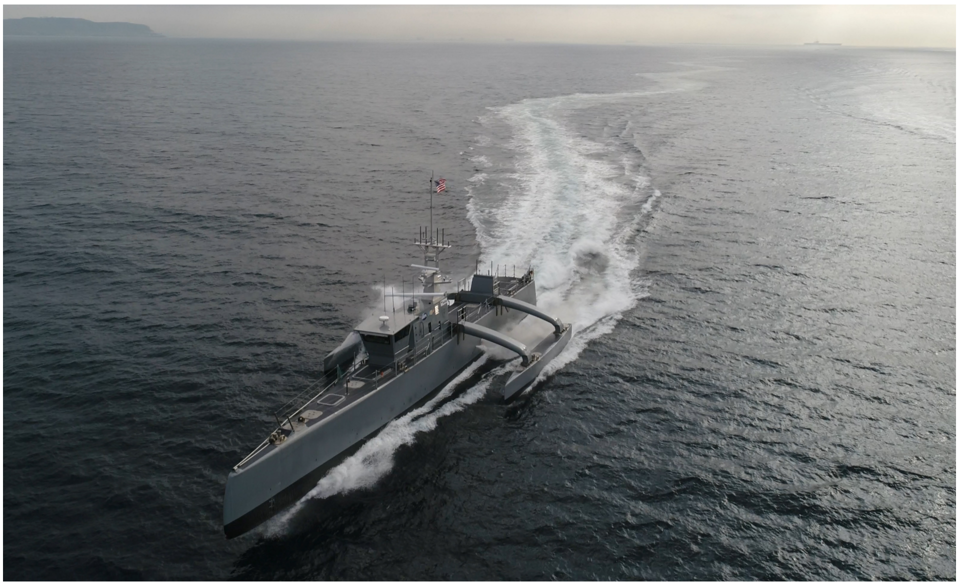


Figure 1.1: DARPA's *Sea Hunter* at sea tests in February 2017. The *Sea Hunter* is a 132-foot long ASV designed for detecting and tracking submarines for time periods over weeks or months [8]. Courtesy of Defense Advanced Research Projects Agency (DARPA).

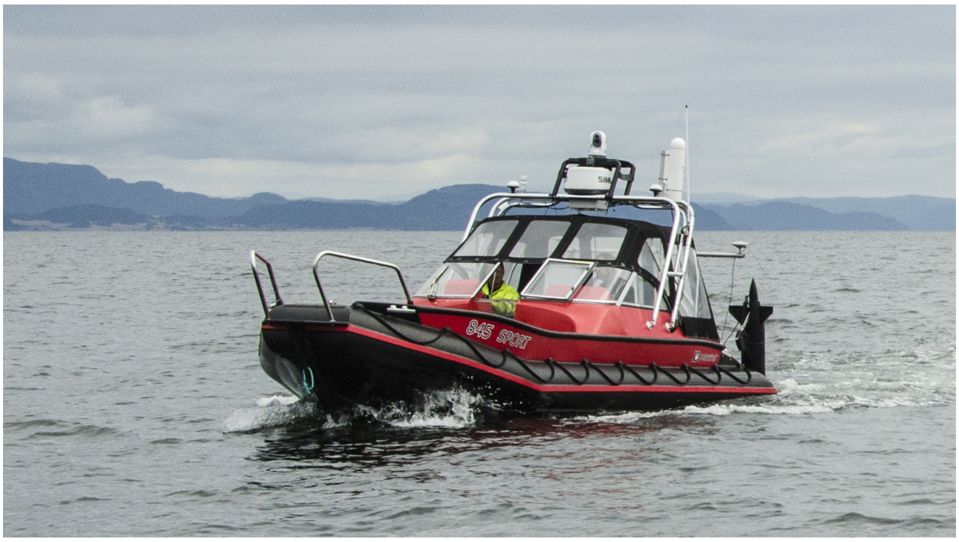


Figure 1.2: The Telemetron ASV, an 8.45 m long dual-use vessel designed for both manned and unmanned operations. Courtesy of Maritime Robotics.

Figure 1.3: An illustration of the Yara Birkeland vessel, which is an example of a MASS. Courtesy of Marin Teknikk and Kongsberg Maritime.

(recently bought by Kongsberg Maritime) and Finferries, which navigated autonomously in Finland in 2018 [11]. The Finnish company Wärtsilä also demonstrated autonomous transit and docking of a car ferry in Norway in 2018 [12]. Improved cost efficiency and reduced environmental impact due to lower fuel consumption may be the most obvious benefits for these applications, but there is also a significant potential for increased safety. Reports state that human errors are the cause of more than 75% of maritime accidents [13, 14], a number that autonomous technology most likely can reduce.

In order to employ ASVs in areas where other vessels are present, the ASVs must be able to avoid collisions while adhering to the International Regulations for Preventing Collisions at Sea (COLREGs). The COLREGs contains a set of rules describing how vessels should maneuver in situations where a risk of collision exists [15]. This requires developing a robust collision avoidance (COLAV) system, which is an important enabling technology to realize the full potential of ASVs.

Several terms are often interchanged in the literature on control and autonomy of vessels at sea. Some of these are autonomous marine vehicle (AMV), ASV, maritime autonomous surface ship (MASS) and unmanned surface vehicle (USV). In this regard, the term AMV refers to a general class of autonomous vehicles, operating in the maritime domain on or below the surface. Furthermore, ASV is a subclass of AMV, encapsulating autonomous vessels at the sea surface. The term MASS represents a subclass of ASV, typically used by the maritime industry and referring to vessels longer than 10–12 m. Also notice that autonomous does not necessarily mean unmanned. Analogous to autonomous cars, manned vessels can use autonomous systems supervised by humans on board. Furthermore, USV is a term not directly coupled to ASV: USVs are unmanned vessels typically operated autonomously or by remote control. This thesis will primarily use the terms vessel and ASV.

1.2 The Autosea project

During my PhD, I have worked as part of the Autosea project (2015–2019), which is a knowledge-building project in collaboration between NTNU, DNV GL, Kongsberg Maritime and Maritime Robotics, focusing on sensor fusion and collision avoidance for ASVs. The Autosea project is the first larger project led by NTNU focusing on ASVs.

As shown in Fig. 1.4, the project is divided into two major parts: a sensor fusion module and a collision avoidance module, which delineate natural divisions of labor. I have worked on the collision avoidance module, while other people in the Autosea team has focused on the sensor fusion module.

The Autosea project has had a significant focus on experimental verification, enabled by the industry partners providing easy access to test platforms and sensor systems. In this thesis, results from several full-scale experiments are presented. These experiments have depended on a radar-based tracking system for detecting and tracking obstacles. This system has been developed by Andreas L. Flåten and Erik F. Wilthil, who have worked as other PhD candidates in the Autosea project, and is described in depth by Wilthil [16] and Wilthil, Flåten, and Brekke [17].

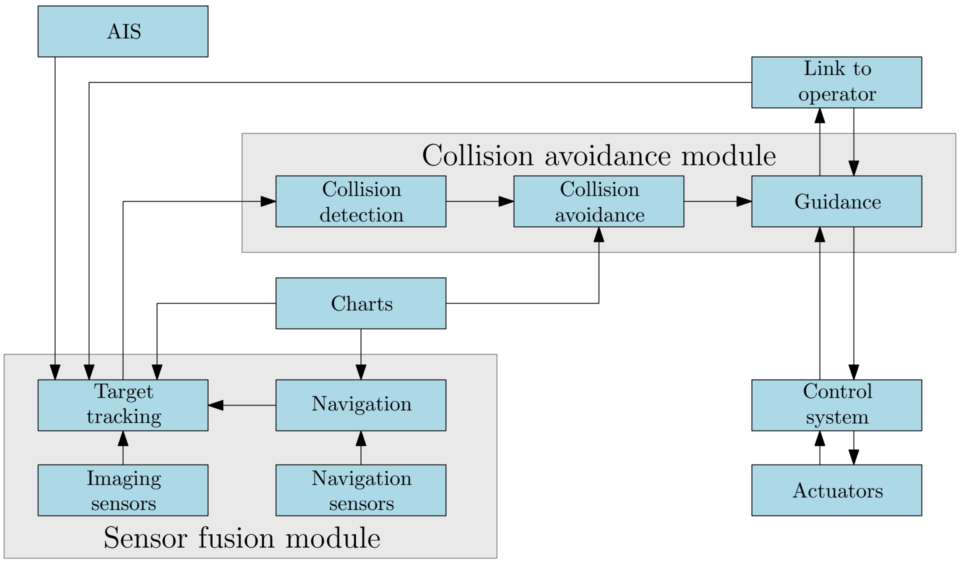


Figure 1.4: A control architecture for an ASV. The sensor fusion module considers navigation and target tracking, and provides the collision avoidance system with obstacle estimates. The collision avoidance module performs path planning and guidance, while generating obstacle avoidance maneuvers based on the information from the tracking system. Notice that the collision avoidance system developed in this thesis has a different structure, but the figure serves as an illustration of the relation between the sensor fusion and collision avoidance modules.

1.3 Contributions at a glance

Several algorithms and methods for collision avoidance and motion control for ASVs have been developed as parts of this thesis. The contributions are discussed in depth in Chapter 3, and is summarized as the development of:

- A vessel model and a procedure for identifying model parameters for high-speed ASVs operating in the displacement, semi-displacement and planing regions.

- Two vessel controllers for high-speed ASVs utilizing model-based feed-forward terms based on an identified model of the Telemetron ASV. The vessel controllers are compared to traditional controllers in full-scale experiments.

- A hybrid COLAV architecture with a three layered COLAV system. The architecture is shown to demonstrate energy-optimized path planning and avoidance of static and moving obstacles in compliance with rules 8 and 13–17 of the COLREGs, including handling of emergency situations.

- Two algorithms for short-term COLAV: a modified dynamic window (DW) algorithm and the branching-course model predictive control (BC-MPC) algorithm. Both algorithms are tested in full-scale experiments with radar-based detection and tracking of obstacles.
- One algorithm for mid-level COLAV, which is interfaced to an existing algorithm for energy-optimized path planning.
- A state machine for interpreting the COLREGs and labeling obstacles with relevant rules.

1.4 Publications

This section lists the publications which have been written during the work on this thesis. The publications are ordered chronologically, and the recommended (thematic) reading order is: Paper B, Paper E, Paper A, Paper D, Paper F, Paper H, Paper C, Paper G and finally Paper I, where the work culminates.

Journal publications and book chapters

Paper B **B.-O. H. Eriksen** and M. Breivik, "Modeling, identification and control of high-speed ASVs: Theory and experiments", in *Sensing and Control for Autonomous Vehicles: Applications to Land, Water and Air Vehicles*, T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer, Eds. Springer International Publishing, 2017, pp. 407–431, ISBN: 978-3-319-55372-6

Paper F **B.-O. H. Eriksen**, M. Breivik, E. F. Wilthil, A. L. Flåten, and E. F. Brekke, "The branching-course MPC algorithm for maritime collision avoidance", 2019, Accepted for publication in Journal of Field Robotics, available at https://arxiv.org/abs/1907.00039

Paper I **B.-O. H. Eriksen**, G. Bitar, M. Breivik, and A. M. Lekkas, "Hybrid collision avoidance for ASVs compliant with COLREGs rules 8 and 13–17", 2019, Submitted to Frontiers in Robotics and AI, available at https://arxiv.org/abs/1907.00198

Conference publications

Paper A **B.-O. H. Eriksen**, M. Breivik, K. Y. Pettersen, and M. S. Wiig, "A modified dynamic window algorithm for horizontal collision avoidance for AUVs", in *Proc. of the 2016 IEEE Conference on Control Applications (CCA)*, (Buenos Aires, Argentina), 2016, pp. 499–506

Paper C **B.-O. H. Eriksen** and M. Breivik, "MPC-based mid-level collision avoidance for ASVs using nonlinear programming", in *Proc. of the 1st IEEE Conference on Control Technology and Applications (CCTA)*, (Mauna Lani, Hawai'i, USA), 2017, pp. 766–772

Paper D **B.-O. H. Eriksen**, E. F. Wilthil, A. L. Flåten, E. F. Brekke, and M. Breivik, "Radar-based maritime collision avoidance using dynamic window", in *Proc. of the 2018 IEEE Aerospace Conference*, (Big Sky, Montana, USA), 2018, pp. 1–9

Paper E **B.-O. H. Eriksen** and M. Breivik, "A model-based speed and course controller for high-speed ASVs", in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*, (Opatija, Croatia), 2018, pp. 317–322

Paper G G. Bitar, **B.-O. H. Eriksen**, A. M. Lekkas, and M. Breivik, "Energy-optimized hybrid collision avoidance for ASVs", in *Proc. of the 17th IEEE European Control Conference (ECC)*, (Naples, Italy), 2019, pp. 2522–2529

Paper H **B.-O. H. Eriksen** and M. Breivik, "Short-term ASV collision avoidance with static and moving obstacles", 2019, Submitted to Modeling, Identification and Control, available at https://arxiv.org/abs/1907.04877

Other publications

- S. Hexeberg, A. L. Flåten, **B.-O. H. Eriksen**, and E. F. Brekke, "AIS-based vessel trajectory prediction", in *Proc. of the 20th IEEE International Conference on Information Fusion (FUSION)*, (Xi'an, China), 2017, pp. 1–8

- M. E. N. Sørensen, M. Breivik, and **B.-O. H. Eriksen**, "A ship heading and speed control concept inherently satisfying actuator constraints", in *Proc. of the 1st IEEE Conference on Control Technology and Applications (CCTA)*, (Mauna Lani, Hawai'i, USA), 2017, pp. 323–330

- B. R. Dalsnes, S. Hexeberg, A. L. Flåten, **B.-O. H. Eriksen**, and E. F. Brekke, "The neighbor course distribution method with Gaussian mixture models for AIS-based vessel trajectory prediction", in *Proc. of the 21st IEEE International Conference on Information Fusion (FUSION)*, (Cambridge, UK), 2018, pp. 580–587

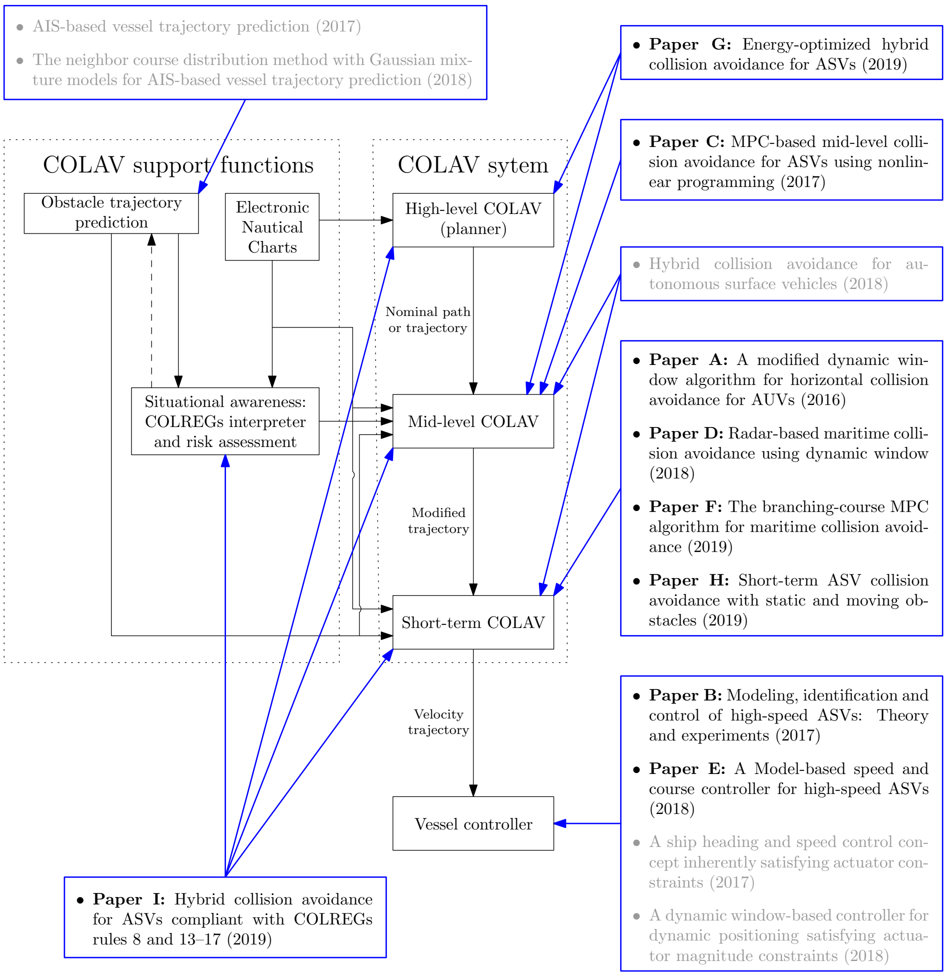
- E. Serigstad, **B.-O. H. Eriksen**, and M. Breivik, "Hybrid collision avoidance for autonomous surface vehicles", in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles (CAMS) 2018*, (Opatija, Croatia), 2018, pp. 1–7

- M. E. N. Sørensen, O. N. Lyngstadaas, **B.-O. H. Eriksen**, and M. Breivik, "A dynamic window-based controller for dynamic positioning satisfying actuator magnitude constraints", in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*, (Opatija, Croatia), 2018, pp. 140–146

All the publications referenced in this section are peer-reviewed. Fig. 1.5 shows the scope of, and relation between, the publications in the hybrid COLAV architecture.

1.5 Outline

The rest of this thesis is structured as follows: Chapter 2 presents background information, Chapter 3 contains an in-depth presentation of the contributions in this thesis and stitches the relationship between the publications together, while Chapter 4 contains conclusions and suggestions for further work. The original publications are included in Chapter 5, while Appendix A contains supplementary material to Paper B.



• AIS-based vessel trajectory prediction (2017)
• The neighbor course distribution method with Gaussian mixture models for AIS-based vessel trajectory prediction (2018)
COLAV support functions
Obstacle trajectory prediction
Electronic Nautical Charts
Situational awareness: COLREGs interpreter and risk assessment
COLAV sytem
High-level COLAV (planner)
Nominal path or trajectory
Mid-level COLAV
Modified trajectory
Short-term COLAV
Velocity trajectory
Vessel controller
• Paper G: Energy-optimized hybrid collision avoidance for ASVs (2019)
• Paper C: MPC-based mid-level collision avoidance for ASVs using nonlinear programming (2017)
• Hybrid collision avoidance for autonomous surface vehicles (2018)
• Paper A: A modified dynamic window algorithm for horizontal collision avoidance for AUVs (2016)
• Paper D: Radar-based maritime collision avoidance using dynamic window (2018)
• Paper F: The branching-course MPC algorithm for maritime collision avoidance (2019)
• Paper H: Short-term ASV collision avoidance with static and moving obstacles (2019)
• Paper B: Modeling, identification and control of high-speed ASVs: Theory and experiments (2017)
• Paper E: A Model-based speed and course controller for high-speed ASVs (2018)
• A ship heading and speed control concept inherently satisfying actuator constraints (2017)
• A dynamic window-based controller for dynamic positioning satisfying actuator magnitude constraints (2018)
• Paper I: Hybrid collision avoidance for ASVs compliant with COLREGs rules 8 and 13–17 (2019)

Figure 1.5: Scope of the articles related to the hybrid COLAV architecture. The articles in gray are works co-authored by the author and not included in the thesis.

Chapter 2

BACKGROUND

This chapter presents a historic background of COLAV and the COLREGs.

2.1 The basis of collision avoidance algorithms

In the early 1980s, researchers started developing COLAV algorithms for use with industrial robot manipulators, marking the start of a new step on the way towards autonomy. One of the first algorithms was presented in 1981, which avoided obstacles by projecting the desired path in the null-space of the obstacles [32]. In 1985, Khatib presented the first version of the now well-known artificial potential field method [33, 34]. This concept is based on forming a potential field where all obstacles give a positive potential based on the distance to the obstacle, and a goal gives a negative potential based on the distance to the goal. A path is found by following the gradient of the potential field, giving motion repelled from the obstacles and attracted towards the goal. In general terms, the artificial potential field method is an unconstrained optimization problem solved by a gradient descent search, where the objective function consists of the potential field. The potential field method presented by Khatib is the basis of many algorithms, e.g. [35–37], and the core idea is still used in COLAV algorithms today [38, 39]. The artificial potential field method does, however, have inherent limitations of being prone to get stuck in local minima, and often having oscillatory behavior in narrow passages and in the presence of obstacles [40]. During the 1990s, researchers pursued alternative methods such as the DW algorithm [37, 41] and the velocity obstacle (VO) algorithm [42]. The DW algorithm is an optimization-based algorithm taking vehicle dynamics into account, while the VO algorithm is a purely kinematic approach based on forward projection of the velocity vectors of the controlled vehicle and obstacles. The VO algorithm calculates cone-shaped regions in the velocity space that will result in collisions. To avoid collision, the algorithm selects a velocity outside of these regions. Its simplicity and intuitive workings have made the VO algorithm gain a lot of popularity, and it is extensively used by researchers today.

The algorithms mentioned so far are often characterized as *reactive* algorithms. The literature characterizes reactive algorithms, sometimes referred to as local or sense–act algorithms, as algorithms considering a limited amount of information (originally only the currently available sensor information), and employing little motion planning over a short time frame. This makes reactive algorithms computationally cheap, and hence well suited for responding to sudden changes in the environment, such as obstacles making unpredictable maneuvers, late detection of obstacles, etc. However, by only considering a limited amount of information, reactive algorithms are prone to making suboptimal decisions in complex situations, and can easily get trapped in local minima. As opposed to reactive algorithms, *deliberate* algorithms, sometimes referred to as global or sense–plan–act algorithms, consider more information and employ a larger amount of planning, typically with longer time horizons than reactive algorithms. Deliberate algorithms, such as the A* or rapidly exploring random tree (RRT) algorithms, choose solutions that may be more globally optimal (given all information), at the cost of an increased computational burden. Optimization-based algorithms with a planning horizon longer than one time step, i.e. model predictive control (MPC)-based algorithms are in general considered as deliberate as they include a concept of planning. As the computational power has increased throughout the years, MPC and optimal control have become increasingly feasible for systems with faster dynamics than traditional process control. This has led MPC to become a more and more preferred tool for designing COLAV algorithms.

The previously clear border between reactive and deliberate algorithms has become somewhat artificial, since increased computational power has allowed for algorithms to incorporate more planning, and few algorithms these days only utilize the currently available sensor information. However, the idea of having some algorithms for robust and fast-reacting short-term planning, and others for mission and cost optimal long-term planning is still relevant. Dividing COLAV algorithms into *short-term* and *long-term* algorithms, where the planning horizon of the algorithms decide the category, makes for a better classification of COLAV algorithms today [19].

A practical COLAV system for an ASV must be able to react to sudden changes in the environment, while also making predictable and optimal maneuvers in a longer time frame. This implies that qualities from both short-term and long-term algorithms are needed. Even though it is possible to design a single algorithm with the required behavior, a more flexible and intuitive approach is to combine short-term and long-term algorithms in a hybrid architecture. Hybrid architectures build upon complementary strengths of different algorithms, and makes it easier for a human operator or supervisor to understand the COLAV system. Some examples of hybrid COLAV architectures are given by Loe [43], Casalino, Turetta, and Simetti

[44], and Švec, Shah, Bertaska, Alvarez, Sinisterra, Ellenrieder, Dhanak, and Gupta [45].

2.2 The maritime rules of the road

The sea has since early times been an important pathway for merchant trade, transportation and information flow for the human species. The maritime industry has always been troubled by the costly problem of collisions between vessels, causing loss of both human lives and material property. This problem has for a long time been tackled by rules on how vessels should maneuver in relation to each other. The first known set of maneuvering rules was made around 400 BC, named the Rhodian Sea Law after the isle of Rhodes [46]. In the 18th century, the common convention was that ships of junior rank should give way to vessels with senior rank. This posed the question of how to decide rank when faced with vessels of different nationality, motivating for a more explicit rule set on how one should avoid collisions at sea. Today's rules date back to 1776, when Admiral Lord Richard Howe issued a signal book to a commander on the vessel HMS *Tartar*. This signal book contained an early reference to rules of the road at sea, and included a statement that in a crossing situation, the vessel with the other vessel to her starboard side should avoid collision by a starboard maneuver [47]. Such an explicit rule reduced the possibility of misunderstanding a situation, and the wording has strong similarities to how the COLREGs describes crossing situations today. The guidelines were, however, not acknowledged to a wide extent at that time.

The size, speed and number of merchant ships grew fast in the early 19th century, resulting in a large increase in the number of accidents. In 1839, a committee in England investigated the causes of 92 accidents involving steamboats, resulting in only 12 being considered as collisions (stated to be a "considerable undercounting" [48, pp. 170]). Anyhow, the committee stated that the lack of a rule set on how to maneuver in collision situations was the main reason behind these 12 collisions. This led several committees to recognize the need for rules of the road regulating how ships should maneuver in close-counter situations. In 1840, the London Trinity House Corporation began lobbying for a common rule set, which the English government passed in 1846 as the Steam Navigation Act [47, 48]. This further materialized in 1862, when England and France agreed on the first international rules for the prevention of collisions at sea [49], which the British Board of Trade drafted in 1860 [47]. In 1884, England revised the rules, and by 1885 other nations such as the United States, Belgium, Germany, France, Japan, Norway and

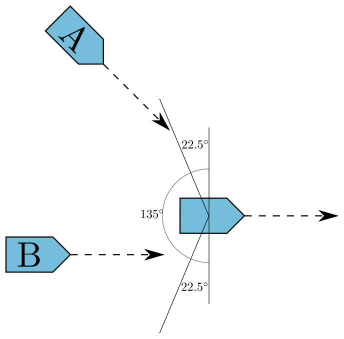
Denmark adopted the new rules [49]. The first international maritime conference was held in Washington, D.C. in 1889, at which minor changes were made to the rules issued in 1884. These rules remained unchanged for over fifty years, and were internationally recognized at an international maritime conference in Brussels in 1910 [47].

Radar was introduced at sea as early as 1937 [50], but it was not until the advancement in radar technology following World War II that civilian ships became equipped with radars. This was initially believed to solve the problem of collisions at sea, but it was quickly seen that the number of collision with ships equipped with radar increased, revealing inherent limitations and skill requirements related to the use of radar for navigation [49]. Techniques from the Navy involving manual plotting of ships relative motion was introduced, but these techniques became less suitable with the drastic increase in ship speed during the 1960s. This led to increased requirements for personnel training, as well as the introduction of new radar technologies in the late 1960s and early 1970s. The use of radar for COLAV was recognized in the rules of the road by the 1960 Safety of Life at Sea conference [47, 49], which was put into action in 1965. In 1963, investigations on the usefulness of traffic separation schemes was initiated after a large increase in loss of ships due to human errors. This led to the introduction of traffic separation schemes in the Dover Strait by the International Maritime Organization (IMO) in 1967, resulting in a significant reduction in the number of collisions involving ships of opposing headings [51]. Following this, the rules of the road were revised again in 1972, when substantial changes were made in relation to the drastically increased use of radar. At the same time, traffic separation schemes were included in the rules [15]. The revised rules came into force in 1977, and is what we today, with minor revisions, refer to as the COLREGs.

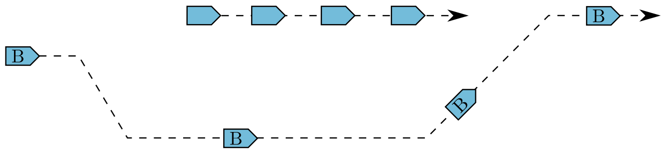
The COLREGs consists of five parts and a total of 37 rules [15], where part B (rules 4–19) contains relevant rules on how vessels in close proximity to each other should maneuver. Of those rules, the most relevant ones are rules 8 and 13–17:

Rule 8 **Action to avoid collision.** This rule requires actions taken to avoid collision to be large enough to be readily observable for other vessels. This implies that one should avoid a series of small incremental changes in speed and/or course. Furthermore, this rule also states that avoidance maneuvers preferably should be performed by course changes, and that maneuvers must be made in ample time.

Rule 13 **Overtaking.** This rule considers overtaking situations, and



(a) Overtaking situation. Both vessel A and B are overtaking the unmarked vessel. Notice that vessel A and B also are in a potential crossing situation.



(b) Resolution of the overtaking situation for vessel B.

Figure 2.1: An example of an overtaking situation, and how it can be solved.

states that a vessel is overtaking another if it approaches the other vessel with a course of more than 22.5° abaft her beam[†]. In such a situation, the overtaking vessel has to stay clear of the overtaken vessel, but there are no requirements on which side of the overtaken vessel it should pass. Fig. 2.1 shows an example of an overtaking situation.

Rule 14 **Head on.** This rule considers head-on situations, which refer to cases where two power-driven vessels approach each other on reciprocal, or nearly reciprocal, courses. In a head-on situation, both vessels should maneuver to starboard, passing port-to-port. There is no explicit definition of what a nearly reciprocal course is, but court decisions indicate that courses opposing each other by ±6° should be considered as nearly reciprocal. Fig. 2.2 shows an example of a head-on situation.

Rule 15 **Crossing.** This rule considers crossing situations, where two

[†]"Abaft" is a nautical preposition interpreted as towards the back of a ship , while the "beam" is the widest section of a vessel. The term "22.5° abaft her beam" hence refers to angles 22.5° behind the widest part of a ship, as illustrated in Fig. 2.1(a).

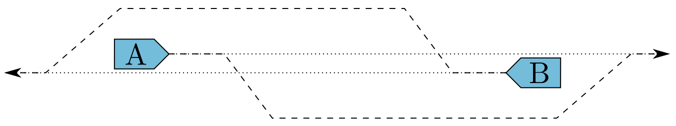
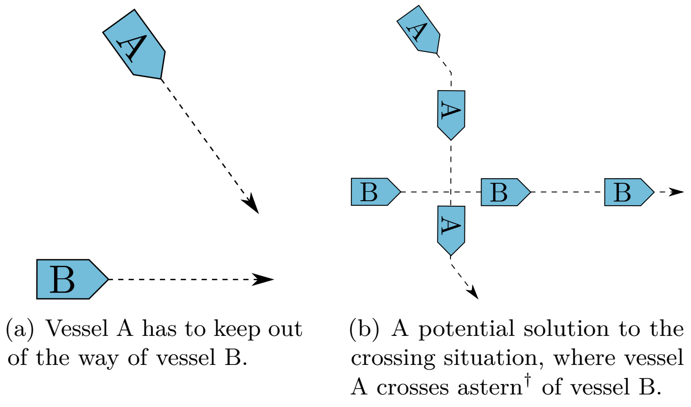


Figure 2.2: An example of a head-on situation, and how it can be solved by both vessels maneuvering to starboard.



(a) Vessel A has to keep out of the way of vessel B.

(b) A potential solution to the crossing situation, where vessel A crosses astern[†] of vessel B.

Figure 2.3: An example of a crossing situation, and how it can be solved.

vessels approaches each other in a situation that is not a head on or an overtaking. In such a situation, the vessel having the other on her starboard side has to keep out of the way of the other, and is deemed the give-way vessel. The other vessel is deemed the stand-on vessel, and has to keep her course and speed. The give-way vessel should preferably avoid passing in front of the stand-on vessel, but this is not strictly forbidden. Fig. 2.3 shows an example of a crossing situation.

Rule 16 **Action by the give-way vessel.** This rule simply states that every vessel that has to maneuver in accordance with the rules should take early and substantial action to avoid collision.

Rule 17 **Action by the stand-on vessel.** This rule has two main parts, where the first requires that a stand-on vessel has to keep its current speed and course in a crossing situation. The second part considers situations where the give-way vessel does not maneuver to avoid collision in accordance with the rules. The stand-on vessel may maneuver if it becomes apparent that such

[†]"Astern" is a nautical adverb meaning behind or towards the rear of a ship.

a situation exists, and must maneuver if the give-way vessel cannot avoid collision alone. If the stand-on vessel chooses to maneuver in a crossing situation, it should avoid maneuvering to port as this can lead to a collision if the give-way vessel maneuvers to starboard.

In addition to the rules listed above, a "complete" COLAV system should also consider rules 6 (safe speed), 9 (narrow channels), 10 (traffic separation schemes), 12 (sailing vessels), 18 (responsibilities between vessels) and 19 (restricted visibility). In particular, Rule 18 is important in defining the responsibilities between different vessel types, e.g. sailing vessels, fishing vessels and vessels with a restricted ability to maneuver.

2.3 Detection and tracking of obstacles at sea

Even though this thesis focuses on COLAV, some attention should be given to the problem of providing obstacle estimates to a COLAV system. In general, we can distinguish between two principles for this:

1. Using systems dependent on infrastructure or communication with other vessels.

2. Using systems not dependent on infrastructure or communication with other vessels, e.g. utilizing exteroceptive sensors.

The obvious benefit of the first principle is the simplicity, with the most apparent system of this category at sea being the automatic identification system (AIS) [52], which transmits information about a vessel's state using the very high frequency (VHF) band. The IMO requires all passenger ships and vessels with a gross tonnage of over 300 to carry AIS transponders, making it easy to receive information about their position, speed and course. AIS transponders do, however, rely on navigation equipment on board the vessels and user-specified information, which results in the possibility of providing inaccurate or invalid data [53]. In addition, one cannot avoid vessels and objects not carrying AIS transponders, e.g. most leisure vessels and kayaks, if only relying on AIS. Naval vessels are also allowed to turn off their AIS transponders. This has been the case in several accidents, e.g. a collision between the Norwegian frigate KMN *Helge Ingstad* and the oil-tanker *Sola TS* on the 8th of November 2018 in Norway, and a collision between the USS *Fitzgerald* and the container ship *ACX Crystal* on the 17th of June 2017 in Japan. It is clear that a robust COLAV system cannot rely on AIS as a sole source of obstacle estimates.

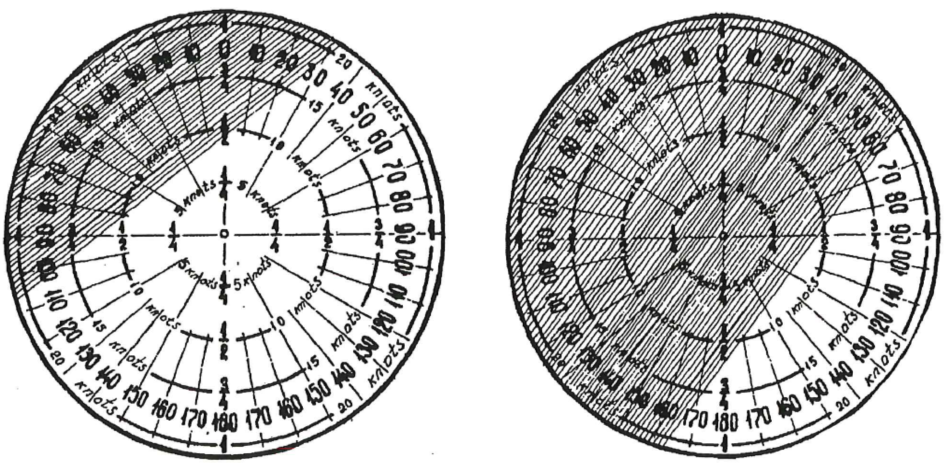
As an alternative to using systems dependent on infrastructure or communication with other vessels, one can employ exteroceptive sensors, such as radars, cameras and lidars, to detect and track other vessels and/or objects. The benefit of this approach is that one can detect vessels, and even objects such as kayaks, swimmers, icebergs, driftwood, etc., without requiring any communication cooperation. On the other hand, exteroceptive sensors provide an added complexity, since one must employ a tracking system in order to provide estimates of the position, speed and course of obstacles. Obstacle estimates provided by such a system will typically also contain more noise than obstacle estimates based on AIS information, which can be challenging to handle [23].

2.4 A historic overview of collision avoidance at sea

This section presents a shallow dive into the ocean of COLAV efforts made by the maritime industry and academia, up until the time I began getting my feet wet. Interested readers are referred to Tam, Bucknall, and Greig [51], Statheros, Howells, and Maier [54], and Campbell, Naeem, and Irwin [55] for additional surveys on the topic.

The introduction of radars on civilian vessels after World War II motivated qualitative research studies on COLAV maneuvers in the 1960s, primarily focused on interpretation of the collision regulations, and their practical applicability [51]. This led to the release of an anti-collision indicator in 1968 [56], an electro-mechanical analogue computer proposing a set of safe courses and speeds, shown in Fig. 2.4. The computations were based on information about the relative bearing, distance and speed of an obstacle, obtained by radar and manually fed into the unit by an operator.

In 1961, researchers began developing automatic radar plotting aid (ARPA) with manual track initialization (MARPA) [49, 50], and at the end of the 1960s fully automatic ARPA units appeared [46, 57, 58]. This paved the way for developing more sophisticated systems for COLAV, since automatically acquired obstacle information became available for *collision avoidance systems*. These early systems do, however, not really justify their name by today's definitions, as will be explained shortly. One of the first such systems, the Norcontrol DataBridge (DB), was developed by the Norwegian company Norcontrol, which today is a part of Kongsberg Maritime. The Norcontrol DB tracked up to eight targets from radar images, and provided position, speed and course estimates of the targets. The system alarmed an operator of collision risks based on the estimates [59]. In addition to this, the



(a) A situation where the ownship[†] would be safe by proceeding with a low speed.

(b) A situation where the ownship have to perform a maneuver in order to be safe.

Figure 2.4: The anti-collision indicator from Mitrofanov [56], where the shaded regions indicate inadmissible combinations of speed and course, while the white regions indicate safe maneuvers. O. Mitrofanov, "An anti-collision indicator", *Journal of Navigation*, vol. 21, no. 2, pp. 163–170, 1968, reproduced with permission.

system could simulate the effect of ownship maneuvers, which could be used as a means of deciding appropriate maneuvers in collision situations. The research leading to this was initiated in 1967, and a prototype of the system, shown in Fig. 2.5, was installed and demonstrated on the dry cargo vessel M/S *Taimyr* in 1969. During the demonstrations, the captain pronounced that he could sail more safely with this system in fog, compared to bright weather without it. Multiple other companies released similar systems in the years to come – Luse [46] states that some six radar-based collision avoidance systems were commercially available in 1972. These early systems typically only provided a watch officer with information about obstacles range, bearing, course, speed and closest point of approach (CPA) [46]. In fact, this functionality is commonly provided by radar sets intended for recreational use on leisure crafts today, meaning that the label *collision avoidance systems* is inappropriate by today's definitions. As the Norcontrol DB, many of the "collision avoidance systems" included simulation functionality, which an operator could use to simulate the effect of a specific maneuver. However, using this to find a suitable maneuver was inefficient as it required trial and error. Given the information from the collision avoidance system, the watch

[†]"Ownship" refers to one's own vessel, i.e. the controlled vessel.

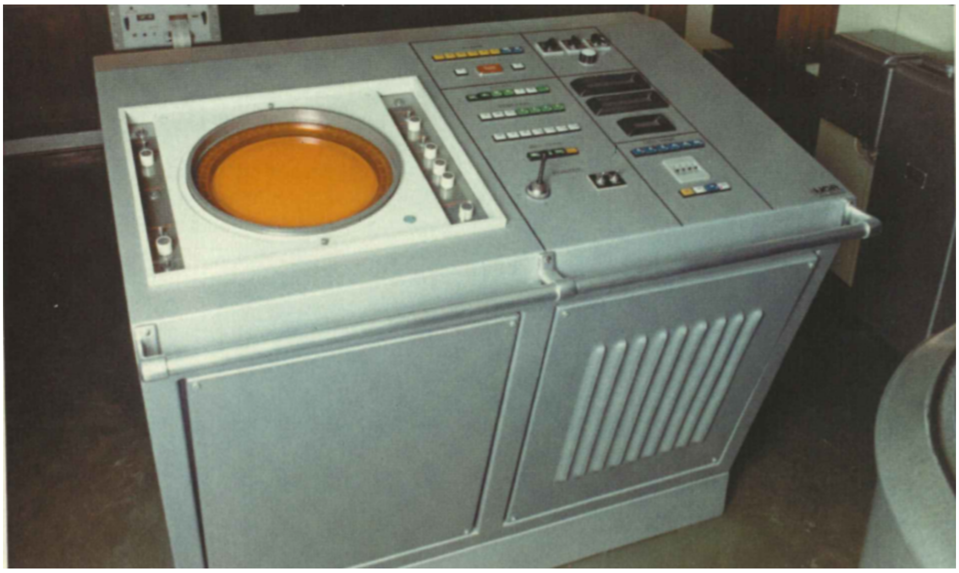


Figure 2.5: The prototype of the Norcontrol DataBridge installed on board the M/S *Taimyr* in 1969. Courtesy of Bjerva [58].

officer was left with the tasks of deciding if there is a risk of collision, decide the applicable maneuver with respect to the rules of the road and finally maneuver the vessel. The Sperry collision avoidance system, developed by Sperry Marine Systems, took this a small step further by using a computer to further interpret the information. It presented the user with a *predicted area of danger*, shown in Fig. 2.6, and two suggested headings in order to pass ahead or astern of an obstacle [60], while assuming that a constant speed was kept. The system made calculations of relative velocities between the vessels, displaying some similarities to the method we today know as VO. The Sperry collision avoidance system was together with the Digiplot collision avoidance system, manufactured by the Iotron Corporation [57], employed on board U.S. Navy ships for testing in the 1970s. In a report from these tests, Gravely jr. [61] states that even though the reliability of the systems were questionable at times, the concept of using COLAV systems could enhance safety in shipping and reduce personnel workload. By 1980, the Sperry collision avoidance system had been installed on board approximately 300 ships worldwide [62].

In the academic community, more complex methods for COLAV were researched in the 1970s, including methods using game theory [63] and optimal control theory [64, 65]. In 1979, de Wit and Oppe [65] applied the Hooke-Jeves direct search method, which is a derivative-free optimization

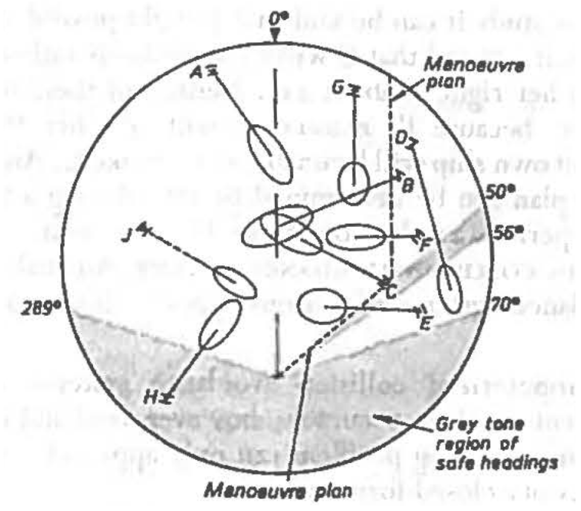


Figure 2.6: The Sperry collision avoidance system display showing a situation with nine obstacles in a vessel-relative coordinate frame. The ellipses are predicted areas of danger, and the gray areas indicate safe headings. R. F. Riggs, "A modern collision avoidance display technique", *Journal of Navigation*, vol. 28, no. 2, pp. 143–155, 1975, reproduced with permission.

method [66], to solve an optimization problem encapsulating COLAV. This has similarities to methods commonly used these days, however with a significantly lower complexity level. Most of the systems developed up until 1980 had neglected the possibility of changing a vessel's speed, and only focused on heading changes. In 1981, Degré and Lefèvre [67] presented an algorithm moving further in the direction of the VO algorithm commonly known and used today. Their approach, shown in Fig. 2.7, generalized the concept of the predicted area of danger to also consider speed changes. Further in the 1980s, the research community sought alternative methods for handling COLAV. This includes a concept of data sharing between ships over VHF [68], an idea of a closed-loop COLAV system for autonomous navigation [69], COLAV when other vessels maneuver [70], and a knowledge-based approach to the COLAV problem [71]. In particular, Dove, Burns, and Stockel [69] mark a new research focus: the systems developed so far were decision support systems, providing information to a watch officer steering the vessel, while Dove, Burns, and Stockel emphasizes that a collision avoidance system could steer the vessel itself.

During the 1990s, the focus on optimization-based methods continued, with increasing complexity as exemplified by Yavin, Frangos, and Miloh [72] and Miele, Wang, Chao, and Dabney [73], where nonlinear programs

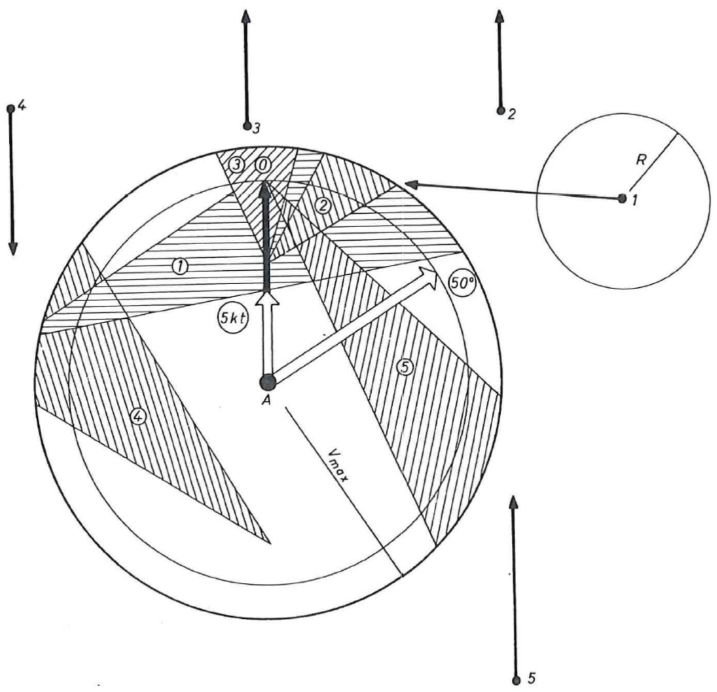


Figure 2.7: The room-to-maneuver algorithm presented by Degré and Lefèvre [67]. The black arrows indicate the speed and course of obstacles, while the shaded regions indicate inadmissible areas of the ownship's velocity space. The arrows and shaded regions correspond to each other as given by the numbers. The circle with radius R around obstacle 1 illustrates a safety region also enforced on the other obstacles. Collision could be avoided by selecting a speed and course outside of the shaded regions. T. Degré and X. Lefèvre, "A collision avoidance system", *Journal of Navigation*, vol. 34, no. 2, pp. 294–302, 1981, reproduced with permission.

(NLPs) were used to formulate the COLAV problem. Other approaches for formulating and solving optimization problems encapsulating COLAV, such as fuzzy logic [74], and genetic and evolutionary algorithms [75, 76], were also researched. It was also reported that radar alone would not be satisfactory for obstacle detection, and Sato and Ishii [77] proposed combining radar with an infrared camera for obstacle detection.

In 2004, Benjamin and Curcio [78] presented a protocol-based algorithm for guidance and COLAV using multi-objective optimization. This algorithm combines multiple functions capturing both guidance and different COLREGs rules in an objective function. To the author's knowledge, this was the first research paper to present closed-loop experimental results, demonstrating their algorithm using two autonomous kayaks [79]. Even though they relied on vessel-to-vessel communication for obtaining obstacle information, greatly simplifying obstacle detection, this marked a point where the research community focused more on practically viable algorithms. The algorithm is included in an open-source large-scale autonomy software named MOOS-IvP [80].

In the mid 2000s, the US Navy and other defense and security organizations became increasingly interested in the use of unmanned vessels at sea. This resulted in an increased focus on COLAV in military-related research institutions. In 2006, the Space and Naval Warfare Systems Center, San Diego suggested a two-layered hybrid approach, dividing the COLAV problem into a near-field reactive component and a far-field deliberate component [6, 81]. The deliberate component avoided obstacles detected by long-range sensors (ARPA, AIS and nautical charts), while the reactive component avoided near-field collisions using raw radar data, nautical charts and several cameras. Their work was done in collaboration with NASA Jet Propulsion Laboratory (JPL), which provided a stereo-camera system used as part of the sensor suite. Opposed to Benjamin, Leonard, Curcio, and Newman [79], this system applied more planning in the sense of introducing a deliberate component planning ahead in future time, and demonstrates a higher complexity level. The idea of hybrid COLAV systems was also pursued in other research communities, e.g. by Loe [43], Casalino, Turetta, and Simetti [44], and Švec, Shah, Bertaska, Alvarez, Sinisterra, Ellenrieder, Dhanak, and Gupta [45]. Loe [43] was, to the author's knowledge, the first to perform full-scale experiments of an autonomous COLAV system partially compliant with the COLREGs. He suggested a two-layered hybrid architecture, using the RRT and DW algorithms, relying on AIS for obtaining obstacle information.

In 2010, Elkins, Sellers, and Monach [82] presented the Autonomous Maritime Navigation (AMN) Project, which investigated the use of ASVs for riverine patrolling. The project was funded by the Naval Surface Warfare Center Combatant Craft Division, and used a system titled control architecture for robotic agent command and sensing (CARACaS) developed at NASA JPL for perception, mission planning and collision avoidance [83]. The AMN project demonstrated the use of a wide range of sensors in an autonomy system, designed for use on several different vessels. Fig. 2.8 shows the combatant maritime vehicle (CMV), one of the vessels used in this project. Huntsberger, Aghazarian, Howard, and Trotz [84] exemplifies further use of the CARACaS system by demonstrating static obstacle avoidance with the next generation of the NASA JPL stereo-camera system used by Larson, Bruch, Halterman, Rogers, and Webster [81].

In 2014, Kuwata, Wolf, Zarzhitsky, and Huntsberger [85] presented an

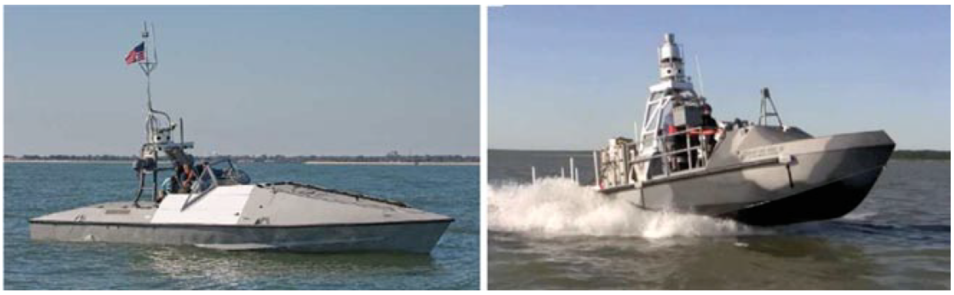


Figure 2.8: Two members of the US Navy autonomous vessel fleet, with the CMV to the right. T. Huntsberger, H. Aghazarian, A. Howard, and D. C. Trotz, "Stereo vision-based navigation for autonomous surface vessels", *Journal of Field Robotics*, vol. 28, no. 1, pp. 3–18, 2011, reproduced with permission.

algorithm based on VO, intended for COLAV in compliance with the COLREGs rules concerning head-on, crossing and overtaking. Kuwata's algorithm was implemented in the CARACaS system and demonstrated in several experiments using radar and stereo-cameras for detecting obstacles. Kuwata, Wolf, Zarzhitsky, and Huntsberger state that to their knowledge, this was the first on-water demonstration of COLAV considering the COLREGs without vehicle-to-vehicle communications. Soon after, Schuster, Blaich, and Reuter [86] presented results from on-water experiments in 2014, where they demonstrated COLAV using a radar for detecting and tracking an obstacle. For COLAV, they use an A*-based algorithm presented by Blaich, Rosenfelder, Schuster, Bittel, and Reuter [87], which uses a domain-based interpretation of the COLREGs [88]. This consists of superimposing a non-symmetrical spatial domain on obstacles, in order to motivate the algorithm to pass obstacles on the "correct" side by making maneuvers on the "wrong" side longer. They state that the algorithm considers the COLREGs, but their results are not compelling in defending this claim. Nevertheless, their work marks a significant contribution outside of the communities funded by defense-related institutions.

In 2016, Johansen, Perez, and Cristofaro [89] proposed the simulation-based model predictive control algorithm, which includes a concept of risk in an optimization-based COLAV algorithm. This algorithm considers a discrete set of course offsets and speed multipliers, which are combined with the output from a guidance system to make the algorithm easily integrated in existing guidance systems. The same year, DARPA's *Sea Hunter* was christened, marking the continued research interest in defense-related institutions [8]. The vessel was built by Leidos and funded by DARPA, which in 2012 requested a US $3 billion grant for developing autonomous technology for submarine tracking [55]. Details of the control and COLAV system is hard to come by, but Leidos states that the path planning system is developed in collaboration with NASA JPL and National Robotics Engineering Center [90]. In 2016, Leidos claimed that the control system had been tested in over 220 simulations and 100 on-water tests evaluating COLREGs scenarios, stating that the COLAV system generally met the expectations with respect to the COLREGs [90].

In 2015, I started my research on COLAV for ASVs. Even though a lot of progress had been made on the topic, the most impressing results were presented by defense-related research institutions, limiting the availability of the research. Results from the academic community were less complete, and generally did not demonstrate convincing results of autonomous navigation and COLAV considering all relevant parts of the COLREGs.

Chapter 3

CONTRIBUTIONS

This chapter contains an in-depth description of the contributions in this thesis.

3.1 A hybrid architecture for collision avoidance

Most of the work in this thesis has been carried out in the scope of a hybrid architecture for COLAV, designed for ASVs – manned and unmanned. The architecture, first published in Paper C, is shown in Fig. 3.1, and contains a three-layered COLAV system, a vessel controller and a number of support functions.

The architecture includes the vessel controller as a component to illustrate the importance of the performance of the underlying vessel controller: if it performs poorly, it may not be able to effectuate the commands specified by the COLAV system with sufficient precision. Therefore, it is important that the vessel controller has a sufficiently good performance so that it does not limit the performance of the COLAV system.

The three layers of the COLAV are the short-term, mid-level and high-level COLAV layers. The high-level COLAV layer, also described as the high-level planner, performs path or trajectory planning from an initial position to a goal position while considering static obstacles. This layer may also take time constraints, energy efficiency, ocean currents, etc., into account, and provides a nominal path or trajectory that the mid-level layer attempts to follow. The layer may run offline, possibly even manually by a human operator, but it can also run online in real time if circumstances such as changing maps, ocean currents, etc., makes replanning necessary. Running the layer offline allows for higher computational requirements, compared to running the layer online in real time. The planning horizon for this layer is typically long, which results in a significant uncertainty associated with estimates of moving obstacles such as e.g. other vessels. Moving obstacles are therefore neglected at this level.

The mid-level COLAV layer inputs a nominal path or trajectory from the high-level COLAV layer, which it attempts to follow while making local adaptations to ensure COLAV with respect to both moving and static obstacles. The COLREGs is a natural part of this layer, since maneuvers

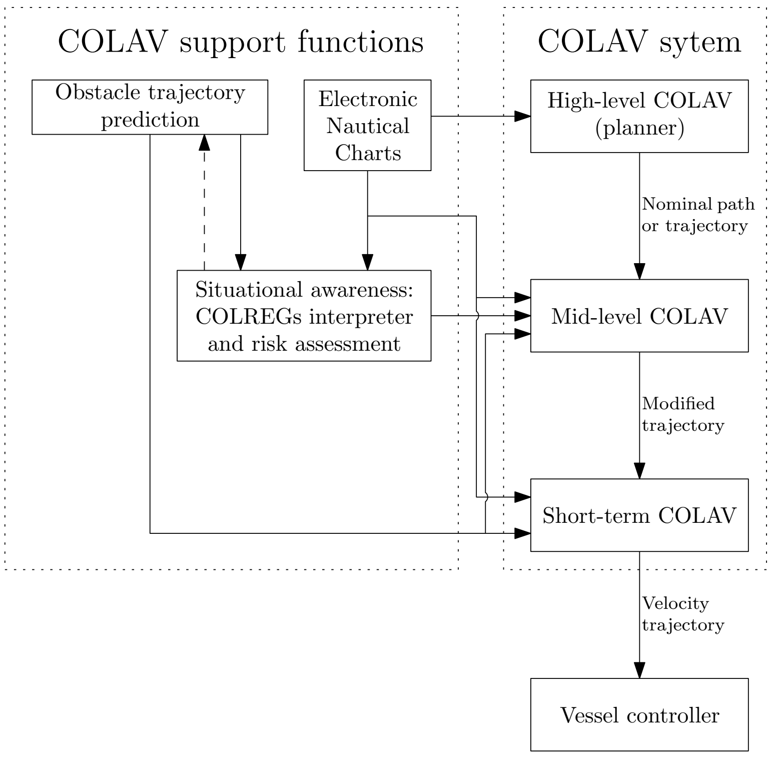

COLAV support functions
COLAV sytem
Obstacle trajectory prediction
Electronic Nautical Charts
High-level COLAV (planner)
Nominal path or trajectory
Situational awareness: COLREGs interpreter and risk assessment
Mid-level COLAV
Modified trajectory
Short-term COLAV
Velocity trajectory
Vessel controller

Figure 3.1: A hybrid architecture for ASV COLAV. The architecture includes a three-layered COLAV system, support functions providing charts, situational awareness and information about future obstacle trajectories, and a vessel controller.

made to avoid other vessels must be made in accordance with the COLREGs. This requires the planning horizon of the layer to be long enough to decide the appropriate action with respect to the COLREGs. The mid-level layer should have a strict enforcement of the COLREGs by e.g. standing on in situations where the ownship has a stand-on obligation, unless a simultaneous give-way obligation must be prioritized. The planning horizon of this layer should be short enough to ensure real-time feasibility, while long enough to allow the layer to make reasoned maneuvers. In particular, one should select the planning horizon sufficiently long to capture the complete length of a COLREGs avoidance maneuver.

The short-term COLAV layer inputs a modified trajectory from the mid-level layer, which it attempts to follow. This layer performs COLAV with respect to obstacles detected too late for the mid-level layer to handle, or which performs sudden maneuvers, while ensuring that the vessel controller is given dynamically feasible commands. The short-term COLAV layer should be computationally robust, making sure that a solution always exists, even in cases where the mid-level layer fails to find a solution. This layer should also handle situations where the ownship is in a stand-on situation, but has to maneuver in accordance with the second part of Rule 17 due to obstacles violating the COLREGs. An example of such a situation is if a give-way vessel fails to avoid collision in a crossing situation where the ownship is deemed the stand-on vessel, e.g. by not maneuvering. Furthermore, such situations may require ignoring the maneuvering aspects of rules 14 and 15, and the short-term layer should therefore have the possibility to ignore these.

The architecture also contains a number of support functions which provide the COLAV system with information. The obstacle trajectory prediction module should provide information about the current position and future trajectory of obstacles. In its simplest form, this can be a system using vessel-to-vessel communication or a tracking system based on exteroceptive sensors, combined with a linear prediction model assuming that obstacles keep their current speed and course. Alternatively, one can apply more sophisticated methods for predicting future trajectories for obstacles. The electronic nautical charts block provides the COLAV system with information about static obstacles such as land, navigation marks, etc., in the vicinity of the ownship. The situational awareness block provides the COLAV system with information about the current situation, such as which COLREGs rules apply for different obstacles, requirements for proceeding with caution, etc.

Paper I populates the hybrid COLAV architecture with three COLAV algorithms, a state machine for interpreting the COLREGs, and a model-based speed and course controller. The architecture is evaluated in simulations, demonstrating good performance and the ability to solve a large variety of different scenarios in compliance with COLREGs rules 8 and 13–17.

3.2 Model identification and vessel controllers

The work in this thesis is performed in a bottom-up approach, initially focusing on designing high-performance vessel controllers to avoid that the vessel controller limits the COLAV system performance. Initially, the focus was on designing a speed and yaw-rate controller, which later was extended to a speed and course controller.

In particular, Paper B presents a method for modeling, identification and control of high-speed ASVs. This method is also applicable to other high-speed vessels operating in the displacement, semi-displacement and planing regions. The foundation of the method is a control-oriented 2 degree of freedom (DOF) non-first principles model, valid for the displacement, semi-displacement and planing regions. A data driven identification procedure populates this model with parameters, based on data from vessel experiments. The modeling and identification method is used to obtain a control-oriented model of the Telemetron ASV, shown to be of high accuracy through a verification experiment. The identified model is used to design four speed and yaw-rate controllers:

- The feedback (FB) controller: a pure proportional-integral (PI) feedback controller with gain scheduling.
- The feedforward (FF) controller: a model-based feedforward controller without any feedback, utilizing both steady-state and acceleration feedforward.
- The feedforward feedback (FF-FB) controller: a combination of the FB and FF controllers.
- The feedback-linearizing (FBL) controller: a traditional feedback-linearizing controller.

These controllers were tested in full-scale experiments in the Trondheimsfjord in Norway on the 13th and 14th of October 2016. The FBL controller displayed poor robustness and resulted in oscillatory vessel behavior, which caused a sensor dropout that aborted the experiment. The oscillatory behavior likely originated from time delays in the control loop, which became an issue when using measurements to select the linearization point in the controller. Due to the practical issues with the FBL controller, this was not used in the rest of the experiments. The FB controller served as a benchmark controller, to compare the other controllers with. Several of the experiments included time-varying references, which a pure feedback controller obviously had problems following. The open-loop FF controller outperformed the FB controller, even when the references reached steady state, displaying the powerful properties of utilizing model-based feedforward terms. The FF-FB controller, combining model-based feedforward and feedback terms, performed even better than the FF controller, and managed to track a time-varying speed and yaw-rate reference to a high degree of precision.

For achieving a certain performance level, model-based feedforward terms reduce the required feedback bandwidth compared to traditional feedback controllers without such terms. This enables a FF-FB controller to have similar performance as a FB controller, with less sensitivity to measurement noise. The COLAV experiments in Paper D used the FF-FB controller as the vessel controller.

In Paper E, the FF-FB controller is extended to control the vessel speed and course instead of the vessel speed and yaw rate. The new controller is named the feedforward-feedback course (FF-FB-C) controller, and was tested in full-scale experiments in the Trondheimsfjord on the 10th of October 2017. In the experiments, the FF-FB-C controller outperformed a PI feedback controller, managing to track a time-varying speed and course reference with high precision. The COLAV experiments in Paper F and Paper H used the FF-FB-C controller as the vessel controller.

3.3 Short-term collision avoidance

Paper A presents a modified DW algorithm for COLAV for autonomous underwater vehicles (AUVs) in the horizontal plane. The modified algorithm introduces a new trajectory prediction method, that compared to the original DW algorithm reduces the mean square prediction error with a factor of 100 when applied to vehicles with second order non-holonomic constraints, such as AUVs and ASVs. The algorithm is reparameterized to be more modular, inputing a desired yaw rate and speed instead of a desired heading and speed. Together with a modified search space, these modifications greatly improve the algorithm performance when applied to a horizontal-plane model of the HUGIN 1000 AUV. This model has the same structure as typical ASV models, making the algorithm a suitable starting point for ASV COLAV.

Paper D presents the first results on closed-loop COLAV, where we coupled the DW algorithm presented in Paper A with the radar-based tracking pipeline developed in the Autosea project. The DW algorithm is reparameterized to use the 2 DOF model of the Telemetron ASV, developed in Paper B. Furthermore, the algorithm is modified to take advantage of the acceleration feedforward capabilities of the FF-FB controller, involving a reformulation of the search space. We present full-scale experimental results from the Trondheimsfjord, performed on the 15th to 19th of May 2017. In these experiments, the target vessel is detected and tracked solely based on radar tracking, without any vessel-to-vessel communication. The experiments revealed that the tracking system provided obstacle estimates with a significant amount of noise, which must be considered when using exteroceptive sensors for obstacle detection and tracking. In particular, the course estimates contained a large amount of noise, which caused problems since the DW algorithm assumes that obstacles will keep their current speed and course to predict the future trajectory of the obstacle. We did, however, successfully avoid collision when applying a smoothing technique on the obstacle estimates, but this also introduced a longer time delay in the sensor system. Based on the experiments, we considered the DW algorithm to be overly sensitive to obstacle estimate noise. In light of this, we decided to focus on developing a new short-term COLAV algorithm, specifically designed to be robust to the expected amount of noise when using exteroceptive sensors for obstacle detection and tracking.

Paper F thus presents a new short-term COLAV algorithm, named the BC-MPC algorithm. In addition to the problems related to obstacle estimate noise, the DW algorithm parameterizes the vessel trajectories using a constant turn rate, assuming that the vessel will keep a constant turn rate for the entire prediction horizon. This does not resemble how vessels maneuver at sea, where one usually either make a clear change in speed and/or course or keep a constant speed and course. The BC-MPC algorithm is designed using sample-based optimization, where the search space consists of a finite number of trajectories parameterized by speed and course. In contrast to other sample-based optimization algorithms for ASV COLAV, the BC-MPC algorithm considers a sequence of maneuvers, enabling the algorithm to consider more complex trajectories than just single-maneuver trajectories. The algorithm complies with COLREGs rules 8, 13 and 17, and favors trajectories following the maneuvering aspects of rules 14 and 15. This is motivated by Rule 17, requiring the stand-on vessel to maneuver to avoid collision in situations where a give-way vessel does not fulfill her requirements to avoid collision, which may require ignoring the maneuvering aspects of rules 14 and 15. If the algorithm chooses to violate these maneuvering aspects, it will pass obstacles with a larger clearance than if the maneuvering aspects are followed. The algorithm was tested on the Telemetron ASV in multiple full-scale experiments, using the radar-based tracking system developed in the Autosea project. The experiments were carried out in the Trondheimsfjord on the 12th of October 2017, and Fig. 3.2 shows a drone photo from the experiments. The experiments demonstrated several distinct COLREGs scenarios, showing that the algorithm managed to solve head-on, crossing from starboard and overtaking situations, in addition to a crossing from port situation where the obstacle did not maneuver. The algorithm chose to maneuver in front of the obstacle in one of the crossing from starboard scenarios, which is a result from the algorithm's loose enforcement of COLREGs rules 14 and 15. Although this was performed with an increased obstacle clearance, and not considered a direct violation of the COLREGs, some criticism should be addressed to the algorithm for not passing behind the vessel. Paper H addresses this particular issue. In addition to the experimental results, Paper F presents a simulation study

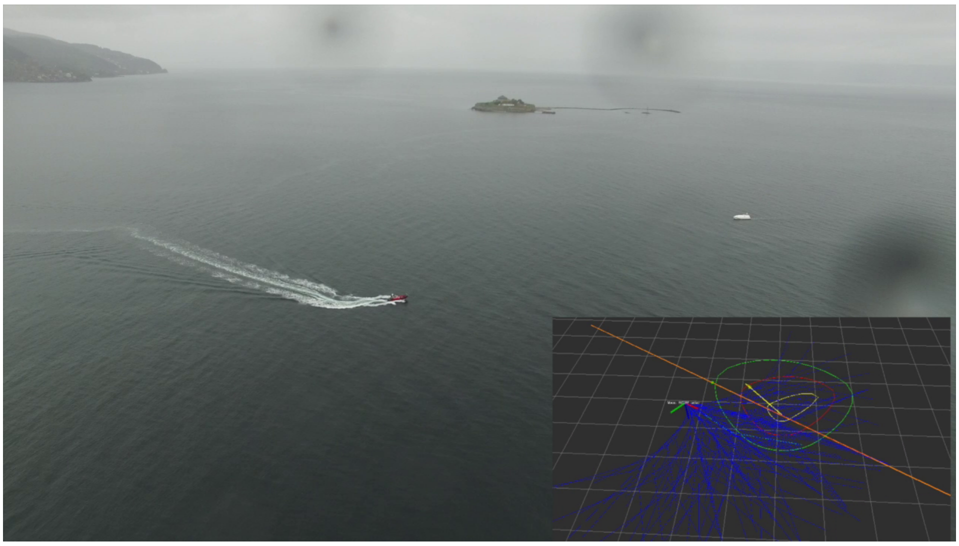


Figure 3.2: A photo from the experiments performed with the BC-MPC algorithm on the 12th of October 2017. The Telemetron ASV is in the middle of the picture, while the obstacle vessel is to the right. The lower right corner shows a visualization of the BC-MPC algorithm. The blue lines represent possible trajectories the algorithm can choose from, the green line represents the selected trajectory, while the orange line represents the desired trajectory. The yellow arrow represents the estimated obstacle position and velocity, while the yellow, red and green regions represent BC-MPC regions. The estimated velocity, in particular the course, fluctuates despite that the obstacle travels steadily along the orange line.

with four multi-obstacle scenarios where the BC-MPC algorithm faces more complex challenges than in the experiments. The scenarios include situations with conflicting COLREGs rules simultaneously active, and maneuvering obstacles.

Paper H extends the BC-MPC algorithm to also consider static obstacles, by including an occupancy grid in the objective function. Using an occupancy grid to model static obstacles enables flexibility in representing arbitrary obstacle shapes, which is beneficial in order to represent coastlines and islands in the vicinity of the ownship. Furthermore, the original version of the algorithm used transitional cost terms to avoid shattering by introducing a penalty for selecting another combination of speed and course than the one selected in the previous iteration. Paper H presents new transitional cost terms, introducing penalties to speed and course changes independently. This motivates the algorithm to still keep the speed constant if changing the course, and vice versa. This, together with alternative tuning parameters compared to the implementation in Paper F, resulted in smoother maneuvers and seemed to make the algorithm prefer crossing behind obstacles in crossing from starboard situations. The algorithm was experimentally validated in the Trondheimsfjord on the 27$^{\text{th}}$ of September 2018, using virtual static obstacles. As in the previous experiments, information about moving obstacles was provided by the radar-based tracking system developed in the Autosea project.

Finally, Paper I improves a minor implementation detail of the BC-MPC algorithm, resulting in less oscillations in areas where the reference trajectory has large course changes.

3.4 Mid-level collision avoidance

Paper C presents an MPC-based mid-level COLAV algorithm. The algorithm is formulated as an NLP, enabling a high level of flexibility in allowing for nonlinear and non-convex objective functions and constraints. The NLP is solved using a gradient-based solver, considering infinite variations of trajectories as opposed to sample-based optimization. The algorithm complies with Rule 8 of the COLREGs, requiring that maneuvers are readily observable for other vessels, implying that sequences of small speed and/or course changes should be avoided.

Paper H improves the numerical properties of the algorithm, and interfaces the algorithm to a high-level planner for energy-optimized path planning. Section 3.5 presents this interface in greater detail.

Paper I extends the mid-level algorithm to also consider COLREGs rules 13–17. The algorithm uses information from a COLREGs interpreter, described in Section 3.6, labeling obstacles as “safe”, “overtaking”, “stand-on”, “give-way”, “head-on” or “emergency” situations. In order to enforce the correct COLREGs behavior in different situations, the NLP is tailored to the current situation as follows:

- Obstacles with “safe” labels do not have any COLREGs requirements, and are included in COLAV constraints.
- Obstacles with “overtaking” labels do not have any COLREGs requirements, and are included in COLAV constraints.
- Obstacles with “stand-on” labels require the ownship to not maneuver. They are therefore not included in the COLAV constraints, and extra terms motivating the algorithm to not change speed and/or course are added to the cost function.

- Obstacles with “give-way” labels require the ownship to avoid collision by passing behind the obstacles. They are included in the COLAV constraints, and potential functions motivating the algorithm to choose trajectories behind each obstacle are added to the cost function.
- Obstacles with “head-on” labels require the ownship to apply starboard maneuvers to avoid collision. They are included in the COLAV constraints, and potential functions motivating the algorithm to choose trajectories on the obstacles’ port sides are added to the cost function.
- Obstacles with “emergency” labels are so close to the ownship, and/or behave so unpredictably, that the mid-level algorithm is unsuited for handling the obstacles. They are therefore not included in the COLAV constraints, and the task of avoiding the obstacles is left to the short-term COLAV algorithm.

Tailoring the NLP to the current situation enables a lot of flexibility. This makes the algorithm easy to extend, for instance to consider special requirements with respect to specific vessel types such as vessels under sail, engaged in fishing or having a restricted ability to maneuver. However, when changing the NLP from iteration to iteration, local minima become a more apparent problem. In order to mitigate this issue, we apply a homotopy scheme when solving the NLP.

3.5 High-level planner

Paper H presents a hybrid architecture consisting of the mid-level algorithm and a high-level planner, conceptualizing the two top layers in the hybrid COLAV system in Fig. 3.1. The high-level planner is based on 2018 Bitar, Breivik, and Lekkas, which is extended to work with high-speed ASVs. The algorithm plans a trajectory from an initial to a goal position, while accounting for static obstacles and minimizing the required energy consumption to reach the goal. The ocean current is an important factor for the energy consumption, and is included in both the mid-level COLAV algorithm and the high-level planner by modeling the relative vessel velocity with respect to the ocean current.

In order to control the energy consumption, the high-level planner specifies a speed in addition to a geometric path, solved by generating a desired trajectory for the mid-level algorithm to follow. However, since the high-level planner does not consider moving obstacles, the speed is the only time-relevant factor of the desired trajectory. This implies that the mid-level algorithm does not need to track the desired trajectory in absolute time – it is sufficient to locally follow the desired speed of the trajectory. Therefore, we define an interface allowing the mid-level algorithm to track the desired trajectory from the high-level planner with a time offset, computed by solving an optimization problem at each time instant the mid-level algorithm is run. In a case where the mid-level algorithm lags behind the desired trajectory, for instance due to circumventing obstacles, a traditional trajectory tracking method would increase the speed to catch up with the desired trajectory. With our interface, we rather adjust the time offset to track a time-offset part of the desired trajectory in such a situation, avoiding a speed increase in order to catch up with the desired trajectory. We describe this as *relative trajectory tracking* in our works. This interface is also generalized for relative trajectory tracking for arbitrary geometrical paths with an assigned speed profile.

3.6 Support functions

The COLAV system has been the main focus during the work in this thesis. However, the support functions, in particular COLREGs interpretation and prediction of future obstacle trajectories, have also received attention.

Interpretation of the COLREGs and deciding which rules apply to different obstacles is important for the mid-level algorithm. Paper I presents a state machine interpreting obstacle situations with respect to the COLREGs, giving input to the mid-level algorithm. A separate state machine is run for each obstacle, and the state-machine labels are:

SF Safe state. This implies that the COLREGs does not enforce any rule with respect to this obstacle.

OT Overtaking state. This implies that COLREGs Rule 13 applies with respect to this obstacle. The state machine does not discriminate whether the ownship is overtaking another vessel or is being overtaken, but this can be done by looking at which vessel has the higher speed.

HO Head-on state. This implies that COLREGs Rule 14 applies with respect to this obstacle.

GW Give-way state. This implies that COLREGs Rule 15 applies with respect to this obstacle, and that the ownship has to give way.

SO Stand-on state. This implies that COLREGs Rule 15 applies with respect to this obstacle, and that the ownship has to stand on.

EM Emergency state. This implies that the obstacle is so close and/or behaves unpredictably, such that special considerations must be made.

A geometrical interpretation combined with CPA and a CPA-like measure named the critical point, defines transitions between the states. An obstacle in a state other than the “safe” state has to transition to the “safe” state before it can transition to another state. This increases the stability of the COLREGs interpreter, and avoids switching between different COLREGs rules.

The topic of predicting future obstacle trajectories have also received some effort. The common approach to this problem is to use a constant velocity model (CVM), assuming that an obstacle continues with the current speed and course, resulting in a straight-line motion prediction. Being a simple and intuitive method, this is an attractive method widely used in the COLAV literature, including all the COLAV algorithms presented in this thesis. However, for prediction horizons in the range of minutes or longer, the uncertainty related to such a prediction, originating from measurement noise and the possibility of obstacles maneuvering, grows to useless levels. This problem is commonly mitigated by neglecting the uncertainty, relying solely on the expectation. Although proven in many applications, this approach removes relevant information. To improve on this, we have investigated AIS-based prediction of future obstacle trajectories exploiting the patterns encoded in a historic AIS dataset including approximately 3 million AIS messages in the vicinity of the Trondheimsfjord. We initialized this work in Hexeberg, Flåten, Eriksen, and Brekke [27] by analyzing the AIS dataset and devising the single point neighbor search (SPNS) method. The SPNS method performs well for predictions up to around 30 minutes, but is unable to include multimodality. The method was further developed in Hexeberg [92] and Dalsnes, Hexeberg, Flåten, Eriksen, and Brekke [29], resulting in the neighbor course distribution method (NCDM), accounting for multimodality by representing the obstacle trajectories as a particle-based probability density function (PDF). Dalsnes [93] fits a Gaussian mixture model to the PDF provided by the NCDM, obtaining an easily evaluable PDF for interfacing the prediction method with COLAV algorithms. A proof of concept, combining the prediction algorithm with the mid-level algorithm in Paper C, shows a potential for making more proactive maneuvers when using the AIS-based prediction method, compared to a CVM. This is a preliminary study, and more effort should be put into verifying and improving the interconnection of COLAV algorithms and the NCDM.

In our AIS-based obstacle trajectory prediction methods, we neglect the ownship’s influence on how other vessels will maneuver. This is an important factor in close-counter situations, and further research should investigate possibilities to relax this. One possible idea is to have a tight coupling between prediction and control, e.g. allowing an MPC algorithm to perform both the prediction and control tasks, coupling the obstacle trajectory prediction with the ownship's (planned) maneuvers.

Chapter 4

CONCLUSIONS AND FURTHER WORK

This thesis contains contributions in the topics of COLAV and motion control for ASVs. This twofold aim is due to this thesis' bottom-up approach to the problem of COLAV for ASVs, realizing that a COLAV system only performs as good as the vessel controller allows it to.

The work has been carried out focusing on high-speed ASVs, often operating in the displacement, semi-displacement and planing regions due to their typically small lengths. The contributions on motion control of ASVs is based on a developed method for modeling and identification of high-speed ASVs, which is experimentally proven to provide a control-oriented model valid for the displacement, semi-displacement and planing regions. For tracking time-varying references, utilizing model-based feedforward terms have large benefits over traditional feedback controllers, which are insufficient for such a task. The modeling and identification method is applied to obtain a model of a high-speed ASV. The identified model is then used to design two vessel controllers: one controlling speed and yaw rate and one controlling speed and course. Both controllers are shown to outperform other controllers in experiments, including a traditional feedback controller and a feedback-linearizing controller. The developed vessel controllers are later used in several full-scale COLAV experiments.

The proposed system has a so-called hybrid architecture, including a three-layered hybrid COLAV system building on complementary strengths of different algorithms, a number of support functions, and a vessel controller. The top COLAV layer performs energy-optimized path planning, finding a path throughout an environment of static obstacles while minimizing the energy consumption required to reach the goal. The middle COLAV layer avoids moving obstacles in a proactive manner, commanding predictable maneuvers which communicate intentions to other vessels. The bottom COLAV layer handles short-term COLAV, responding quickly to obstacles detected late or behaving dangerously, and handling situations where the mid-level algorithm fails to find a solution. The middle and bottom layers divide the COLREGs responsibilities between them. The middle layer should interpret the rules in a strict manner, enforcing the COLREGs rules including the stand-on requirement in crossing situations exactly. The bottom layer should then handle situations where the rules have to be interpreted loosely in order to stay safe, as exemplified by situations where obstacles do not follow the COLREGs in crossing situations.

The hybrid architecture is populated with three COLAV algorithms, a state machine for interpreting the COLREGs, and a model-based speed and course controller, and is shown to have good performance in simulations. The simulations include obstacles maneuvering in accordance with the COLREGs, ignoring the COLREGs rules, and also maneuvering dangerously. The hybrid COLAV system avoids collision in compliance with COLREGs rules 8 and 13–17 in all the situations, while following an energy-optimized trajectory when obstacles do not interfere with it.

Specifically, the top layer depends on an algorithm for energy-optimized path planning, developed in Bitar, Breivik, and Lekkas [91] and Bitar, Vestad, Lekkas, and Breivik [94]. The algorithm uses a model of a high-speed ASV, and produces an energy-optimized nominal trajectory from an initial to a goal position while avoiding static obstacles.

One algorithm for the middle layer is presented in this thesis. This algorithm, referred to as the mid-level algorithm, is based on MPC, formulated as an NLP to enable flexibility in designing the algorithm. The mid-level algorithm inputs the nominal trajectory from the high-level algorithm, following it using relative trajectory tracking while avoiding moving and static obstacles. The concept of relative trajectory tracking involves tracking the trajectory with a time offset. This time offset is actively controlled to avoid the mid-level algorithm to speed up in situations where it lags behind the desired position, but rather adjusting the time offset to track the closest portion of the nominal trajectory. Furthermore, the mid-level algorithm considers COLREGs rules 8, 13–16 and parts of Rule 17, which requires a stand-on vessel to not maneuver, leaving the responsibility in such a situation to the give-way vessel. To facilitate this, a state machine classifying obstacles with relevant COLREGs rules gives input to the algorithm. This COLREGs interpreter combines two risk-based measures with a geometrical interpretation of the situation to label obstacles as either “safe”, “overtaking”, “stand-on”, “give-way”, “head-on” or “emergency” situations. The labels are used to tailor the NLP to the current situation, in order to achieve the desired COLREGs behavior. The mid-level algorithm is currently only verified through simulations.

Furthermore, the thesis presents two algorithms for short-term COLAV. Both algorithms are tested in full-scale experiments, using a radar-based system developed in the Autosea project for detecting and tracking obstacles. The first algorithm is based on the DW algorithm, adapted for use with high-speed ASVs. This algorithm is shown to have robustness issues when used with the radar-based tracking system, which inherently provides obstacle estimates with a considerable amount of noise. In particular, noisy course estimates influence predictions of future obstacle trajectories heavily when employing the commonly used concept of assuming that obstacles keep a constant speed and course. To remedy the robustness issues, the BC-MPC algorithm has been developed with the design idea of being robust with respect to obstacle noise. This algorithm uses MPC and sample-based optimization techniques, and is demonstrated to be robust with respect to noisy obstacle estimates in several full-scale experiments using radar-based detection and tracking of obstacles. The BC-MPC algorithm originally only considered moving obstacles, but is extended to also consider static obstacles. With respect to the COLREGs, the algorithm considers rules 8, 13–15 and parts of Rule 17. More specifically, it handles parts of Rule 17 by maneuvering in situations where the ownship has a stand-on role, but must maneuver due to obstacles violating the COLREGs.

Although the BC-MPC algorithm is experimentally validated in full-scale experiments using a radar-based tracking system at multiple occasions, the complete hybrid COLAV system is only tested in simulations. This is an obvious point that should receive attention in the future, and can be tackled by first performing simulations including disturbances such as obstacle estimate noise. Furthermore, the COLAV system should be extended to also consider other parts of the COLREGs, including special responsibilities between vessels under sail, engaged in fishing or having a restricted ability to maneuver. The mid-level algorithm is well-suited for this extension. The simulations and experiments presented in this thesis represent situations where an ASV faces collision risks involving manned vessels. In a future where autonomous vessels become common, situations will occur where autonomous vessels must avoid collision with each other. This makes it relevant to also perform simulations with multiple ASVs facing each other, both using the same and different COLAV systems. None of the algorithms developed in this thesis are without weaknesses and room for improvement, and all the corresponding papers contain individual and more detailed suggestions for further work.

We are on the edge of developing autonomous surface vehicles, and a tide is coming in that will bring safer, more efficient maritime transportation to all.

Chapter 5

ORIGINAL PUBLICATIONS

Paper A A modified dynamic window algorithm for horizontal collision avoidance for AUVs

Postprint of **B.-O. H. Eriksen**, M. Breivik, K. Y. Pettersen, and M. S. Wiig, "A modified dynamic window algorithm for horizontal collision avoidance for AUVs", in *Proc. of the 2016 IEEE Conference on Control Applications (CCA)*, (Buenos Aires, Argentina), 2016, pp. 499–506.

A Modified Dynamic Window Algorithm for Horizontal Collision Avoidance for AUVs

Bjørn-Olav H. Eriksen, Morten Breivik, Kristin Y. Pettersen, Martin S. Wiig

***Abstract*— Much research has been done on the subject of collision avoidance (COLAV). However, few results are presented that consider vehicles with second-order nonholonomic constraints, such as autonomous underwater vehicles (AUVs). This paper considers the dynamic window (DW) algorithm for reactive horizontal COLAV for AUVs, and uses the HUGIN 1000 AUV in a case study. The DW algorithm is originally developed for vehicles with first-order nonholonomic constraints and is hence not directly applicable for AUVs without resulting in degraded performance. This paper suggests further developments of the DW algorithm to make it better suited for use with AUVs. In particular, a new method for predicting AUV trajectories using a linear approximation which accounts for second-order nonholonomic constraints is developed. The new prediction method, together with a modified search space, reduces the mean square prediction error to about one percent of the original algorithm. The performance and robustness of the modified DW algorithm is evaluated through simulations using a nonlinear model of the HUGIN 1000 AUV.**

I. Introduction

Collision avoidance (COLAV) systems are necessary for autonomous operation of vehicles, including autonomous underwater vehicles (AUVs). AUVs are often engaged in long term operations in deep waters with limited communication possibilities, which increase the reliability requirements of the COLAV system. It is clear that if a collision occurs and the AUV is immobilized, a salvage operation will be both costly and time consuming. In addition, a delayed operation may potentially have large economical consequences.

The topic of COLAV may be split in two main areas [1]:

- *Obstacle detection*, which focuses on detecting obstacles, usually based on sonar data regarding AUVs.
- *Obstacle avoidance*, which consists of generating appropriate steering commands in order to avoid collisions with detected obstacles.

A *COLAV system* must include both obstacle detection and avoidance in order to avoid collisions. This paper will only focus on obstacle avoidance. For details on how obstacle detection can be handled, see [2] and the references therein.

There exists a number of both *reactive* (local) and *deliberate* (global) COLAV algorithms. Reactive algorithms base their actions only on real-time sensor data, which makes the algorithms computationally inexpensive and suitable for reacting to sudden changes in the environment. Using no a priori information can, however, make the vehicle sensitive to local minima or traps, which can make it unable to reach the goal. In comparison, deliberate algorithms include a priori information to plan future actions. The provided actions are more likely to make the vehicle converge towards the goal, at the cost of increased computational time and reduced ability to react rapidly. Reactive and deliberate algorithms are often combined in *hybrid* architectures, where they are executed in parallel at different sampling frequencies [3], [4].

One of the existing reactive[1] algorithms is the dynamic window (DW) algorithm [5], which was originally intended for vehicles with first-order nonholonomic constraints and constant acceleration limits. AUVs, however, have second-order nonholonomic constraints, and also nonlinear responses which result in time-varying acceleration limits. Applying the original DW algorithm to AUVs will thus result in degraded performance. This paper therefore suggests a number of modifications to the original DW algorithm to increase the performance when applied to vehicles with second-order nonholonomic constraints and time-varying acceleration limits. In particular, a new trajectory prediction method taking second-order nonholonomic constraints into account is developed. This, together with a modified search space, reduces the mean square prediction error to about one percent of the original algorithm. The generality of the algorithm is also improved to facilitate a more layered architecture.

A comparison between the modified and original DW algorithm is obtained through simulations for various static obstacle environments using a nonlinear model of the HUGIN 1000 AUV [6], shown in Figure 1. The simulations show a significant performance improvement when applying the modified DW algorithm.

In Section II, a 3 degrees-of-freedom (DOF) control model is presented. Section III contains the DW algorithm and the proposed modifications. Simulation results are presented in Section IV, while Section V concludes the paper.

II. Control-oriented AUV model

In this section, a generic 3 DOF AUV control model is defined. The model is based on the following assumptions:

Assumption 1: The AUV model describes the motion of the *pivot point* of the vehicle.

Bjørn-Olav H. Eriksen, Morten Breivik and Kristin Y. Pettersen are with the Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway. Email: {bjorn-olav.h.eriksen, morten.breivik}@ieee.org and kristin.y.pettersen@itk.ntnu.no

Martin S. Wiig is with the Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, NTNU, NO-7491 Trondheim, Norway and the Norwegian Defence Research Establishment, NO-2027 Kjeller, Norway. Email: Martin-Syre.Wiig@ffi.no

[1]One may argue that the DW algorithm is not strictly reactive since it plans trajectories in time. It does, however, only rely on real-time sensor data, thus it is considered to be a reactive algorithm.

Fig. 1. The HUGIN 1000 AUV, courtesy of the Norwegian Defence Research Establishment.

Remark 1: When the body-fixed coordinate system is positioned in the pivot point, the rudders do not affect the sway dynamics directly. For vehicles which are controllable in yaw it is always possible to transform a model represented in an arbitrary point on the AUV to the pivot point [7].

Assumption 2: The vehicle is neutrally buoyant and the center of gravity (CG) is located below the center of buoyancy (CB) on a vertical line.

Assumption 3: Heave speed and the roll and pitch angles are assumed equal to zero.

Remark 2: Assuming zero roll angle is a common assumption for slender body vehicles such as AUVs [8]. The CG is located below the CB on a vertical line, which passively stabilizes the roll motion. Further, by equally utilizing the top and bottom rudders, and the port and starboard rudders, no roll moment is generated by the rudders. For an AUV equipped with a depth controller, close to zero heave speed and pitch angle is achieved when the AUV is traveling in a horizontal plane.

Assumption 3 allows a control model to be formulated in 3 DOF (surge, sway and yaw), while Assumption 2 implies that no restoring forces are applied. Using the SNAME [9] notation, the 3 DOF control model is therefore given as:

$$\dot{\boldsymbol{\eta}}^n_{b/n} = \boldsymbol{R}(\boldsymbol{\eta}^n_{b/n})\boldsymbol{\nu}^b_{b/n} \tag{1a}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}}^b_{b/n} + \boldsymbol{C}(\boldsymbol{\nu}^b_{b/n})\boldsymbol{\nu}^b_{b/n} + \boldsymbol{D}(\boldsymbol{\nu}^b_{b/n})\boldsymbol{\nu}^b_{b/n} = \boldsymbol{B}\boldsymbol{f}^b_b, \tag{1b}$$

where $\boldsymbol{\eta}^n_{b/n} = \begin{bmatrix} x & y & \psi \end{bmatrix}^T \in \mathbb{R}^2 \times SO(2)$ is the position and orientation of the body-fixed frame $\{b\}$ represented in the north-east-down-fixed inertial frame $\{n\}$. Further, $\boldsymbol{\nu}^b_{b/n} = \begin{bmatrix} u & v & r \end{bmatrix}^T \in \mathbb{R}^3$ is the velocity of $\{b\}$ with respect to $\{n\}$ represented in $\{b\}$, and $\boldsymbol{f}^b_b = \begin{bmatrix} X & N \end{bmatrix}^T \in \mathbb{R}^2$ is the actuator input in $\{b\}$. It should be noted that the model (1) does not account for external forces such as ocean currents, wind and waves. For notational simplicity, $\boldsymbol{\eta}^n_{b/n}, \boldsymbol{\nu}^b_{b/n}, \boldsymbol{f}^b_b$ are further denoted as $\boldsymbol{\eta}, \boldsymbol{\nu}, \boldsymbol{f}$. The transformation matrix from $\{b\}$ to $\{n\}$ is given as:

$$\boldsymbol{R}(\boldsymbol{\eta}) = \boldsymbol{R}(\psi) = \begin{bmatrix} c(\psi) & -s(\psi) & 0 \\ s(\psi) & c(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2a}$$

where $c(\cdot) = \cos(\cdot)$ and $s(\cdot) = \sin(\cdot)$. The vessel dynamics matrices are given as:

$$\boldsymbol{M} = \begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & m_{23} \\ 0 & m_{23} & m_{33} \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} b_{11} & 0 \\ 0 & b_{22} \\ 0 & b_{32} \end{bmatrix}$$
$$\boldsymbol{C}(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & -m_{22}v - m_{23}r \\ 0 & 0 & m_{11}u \\ m_{22}v + m_{23}r & -m_{11}u & 0 \end{bmatrix} \tag{2b}$$
$$\boldsymbol{D}(\boldsymbol{\nu}) = -\begin{bmatrix} X_u + X_{|u|u}|u| & 0 & 0 \\ 0 & Y_v & Y_r \\ 0 & N_v & N_r \end{bmatrix} \frac{|u'|}{u_0}.$$

Here, the term $\frac{|u'|}{u_0}$, where $u_0 > 0$ is the nominal surge speed, is used for speed-scaling of the damping coefficients. The term $|u'| \triangleq \max(|u|, \mu)$, where $\mu > 0$ is an arbitrary constant, guarantees some damping for low surge speeds. The constants in $\boldsymbol{B}$ are given as $b_{11} = b_{32} = 1$, while b_{22} captures the coupling from the yaw torque to the sway force in the actuator model, given from $Y = -\frac{1}{l_x}N \triangleq b_{22}N$. The force vector $\boldsymbol{f}$ is modeled as:

$$\boldsymbol{f} = \begin{bmatrix} X \\ N \end{bmatrix} = \begin{bmatrix} T_{|n|n}|n_p|n_p + T_{un}un_p \\ -Y_{\delta u^2} l_x \delta_\psi \end{bmatrix}, \tag{3}$$

where n_p and δ_ψ are the propeller speed and rudder deflection angle, respectively, while $T_{|n|n}$, T_{un} and $Y_{\delta u^2}$ are propeller and rudder coefficients, and l_x is the rudder lever arm along the x-axis. The actuators are limited by both saturation and rate limitations:

$$n_p \in [n_{p_{\min}}, n_{p_{\max}}], \; \|\delta_\psi\|_\infty \leq \delta_{\max}, \; \left\|\dot{\delta}_\psi\right\|_\infty \leq \dot{\delta}_{\max}. \tag{4}$$

The actuator limitations are not considered in the mathematical model (1) since $\boldsymbol{f}$ is selected as the control input, but are, however, to be included in the DW algorithm. For positive surge speeds, and requiring that the DW algorithm only specifies feasible commands, it is always possible to calculate n_p and δ_ψ given a force vector $\boldsymbol{f}$. Hence, a unique inverse function $\boldsymbol{f}^{-1}$ exists.

It should be noted that when Assumption 1 is satisfied, the following property holds [7]:

$$\boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{f} = \begin{bmatrix} \frac{b_{11}}{m_{11}}X \\ 0 \\ \frac{m_{22}b_{32} - m_{23}b_{22}}{m_{22}m_{33} - m_{23}^2}N \end{bmatrix}. \tag{5}$$

III. The Dynamic Window Algorithm

A. Introduction

The DW algorithm is a velocity space method, intended to prohibit infeasible control commands by specifying a desired *velocity pair* consisting of a desired forward speed and a desired rotational rate as reference signals for the vehicle speed controllers. The algorithm was originally designed for a car-like mobile robot with first-order nonholonomic constraints, moving in 3 DOF [5]. The original paper predicts vehicle trajectories as circular arcs with radii of $M_R^i = \frac{u_i}{r_i}$ (and straight lines for $r_i = 0$) for a discrete set of desired velocity pairs (u_i, r_i). The original trajectory prediction is

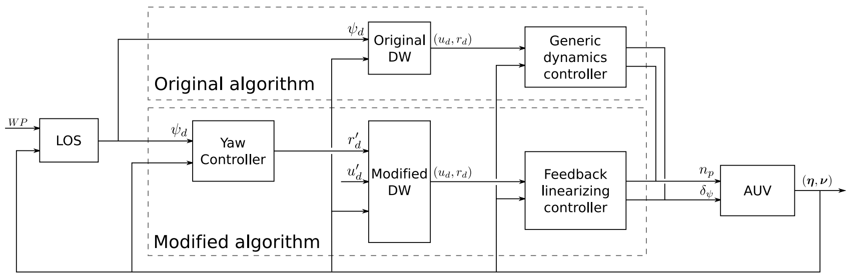


Fig. 2. Architecture overview. The modified algorithm facilitates a more modular architecture than the original algorithm.

quite accurate for vehicles with only first-order nonholonomic constraints, since they have no sideways motion.

As seen in Figure 2, the original DW algorithm inputs a desired heading ψ_d to guide the vehicle towards a goal. In contrast, the modified algorithm inputs a desired surge speed and yaw rate (u'_d, r'_d), which facilitates a more modular architecture in addition to allowing the surge speed to be externally guided. Notice that (u'_d, r'_d) is a desired velocity pair used as an *input* to the modified algorithm, while (u_d, r_d) is the *output* of the algorithm used as the reference for the vehicle dynamics controller.

Here, the DW algorithm is combined with a line-of-sight (LOS) guidance law, given as [8]:

$$\psi_{LOS} = \alpha_p - \arctan\left(\frac{e}{\Delta}\right), \tag{6}$$

where α_p is the path heading and e is the cross track error. The tuning parameter $\Delta > 0$ is the lookahead-distance, given in meters. To generate a desired yaw rate for the modified DW algorithm, the following yaw controller is proposed:

$$r'_d = -k_\psi\left(\psi - \psi_d\right) + \dot{\psi}_d, \tag{7}$$

where $k_\psi > 0$ is a constant gain and $\psi_d = \psi_{LOS}$.

For an implementation of the DW algorithm with an integral line-of-sight (ILOS) guidance law for compensation of ocean currents, also including a proof of convergence to a straight line path for the combined system, see [2]. It should be noted that the case presented in this paper is a special case of the case considered in [2], hence the convergence proof also applies to the system presented here.

B. The original dynamic window algorithm

Three 2D search spaces in forward (surge) speed and rotational (yaw) rate accounts for the kinematic and kinetic constraints of the vehicle. The *dynamic window* allows a time interval T (usually smaller than the sample time) for acceleration of the vehicle:

$$V_d = \Big\{(u,r) \in \mathbb{R}\times\mathbb{R}\Big| u \in [u^* + \dot{u}_{\min}T, u^* + \dot{u}_{\max}T] \\ \wedge\, r \in [r^* - \dot{r}_{\max}T, r^* + \dot{r}_{\max}T]\Big\}, \tag{8}$$

where u^*, r^* are the current forward speed and rotational rate and $\dot{u}_{\min} < 0, \dot{u}_{\max} > 0, \dot{r}_{\max} > 0$ are the vehicle accelerations limits. Note that the original algorithm assumes that the yaw acceleration limits are symmetric, hence $\dot{r}_{\min} = -\dot{r}_{\max}$. The set of *possible velocities* is:

$$V_s = \Big\{(u,r) \in \mathbb{R}\times\mathbb{R}\Big| u \in [0, u_{\max}] \\ \wedge\, r \in [-r_{\max}, r_{\max}]\Big\}, \tag{9}$$

where $u_{\max}$ and $r_{\max}$ are the maximum forward speed and rotational rate. The dynamic window and the set of possible velocities account for the actuator limitations. Finally, the set of *admissible velocities* ensures that the vehicle is able to stop before it collides with an obstacle:

$$V_a = \Big\{(u,r) \in \mathbb{R}\times\mathbb{R}\Big| u \leq \sqrt{2\cdot \mathrm{dist}(u,r)\cdot|\dot{u}_{\min}|} \\ \wedge\, |r| \leq \sqrt{2\cdot \mathrm{dist}(u,r)\cdot \dot{r}_{\max}}\Big\}, \tag{10}$$

where $\mathrm{dist}(u,r)$ expresses the distance which the vehicle can travel along the trajectory given the velocity pair (u,r) without colliding with an obstacle.

The optimal velocity pair is selected through maximizing an objective function over the resulting search space $V_r = V_d \cap V_s \cap V_a$:

$$\max_{(u,r)} G(u,r) = \sigma\left(\alpha\cdot\mathrm{heading}(u,r) + \beta\cdot\mathrm{dist}(u,r) \\ +\gamma\cdot\mathrm{velocity}(u,r)\right) \tag{11}$$
$$\text{s.t. } (u,r) \in V_r,$$

where $\alpha, \beta, \gamma > 0$ are tuning parameters and $\mathrm{heading}(u,r)$ measures the alignment between the trajectory corresponding to the velocity pair (u,r) and a desired heading. The term $\mathrm{velocity}(u,r)$ favors holding a high surge speed, while σ is a low-pass filter to reduce fluctuations in the control output. The optimization problem (11) is solved by numerically computing the objective value for all $(u,r) \in V_r$ (discretized) and selecting the one with the highest objective value.

The original algorithm has two significant limitations when applied to AUVs:

- The lack of modeling the sideways motion of vehicles with second-order nonholonomic constraints results in inaccurate trajectory predictions.
- Using a rectangular search space which does not include any actuator modeling can cause infeasible control references to be specified.

Furthermore, the *heading* term in (11) compares a desired heading input with a resulting heading for the trajectories, requiring the algorithm to include a mapping function to compute the resulting heading for the different velocity pairs.

Several modifications to the original DW algorithm have been proposed. Some of these include global information in order to handle local minima in the environment, see [10], [11], [12], [13]. In particular, [13] combines the focussed D* (FD*) [14] global planner with the DW algorithm in a hybrid architecture. The *heading* and *velocity* terms are replaced by a measure of alignment to the FD* path, hence also removing the required mapping from angle to angular rate. To reduce the prediction error of the vehicle trajectories, [15] uses clothoid curves instead of circular arcs and straight lines to model the vehicle trajectories. A simplified set of equations is used in [4] to simulate trajectories for an autonomous surface vehicle (ASV), accounting for an estimated lateral speed.

C. A new search space and objective function

In this section we present a modified search space to better suit the nonlinear responses of AUVs. The objective function is also changed to facilitate a more layered architecture.

To ensure feasible steering commands for the AUV, the search space is modified to account for the actuator model (3) and limitations (4). Allowing a small time interval $T_a < T$ to be used for changing the control inputs, the feasible actuator commands are:

$$\begin{aligned} &\delta_\psi \in \text{sat}\left(\left[\delta_\psi^* - T_a\dot{\delta}_{\max}, \delta_\psi^* + T_a\dot{\delta}_{\max}\right], \delta_{\max}\right) \\ &n_p \in \left[n_{p_{\min}}, n_{p_{\max}}\right], \end{aligned} \tag{12}$$

where δ_ψ^* denotes the current rudder deflection angle, and $\text{sat}(\cdot)$ is a saturation function. It should be noted in general that rate limitation can also be imposed on the propeller speed. By denoting $\boldsymbol{Bf} = \boldsymbol{\tau}(\boldsymbol{\nu}, \delta_\psi, n_p)$, and since the actuators are linearly independent, the acceleration limits can be found by solving:

$$\dot{\boldsymbol{\nu}}_i = \boldsymbol{M}^{-1}\left(\boldsymbol{\tau}_i - \boldsymbol{C}(\boldsymbol{\nu}^*)\boldsymbol{\nu}^* - \boldsymbol{D}(\boldsymbol{\nu}^*)\boldsymbol{\nu}^*\right), \tag{13}$$

where $i \in \{\min, \max\}$, $\boldsymbol{\tau}_{\min} \triangleq \boldsymbol{\tau}(\boldsymbol{\nu}^*, \max(\delta_\psi), \min(n_p))$, $\boldsymbol{\tau}_{\max} \triangleq \boldsymbol{\tau}(\boldsymbol{\nu}^*, \min(\delta_\psi), \max(n_p))$ and $\boldsymbol{\nu}^*$ denotes the current vehicle velocity. It is worth noticing that a positive rudder deflection results in negative yaw moment. The dynamic window (8) is then modified as:

$$\begin{aligned} V_d = \Big\{(u, r) \in \mathbb{R} \times \mathbb{R} \Big| u \in [u^* + \dot{u}_{\min}T, u^* + \dot{u}_{\max}T] \\ \wedge r \in [r^* + \dot{r}_{\min}T, r^* + \dot{r}_{\max}T]\Big\}. \end{aligned} \tag{14}$$

Notice that in contrast to the original algorithm the yaw rate acceleration limit is no longer assumed to be symmetric.

By defining a function $g(u, r)$ which is positive semidefinite for feasible velocities with respect to actuator saturations, the set of possible velocities (9) can be generalized as:

$$V_s = \Big\{(u, r) \in \mathbb{R} \times \mathbb{R} \Big| g(u, r) \geq 0\Big\}. \tag{15}$$

The function $g(u, r)$ is approximated by numerically calculating the boundaries of the possible steady state solutions of the kinetics (1b), see [2] for details. The sets V_d, V_s and the dynamically feasible velocity set $V_f = V_d \cap V_s$ are illustrated in Figure 3.

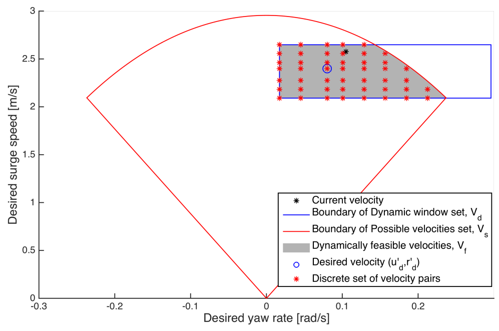


Fig. 3. The dynamically feasible velocity set V_f (in grey), together with the boundaries of the dynamic velocity window (in blue) and the possible velocity set (in red). The dynamically feasible velocity set is discretized uniformly, and note that the desired velocity pair (u'_d, r'_d) is included in the discrete search space.

To simplify the implementation, the configuration space of the AUV is reduced from $\mathbb{R}^2 \times SO(2)$ to $\mathbb{R}^2$ by approximating the AUV footprint as a circle. This is done by expanding the detected obstacles with the maximum AUV radius, and representing the AUV as a particle. Inspired by [16], two regions are defined on the detected obstacles to control the obstacle clearance; the "avoidance region"

$$\Omega \triangleq \Big\{\boldsymbol{p} \in \mathcal{C} \Big| \, \|\boldsymbol{p} - \boldsymbol{p}_{obs}\|_2 \leq \bar{r}\Big\}, \tag{16}$$

and the "antitarget region"

$$\mathcal{T} \triangleq \Big\{\boldsymbol{p} \in \mathcal{C} \Big| \, \|\boldsymbol{p} - \boldsymbol{p}_{obs}\|_2 \leq r^*\Big\} \tag{17}$$

where $\boldsymbol{p}_{obs} \in \mathbb{R}^2$ is the position of obstacles, $\mathcal{C} = \mathbb{R}^2$ is a collapsed, heading independent configuration space and $\bar{r} > r^* > 0$ are scalars defining the size of Ω and $\mathcal{T}$. In particular, r^* is the maximum radius of the AUV corresponding to approximating the AUV footprint as a circle. The antitarget region is interpreted as the region where a collision may occur, while the avoidance region is interpreted as a safety region that is not desirable to enter. The regions are illustrated in Figure 4.

The set of admissible velocities (10) is modified as:

$$\begin{aligned} V_a = \Big\{(u, r) \in \mathbb{R} \times \mathbb{R} \Big| u \leq \sqrt{2\rho'(u, r)|\dot{u}_{\min}|} \\ \wedge |r| \leq \left\{ \begin{array}{ll} \sqrt{2\rho'(u, r)|\dot{r}_{\max}|} & , r < 0 \\ \sqrt{2\rho'(u, r)|\dot{r}_{\min}|} & , r \geq 0 \end{array} \right\}. \end{aligned} \tag{18}$$

The function $\rho'(u, r)$ expresses the remaining distance the AUV can travel along the resulting trajectory at the next iteration without entering the antitarget region $\mathcal{T}$:

$$\rho'(u, r) = \max(\rho(u, r) - \Delta_s, 0), \tag{19}$$

where $\rho(u, r)$ expresses the distance the AUV can travel along the resulting trajectory before it enters $\mathcal{T}$ and Δ_s expresses the distance the AUV travels until the next iteration.

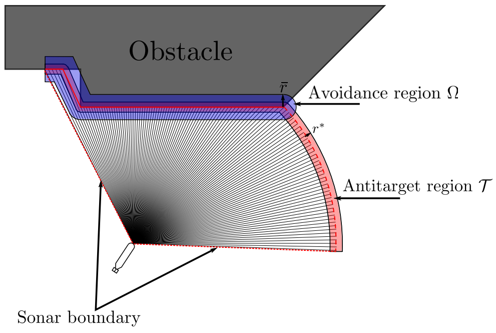


Fig. 4. Obstacle regions. Notice that the antitarget region is extended along the forward sonar boundary to account for possible obstacles just outside of the sonar range. Black lines illustrate sonar range measurements.

To improve the generality of the algorithm and remove the need for computing resulting headings for the velocity pairs, the *heading* term of the objective function is replaced with a term taking a desired yaw rate as input. This is inspired by [17] and [18]. In addition, inspired by [13], the $dist(u,r)$ term is scaled by the trajectory velocity, resulting in a term which approximates the time until collision. To motivate the algorithm to keep out of the avoidance region, $\text{dist}(u,r)$ expresses the time until the AUV enters Ω. The algorithm input is assumed to be smooth, so the low-pass filter is omitted. The objective function in (11) is thus modified as:

$$G(u,r) = \alpha\cdot\text{yawrate}(u,r,r'_d) + \beta\cdot\text{dist}(u,r) + \gamma\cdot\text{velocity}(u,r,u'_d), \quad (20)$$

where u'_d and r'_d are inputs to the algorithm, and

$$\text{yawrate}(u,r,r'_d) = 1 - \frac{|r'_d - r|}{\max_{r\in V_r}(|r'_d - r|)}, \quad (21)$$

$$\text{velocity}(u,r,u'_d) = 1 - \frac{|u'_d - u|}{\max_{u\in V_r}(|u'_d - u|)}, \quad (22)$$

$$\text{dist}(u,r) = \frac{\bar{\rho}(u,r)}{\frac{1}{T}\int_0^T \|\boldsymbol{\chi}(u,r,t)\|_2\,\mathrm{d}t}, \quad (23)$$

where $\bar{\rho}(u,r)$ is the distance the AUV can travel along the trajectory specified by the velocity pair (u,r) until it enters Ω, and $\boldsymbol{\chi}(u,r,t)$ is the predicted AUV surge and sway speed along the trajectory specified by the velocity pair (u,r).

D. A new trajectory prediction method

To account for second-order nonholonomic constraints in the trajectory prediction, we propose to use partial feedback linearization to linearize the surge and yaw dynamics, while leaving the sway motion uncontrolled. The closed loop dynamics are then derived and used for predicting the AUV trajectories, hence including both sway and controller dynamics in the AUV trajectory prediction. This approach is similar to the one presented in [19], but does not require a linear model and is hence more flexible. In contrast to the approach suggested by [4], the actual equations of motion are used and the kinetics are solved analytically. This makes the prediction more accurate and requires less computations.

By solving (1b) for $\dot{\boldsymbol{\nu}}$, the system can be described as:

$$\dot{\boldsymbol{\nu}} = \boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{f} - \boldsymbol{n}(\boldsymbol{\nu}), \quad (24)$$

where $\boldsymbol{n}(\boldsymbol{\nu}) = \boldsymbol{M}^{-1}\left(\boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu}\right)$.

To formulate the control law, the system is divided into two parts. This is done by using the matrices $\boldsymbol{\Gamma}_1$ and $\boldsymbol{\Gamma}_2$:

$$\boldsymbol{\Gamma}_1 \triangleq \begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 1\end{bmatrix}, \quad \boldsymbol{\Gamma}_2 \triangleq \begin{bmatrix}0 & 1 & 0\end{bmatrix}, \quad (25)$$

that satisfies $\boldsymbol{\Gamma}_1^T\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2 = \mathbf{I}$. The system (24) can be written as:

$$\begin{aligned}\dot{\boldsymbol{\nu}} &= \left(\boldsymbol{\Gamma}_1^T\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\right)\left(\boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{f} - \boldsymbol{n}(\boldsymbol{\nu})\right)\\ &= \boldsymbol{\Gamma}_1^T\left(\boldsymbol{\Gamma}_1\boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{f} - \boldsymbol{\Gamma}_1\boldsymbol{n}(\boldsymbol{\nu})\right) - \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{n}(\boldsymbol{\nu}).\end{aligned} \quad (26)$$

Notice that $\boldsymbol{\Gamma}_2\boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{f} = 0$. Hence, the system is divided in two parts where $\boldsymbol{\Gamma}_1^T\boldsymbol{\Gamma}_1$ maps dynamics to surge and yaw, while $\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2$ maps dynamics to sway. To remove the nonlinearities in surge and yaw, we select the feedback linearizing control law:

$$\boldsymbol{f} = \left(\boldsymbol{\Gamma}_1\boldsymbol{M}^{-1}\boldsymbol{B}\right)^{-1}\left(\boldsymbol{\Gamma}_1\boldsymbol{n}(\boldsymbol{\nu}) + \boldsymbol{a}_{1d}\right), \quad (27)$$

where $\boldsymbol{a}_{1d} = \begin{bmatrix}\dot{u}_d & \dot{r}_d\end{bmatrix}^T$ is the desired acceleration.

Remark 3: By construction, it is shown in [2] that $\boldsymbol{\Gamma}_1\boldsymbol{M}^{-1}\boldsymbol{B}$ is of full rank and hence invertible.

Inserting (27) into (26) gives:

$$\dot{\boldsymbol{\nu}} = \boldsymbol{\Gamma}_1^T\boldsymbol{a}_{1d} - \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{n}(\boldsymbol{\nu}). \quad (28)$$

The desired acceleration is selected using a proportional controller with a reference feedforward:

$$\begin{aligned}\boldsymbol{a}_{1d} &= \dot{\boldsymbol{\nu}}_{1d} - \boldsymbol{K}_p\left(\boldsymbol{\nu}_1 - \boldsymbol{\nu}_{1d}\right)\\ &= \dot{\boldsymbol{\nu}}_{1d} - \boldsymbol{K}_p\left(\boldsymbol{\Gamma}_1\boldsymbol{\nu} - \boldsymbol{\nu}_{1d}\right),\end{aligned} \quad (29)$$

where $\boldsymbol{K}_p = \begin{bmatrix}k_u & 0\\ 0 & k_r\end{bmatrix} > 0$ is a gain matrix, $\boldsymbol{\nu}_1 = \begin{bmatrix}u & r\end{bmatrix}^T$ and $\boldsymbol{\nu}_{1d} = \begin{bmatrix}u_d & r_d\end{bmatrix}^T$.

Inserting (29) into (28), and defining

$$\tilde{\boldsymbol{\nu}} = \begin{bmatrix}\tilde{u}\\ v\\ \tilde{r}\end{bmatrix} \triangleq \boldsymbol{\nu} - \boldsymbol{\Gamma}_1^T\boldsymbol{\nu}_{1d}, \quad (30)$$

results in the dynamics:

$$\dot{\tilde{\boldsymbol{\nu}}} = -\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1\tilde{\boldsymbol{\nu}} - \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{n}(\boldsymbol{\nu}). \quad (31)$$

Note that $\boldsymbol{n}(\boldsymbol{\nu})$ takes $\boldsymbol{\nu}$ as its argument. Also, it is important to notice that (31) is linear in both surge and yaw:

$$\dot{\tilde{u}} = -k_u\tilde{u}, \quad \dot{\tilde{r}} = -k_r\tilde{r}, \quad \dot{v} = -n_2(\boldsymbol{\nu}), \quad (32)$$

where $n_2(\boldsymbol{\nu})$ is the contribution from the Coriolis-centripetal and damping matrices in sway, given as:

$$\begin{aligned}n_2(\boldsymbol{\nu}) = \frac{1}{m_{22}m_{33} - m_{23}^2}\Big(&(m_{33}d_{22} - m_{23}d_{32})\,v\\ &- m_{23}\,(m_{22} - m_{11})\,uv + \left(m_{33}m_{11} - m_{23}^2\right)ur\\ &+ (m_{33}d_{23} - m_{23}d_{33})\,r\Big),\end{aligned} \quad (33)$$

where $m_{ij} = M_{i,j}$ and $d_{ij} = D_{i,j}$.

By approximating the last term of (31) using a first-order Taylor series, the dynamics of the AUV is also linear in sway. The Taylor approximation is given as:

$$\begin{aligned} \boldsymbol{n}(\boldsymbol{\nu}) &\approx \boldsymbol{n}(\boldsymbol{\nu}^*) + \underbrace{\left.\frac{\mathrm{d}\boldsymbol{n}(\boldsymbol{\nu})}{\mathrm{d}\boldsymbol{\nu}}\right|_{\boldsymbol{\nu}=\boldsymbol{\nu}^*}}_{\boldsymbol{N}} (\boldsymbol{\nu} - \boldsymbol{\nu}^*) \\ &= \boldsymbol{n}(\boldsymbol{\nu}^*) + \boldsymbol{N}\boldsymbol{\nu} - \boldsymbol{N}\boldsymbol{\nu}^* \\ &= \boldsymbol{N}\boldsymbol{\nu} + \boldsymbol{b}(\boldsymbol{\nu}^*), \end{aligned} \tag{34}$$

where the current velocity $\boldsymbol{\nu}^*$ is selected as the linearization point and $\boldsymbol{b}(\boldsymbol{\nu}^*) = \boldsymbol{n}(\boldsymbol{\nu}^*) - \boldsymbol{N}\boldsymbol{\nu}^*$ is a constant term. The matrix $\boldsymbol{N}$ is stated in [2].

Inserting (34) into (31) yields:

$$\dot{\tilde{\boldsymbol{\nu}}} = \boldsymbol{A}\tilde{\boldsymbol{\nu}} + \boldsymbol{\beta}\boldsymbol{\nu}_{1d} + \boldsymbol{G}, \tag{35}$$

where:

$$\begin{aligned} \boldsymbol{A} &= -\left(\boldsymbol{\Gamma}_1^T \boldsymbol{K}_p \boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T \boldsymbol{\Gamma}_2 \boldsymbol{N}\right) \\ \boldsymbol{\beta} &= -\boldsymbol{\Gamma}_2^T \boldsymbol{\Gamma}_2 \boldsymbol{N} \boldsymbol{\Gamma}_1^T \\ \boldsymbol{G} &= -\boldsymbol{\Gamma}_2^T \boldsymbol{\Gamma}_2 \boldsymbol{b}(\boldsymbol{\nu}^*). \end{aligned} \tag{36}$$

This is a linear time-invariant system perturbed by a nonvanishing perturbation $\boldsymbol{G}$ (hence $\tilde{\boldsymbol{\nu}} = \boldsymbol{0}$ and $\boldsymbol{\nu}_{1d} \equiv \boldsymbol{0}$ does not imply $\dot{\tilde{\boldsymbol{\nu}}} = \boldsymbol{0}$). The time evolution of (35) is:

$$\begin{aligned} \tilde{\boldsymbol{\nu}}(t) &= \mathrm{e}^{\boldsymbol{A}(t-t_0)}\tilde{\boldsymbol{\nu}}(t_0) \\ &\quad + \int_{t_0}^{t} \mathrm{e}^{\boldsymbol{A}(t-\sigma)} \left(\boldsymbol{\beta}\boldsymbol{\nu}_{1d}(\sigma) + \boldsymbol{G}\right) \mathrm{d}\sigma. \end{aligned} \tag{37}$$

The desired surge speed and yaw rate are considered constant for each trajectory, hence $\boldsymbol{\nu}_{1d}$ is constant for each trajectory. By letting $t_0 = 0$ s, (37) can be expressed as [20]:

$$\tilde{\boldsymbol{\nu}}(t) = \mathrm{e}^{\boldsymbol{A}t}\tilde{\boldsymbol{\nu}}(0) - \boldsymbol{A}^{-1}\left(\boldsymbol{I} - \mathrm{e}^{\boldsymbol{A}t}\right)\left(\boldsymbol{\beta}\boldsymbol{\nu}_{1d} + \boldsymbol{G}\right). \tag{38}$$

The kinematics (1a) are simulated in discrete time using the modified Euler method [21]:

$$\begin{aligned} \boldsymbol{\eta}(t_{n+1}) &= \boldsymbol{\eta}(t_n) + h\boldsymbol{k}_2 \\ \boldsymbol{k}_1 &= \boldsymbol{R}(\boldsymbol{\eta}(t_n))\boldsymbol{\nu}(t_n) \\ \boldsymbol{k}_2 &= \boldsymbol{R}(\boldsymbol{\eta}(t_n) + \frac{h}{2}\boldsymbol{k}_1)\boldsymbol{\nu}(t_n + \frac{h}{2}), \end{aligned} \tag{39}$$

where h is the integration time step, and $\boldsymbol{\nu}(t)$ is computed from (38) and (30). It should be noted that the absolute position is not required in the DW implementation, as the sonar measurements are given in $\{b\}$. Hence, the AUV prediction can be done in $\{b\}$ by selecting $\boldsymbol{\eta} = \begin{bmatrix}0 & 0 & \psi\end{bmatrix}^T$.

IV. Simulation results

A number of simulations have been conducted to compare the modified DW algorithm with the original DW algorithm, in order to evaluate the performance of the two algorithms. A 6 DOF nonlinear model of the HUGIN 1000 AUV with a horizontally oriented forward looking sonar, developed by the Norwegian Defence Research Establishment and implemented in SIMULINK, has been used for testing the algorithms. The HUGIN 1000 AUV model satisfies Assumptions 1-3 given in Section II.

TABLE I
SIMULATION PARAMETERS:

Parameter	Value	Description
k_u	$1\ \mathrm{s}^{-1}$	Surge controller gain
k_r	$1\ \mathrm{s}^{-1}$	Yaw rate controller gain
k_ψ	$0.2\ \mathrm{s}^{-1}$	Yaw controller gain
Δ	8 m	LOS lookahead distance
α	1	DW yaw rate scaling constant
β	$9\ \mathrm{s}^{-1}$	DW distance scaling constant
γ	3	DW surge speed scaling constant
u'_d	$2\ \mathrm{m\,s}^{-1}$	Desired surge speed
ΔT_{DW}	1 s	Dynamic window algorithm sampling time
$\bar{r}$	6 m	Size of the avoidance region Ω
r^*	3.5 m	Size of the antitarget region $\mathcal{T}$

The control system was implemented in MATLAB, with parameters as in Table I. The model parameters are not stated due to confidentiality reasons. Further details about the simulator are given in [2].

To illustrate the improvement of predicting the AUV trajectories using the proposed linear approximation compared to the original approach, a set of AUV trajectories are predicted using a search space consisting of three desired surge velocities and three desired yaw rates. Assuming that V_s does not impose any limitations on the search space, this results in nine velocity pairs. The initial velocity is chosen as $\boldsymbol{\nu}(0) = \begin{bmatrix}2 & 0 & 0\end{bmatrix}^T$ and the trajectories are predicted for 30 s. Figure 5 shows the actual AUV trajectories together with predicted trajectories using both the original prediction and the new linear approximation, for three of the velocity pairs (the other six trajectories look similar, see [2]). Table II shows the average prediction error using both the original prediction and the new linear approximation. From Table II and Figure 5 it is clear that the linear approximation is much

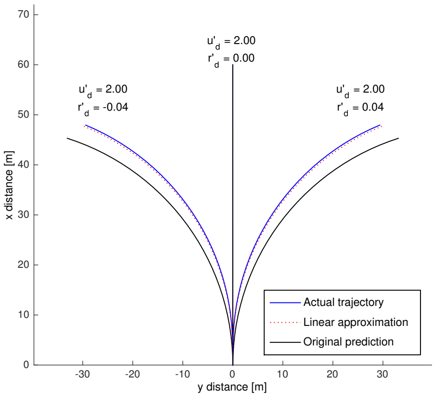


Fig. 5. Actual and approximated AUV trajectories, given initial velocity $\boldsymbol{\nu}(0) = [2 \quad 0 \quad 0]^T$ and $u'_d = 2\ \mathrm{m\,s}^{-1}$.

TABLE II

MEAN SQUARE ERROR OF PREDICTED AUV TRAJECTORIES:

Time span	Original prediction	Linear approximation	Linear approximation vs. original prediction
$t = [0, 5]$ s	0.100 m^2	0.000576 m^2	0.576 %
$t = [0, 30]$ s	4.67 m^2	0.045 m^2	0.964 %

more accurate than the original prediction method, especially for the initial part of the trajectories.

Simulations in various environments all indicates that the modified algorithm achieves a more consistent and secure clearance to obstacles. One of the comparisons is shown in Figure 6. Following the straight line path between the three waypoints would result in a collision, and an alternative path is therefore found by the DW algorithm. In this case, the modified algorithm chooses a shorter route than the original algorithm. The modified algorithm achieves a larger obstacle clearance, and makes the AUV stay well clear of the antitarget region at all times. In contrast, the original algorithm makes the AUV enter the antitarget region during the simulation. See Figure 7 and Table III for details.

The robustness of the COLAV system is assessed through a Monte Carlo simulation with 1500 samples. The AUV COLAV system is simulated in environments generated by filtering and thresholding matrices of normal distributed random elements, to represent the environments as obstacle grids. Further, based on simulated sonar ranges, estimates of $\mathcal{T}$ and Ω are generated as shown in Figure 4. As shown in Table IV, 17.2 % of the simulations of the modified algorithm came closer than 3 m to an obstacle, and hence may have caused a collision (recall that the AUV is represented as a particle, and the radius of the AUV is approximately 3 m). Some trajectories even resulted in a minimum distance of 0.1 m, surely causing a collision. A closer inspection of the trajectories reveals that when the AUV reduces the surge speed to avoid collisions in local minima, the speed scaling of the damping causes the AUV to slide sideways for a long time after the surge speed reaches zero. This is considered to be a simulation artifact, since the model (1) does not capture the correct vehicle dynamics at low surge speeds. For further elaboration of the simulation artifact, see [2]. Notice, however, that 71.3 % of the simulations of the original algorithm came closer than 3 m, demonstrating a large improvement with regards to the original algorithm. From Table IV and Figure 8 it is clear that the modified DW algorithm consistently achieves a larger obstacle clearance. An interesting result is that the modified DW algorithm made the AUV reach the final waypoint only in 31.9 % of the simulations, hence the AUV got trapped in local minima in 68.1 % of the simulations. This demonstrates the need for adapting the global path underway for example by using a deliberate planner together with the DW algorithm in a hybrid architecture, to avoid local minima. However, the modified DW algorithm performed better than the original DW algorithm which only reached the final WP in 28 % of the simulations.

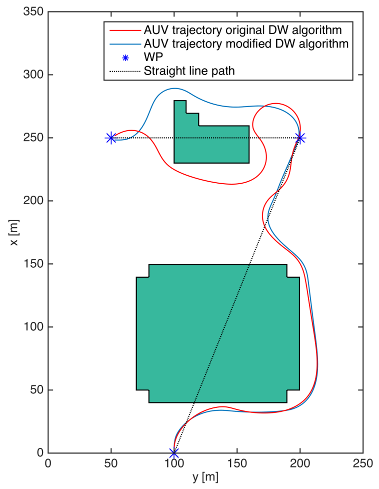


Fig. 6. AUV trajectories using the modified and original DW algorithms. The AUV starts at $(0, 100)$ m.

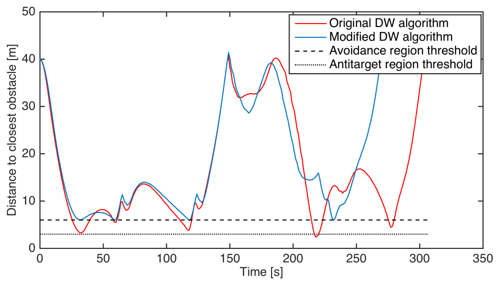


Fig. 7. Distance to closest obstacle.

V. CONCLUSION

We have in this paper proposed a number of modifications to the dynamic window (DW) algorithm to make it suitable for vehicles with second-order nonholonomic constraints and time-varying acceleration limitations.

Based on simulations, a new AUV trajectory prediction method accounting for second-order nonholonomic con-

TABLE III

TRAJECTORY DATA, ORIGINAL AND MODIFIED DW ALGORITHM:

Parameter	Original algorithm	Modified algorithm
Trajectory length to end WP	602 m	539 m
Trajectory time to end WP	307 s	273 s
Average surge speed	1.94 m s^{-1}	1.95 m s^{-1}
Minimum obstacle clearance	2.4 m	5.9 m

TABLE IV
SUMMARY OF MONTE CARLO SIMULATION:

Min. obs. clearance	Original algorithm		Modified algorithm	
	Perc. of simulations	Perc. of which reached goal	Perc. of simulations	Perc. of which reached goal
$[0,1]$ m	33.9 %	0.8 %	1.8 %	0 %
$(1,2]$ m	4.1 %	17.7 %	5.1 %	0 %
$(2,3]$ m	33.3 %	20.0 %	10.3 %	0 %
$(3,4]$ m	26.5 %	68.8 %	38.3 %	8.7 %
$(4,5]$ m	0.5 %	87.5 %	5.3 %	52.5 %
$(5,6]$ m	0.6 %	100.0 %	31.0 %	62.2 %
$(6,\infty)$ m	1.0 %	100.0 %	8.2 %	79.7 %
All	100.0 %	28.0 %	100.0 %	31.9 %

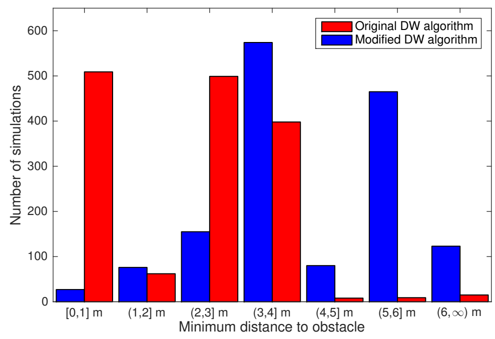


Fig. 8. Minimum obstacle clearance in the Monte Carlo simulation.

straints reduces the mean square prediction error to about one percent of the original method. Together with a modified search space, this improves the performance of the DW algorithm in terms of obstacle clearance and accuracy when applied to AUVs. Based on a Monte Carlo simulation of 1500 samples, the modified DW algorithm is believed to be robust with respect to obstacle configurations. This should however be investigated further. Due to the new prediction method, the computational cost of the modified algorithm is moderately larger than that of the original algorithm. On the other hand, the increased prediction accuracy makes it possible to run the DW algorithm at a lower sampling frequency.

In order to develop a practical COLAV system, further research will be put into combining the reactive DW algorithm with deliberate planning algorithms to ensure global convergence. The suitability of the DW algorithm for use with ASVs will also be evaluated.

ACKNOWLEDGMENT

This work was supported by the Research Council of Norway through project number 244116, and through the Centres of Excellence funding scheme with project number 223254.

REFERENCES

[1] C. S. Tan, R. Sutton, and J. Chudley, "Collision avoidance systems for autonomous underwater vehicles part A: A review of obstacle detection," *Journal of Marine Science and Environment*, vol. C2, pp. 39–50, 2004.
[2] B.-O. H. Eriksen, "Horizontal collision avoidance for autonomous underwater vehicles," Master's thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2015. [Online]. Available: http://hdl.handle.net/11250/2352502
[3] C. S. Tan, R. Sutton, and J. Chudley, "Collision avoidance systems for autonomous underwater vehicles part B: A review of obstacle avoidance," *Journal of Marine Science and Environment*, vol. C2, pp. 51–62, 2004.
[4] Ø. A. G. Loe, "Collision avoidance for unmanned surface vehicles," Master's thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2008. [Online]. Available: http://www.diva-portal.org/smash/record.jsf?pid=diva2:347606
[5] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.
[6] Kongsberg Maritime. (2014) Autonomous underwater vehicle - HUGIN. Accessed: 2014-10-06. [Online]. Available: http://www.km.kongsberg.com/ks/web/nokbg0240.nsf/AllWeb/B3F87A63D8E419E5C1256A68004E946C?OpenDocument
[7] W. Caharija, K. Y. Pettersen, J. T. Gravdahl, and E. Børhaug, "Integral LOS guidance for horizontal path following of underactuated autonomous underwater vehicles in the presence of vertical ocean currents," in *Proc. of American Control Conference*, Montréal, Canada, 2012, pp. 5427–5434.
[8] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons Ltd, 2011.
[9] SNAME, "Nomenclature for treating the motion of a submerged body through a fluid," The Society of Naval Architects and Marine Engineers, New York, USA, Tech. Rep., 1950.
[10] O. Brock and O. Khatib, "High-speed navigation using the global dynamic window approach," in *Proc. of IEEE International Conference on Robotics and Automation*, Detroit, Michigan, 1999.
[11] P. Ögren and N. E. Leonard, "A tractable convergent dynamic window approach to obstacle avoidance," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, 2002, pp. 595–600.
[12] I. Tusseyeva, S.-G. Kim, and Y.-G. Kim, "3D global dynamic window approach for navigation of autonomous underwater vehicles," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 13, pp. 91–99, 2013.
[13] M. Seder, K. Macek, and I. Petrovic, "An integrated approach to real-time mobile robot control in partially known indoor environments," in *Proc. of the 31st Annual Conference of IEEE Industrial Electronics Society*, Raleigh, North Carolina, USA, 2005, pp. 1785–1790.
[14] A. Stentz, "The focussed D* algorithm for real-time replanning," in *Proc. of the International Joint Conference on Artificial Intelligence*, Pittsburgh, Pennsylvania, USA, 1995.
[15] C. Schröter, M. Höchemer, and H.-M. Gross, "A particle filter for the dynamic window approach to mobile robot control," in *Proc. of the 52nd International Scientific Colloquium*, Ilmenau, Germany, 2007, pp. 425–430.
[16] E. J. Rodríguez-Seda, C. Tang, M. W. Spong, and D. M. Stipanovic, "Trajectory tracking with collision avoidance for nonholonomic vehicles with acceleration constraints and limited sensing," *The International Journal of Robotics Research*, vol. 33, pp. 1569–1592, 2014.
[17] H. Berti, A. Sappa, and O. Agamennoni, "Improved dynamic window approach by using Lyapunov stability criteria," *Latin American Applied Research*, vol. 38, no. 4, pp. 289–298, 2008.
[18] P. Inñigo-Blasco, F. Díaz-del Río, S. Vicente Díaz, and D. Cagigas Muñiz, "The shared control dynamic window approach for non-holonomic semi-autonomous robots," in *Proc. of the 45th International Symposium on Robotics*, Munich, Germany, 2014, pp. 355–360.
[19] D. Kiss and G. Tevesz, "Advanced dynamic window based navigation approach using model predictive control," in *Proc. of the 17th International Conference on Methods and Models in Automation and Robotics*, Miedzyzdroje, Poland, 2012, pp. 148–153.
[20] J. P. Hespanha, *Linear Systems Theory*. Princeton University Press, 2009.
[21] O. Egeland and J. T. Gravdahl, *Modeling and Simulation for Automatic Control*. Marine Cybernetics, 2003.

Paper B Modeling, identification and control of high-speed ASVs: Theory and experiments

Postprint of **B.-O. H. Eriksen** and M. Breivik, “Modeling, identification and control of high-speed ASVs: Theory and experiments”, in *Sensing and Control for Autonomous Vehicles: Applications to Land, Water and Air Vehicles*, T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer, Eds. Springer International Publishing, 2017, pp. 407–431, ISBN: 978-3-319-55372-6.

Modeling, Identification and Control of High-Speed ASVs: Theory and Experiments

Bjørn-Olav Holtung Eriksen, Morten Breivik

Abstract This paper considers a powerful approach to modeling, identification and control of high-speed autonomous surface vehicles (ASVs) operating in the displacement, semi-displacement and planing regions. The approach is successfully applied to an 8.45 m long ASV capable of speeds up to 18 m/s, resulting in a high-quality control-oriented model. The identified model is used to design four different controllers for the vessel speed and yaw rate, which have been tested through full-scale experiments in the Trondheimsfjord. The controllers are compared using various performance metrics, and two controllers utilizing a model-based feedforward term is shown to achieve outstanding performance.

1 Introduction

The development of autonomous vehicles is moving rapidly forward. The automotive industry is particularly leading this trend. At sea, there is also a great potential for such vehicles, which are typically referred to as autonomous surface vehicles (ASVs). The use of such vehicles have scientific, commercial and military applications, and can result in reduced costs, increased operational persistence and precision, widened weather window of operations, improved personnel safety, and more environmentally friendly operations. In [1], an early overview of unmanned surface vehicles is given, while a more recent survey is presented in [8].

In this paper, we focus on modeling and control of small, agile ASVs which can operate at high speeds with aggressive maneuvers. These vehicles typically cover the whole range of speed regions for a surface vehicle, namely the displacement, semi-displacement and planing regions. Hence, they are challenging to model and

Bjørn-Olav Holtung Eriksen and Morten Breivik
Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, e-mail: bjorn-olav.h.eriksen@ieee.org and morten.breivik@ieee.org

control, and it therefore becomes challenging to develop a robust and precise motion control system which allows the vehicles to utilize their full potential. Specifically, this paper revisits the modeling and control approach originally suggested in [4] and further developed and reported in [3]. The method represents a control-oriented modeling approach and underlines the importance of developing and using good feedforward terms in the control law. The high-quality performance of the resulting motion control system was validated through several full-scale experiments with ASVs in the Trondheimsfjord in 2008 and 2009, both for target tracking and formation control applications. In this paper, we further develop this approach and go into greater details concerning the modeling and identification procedure and results.

Full-scale identification experiments based on the suggested modeling approach are conducted with a dual-use (manned/unmanned) ASV named Telemetron, see Figure 1, which is owned and operated by the company Maritime Robotics. The resulting identified model is shown to be very precise and cover the entire operational envelope of the ASV. This model subsequently forms the basis for a detailed performance comparison between four qualitatively different controllers, which are implemented and experimentally tested to control the speed and yaw rate of the Telemetron ASV. In particular, the controllers are: A PI feedback (FB) controller; a pure model-based feedforward controller based on the identified model (FF); a controller which is a combination of model-based feedforward and PI feedback (FF-FB); and a controller using feedback signals in the model-based feedforward term in combination with PI feedback, which can be characterized as a feedback linearization (FBL) controller. Relevant performance metrics are defined and used to compare these controllers to determine which is most precise and energy efficient.

Other relevant work concerning a control-oriented modeling approach can be found in e.g. [11] and [10].

The rest of the paper is structured as follows: Chapter 2 presents the main characteristics of the control-oriented modeling approach. Chapter 3 describes the model identification in detail, from experimental design to parameter identification. Chapter 4 describes the four controllers which are considered in the paper, while Chapter 5 presents the results from the motion control experiments. Finally, Chapter 6 concludes the paper.

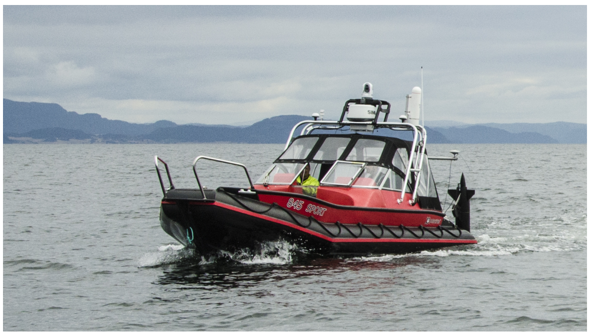


Fig. 1 The Telemetron ASV, which is a Polarcirkel Sport 8.45 m long dual-use ASV capable of speeds up to 18 m/s. Courtesy of Maritime Robotics.

2 2DOF control-oriented vessel model

The vast majority of surface vessel models are based on the 3DOF model [7]:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\nu} \tag{1a}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}_{RB}(\boldsymbol{\nu}) + \boldsymbol{C}_A(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r = \boldsymbol{\tau} + \boldsymbol{\tau}_{\text{wind}} + \boldsymbol{\tau}_{\text{wave}}, \tag{1b}$$

where $\boldsymbol{\eta} = \begin{bmatrix} N & E & \psi \end{bmatrix}^T \in \mathbb{R}^2 \times S^1$ is the vessel pose, $\boldsymbol{\nu} = \begin{bmatrix} u & v & r \end{bmatrix}^T \in \mathbb{R}^3$ is the vessel

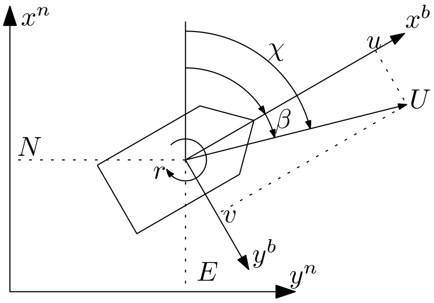


Fig. 2 Vessel variables. The superscripts $(\cdot)^n$ and $(\cdot)^b$ denote the NED and body-frames [7], respectively. The variables N, E and ψ are the vessel pose, u, v and r are the vessel velocity and U is the vessel speed over ground. The course χ is the sum of the heading ψ and the sideslip β.

velocity and $\boldsymbol{\nu}_r$ denotes the relative velocity between the vessel and the water. The terms $\boldsymbol{\tau}, \boldsymbol{\tau}_{\text{wind}}, \boldsymbol{\tau}_{\text{wave}} \in \mathbb{R}^3$ represent the control input, wind and wave environmental disturbances, respectively. The matrix $\boldsymbol{R}(\psi)$ is the rotation matrix about the z-axis, the inertia matrix is $\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A$ where $\boldsymbol{M}_{RB}$ is the rigid-body mass and $\boldsymbol{M}_A$ is the added mass caused by the moving mass of water. The matrices $\boldsymbol{C}_{RB}(\boldsymbol{\nu})$ and $\boldsymbol{C}_A(\boldsymbol{\nu}_r)$ represent the rigid-body and hydrodynamic Coriolis and centripetal effects, respectively, while $\boldsymbol{D}(\boldsymbol{\nu}_r)$ captures the hydrodynamic damping of the vessel. An important limitation of (1b) is that it can be challenging to use for vessels operating outside of the displacement region. For approximating the operating region of a surface vessel, it is common to use the Froude number, defined as [6]:

$$Fn = \frac{U_r}{\sqrt{Lg}}, \tag{2}$$

where U_r is the vessel speed through water, L is the submerged vessel length and g is the acceleration of gravity. For Fn less than approximately 0.4, the hydrostatic pressure mainly carries the weight of the vessel, and we operate in the displacement region. When Fn is higher than 1.0 to 1.2, the hydrodynamic force mainly carries the weight of the vessel, and we operate in the planing region. For Fn between these values, we are in the semi-displacement region [6].

Typical ASVs have vessel lengths of up to 10 m, submerged length of up to 8 m and operating speeds up to 18 m/s. From Table 1, we see that an ASV with a submerged length of 8 m exits the displacement region already at 3.54 m/s, and enters the planing region at 8.86 m/s. Hence, (1b) is typically only suited for a small part of the ASV operating region, which motivates for an alternative model.

ASVs are generally underactuated, hence it it not possible to independently control surge, sway and yaw. We therefore choose to reduce the model to the 2DOF

Table 1: Operating speeds for displacement and planing regions. *Supply ships typically operate with speeds up to 7 m/s. It is therefore clear that supply ships generally operate in the displacement region.

Vessel type	**Submerged length**	**Maximum speed in displacement** ($Fn = 0.4$)	**Minimum speed in planing** ($Fn = 1.0$)
Small ASV	4 m	2.51 m/s	6.26 m/s
Large ASV	8 m	3.54 m/s	8.86 m/s
Small supply ship	50 m	8.86 m/s*	22.1 m/s*
Large supply ship	100 m	12.5 m/s*	31.3 m/s*

which we want to control, namely the speed over ground (SOG) $U = \sqrt{u^2 + v^2}$ and yaw rate (rate of turn, ROT). The kinematic equation (1a) is therefore modified to:

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \begin{bmatrix} \cos(\chi) & 0 \\ \sin(\chi) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ r \end{bmatrix} \\ \dot{\chi} &= r + \dot{\beta}, \end{aligned} \tag{3}$$

where $\chi = \psi + \beta$ is the vessel course angle and β is the vessel sideslip. It should be noted that this model implies that:

- Since $U \geq 0$, we assume that the vessel is traveling forward, that is $u \geq 0$.
- The sideslip β enters the kinematic equation. For kinematic control (e.g. path following), this must be addressed by e.g. controlling course instead of heading.

To relax the limitation of operating in the displacement region implied by (1b), we propose a normalized non first-principles model. This is inspired by [4] and [3] where a steady-state model in a similar form is developed. Since the actual control input of the vessel is not forces, but rather motor throttle and rudder angle, we select these as inputs to the model. As a result, we also implicitly model the actuator dynamics. Let the motor throttle be given as $\tau_m \in [0, 1]$ and the rudder input be given as $\tau_\delta \in [-1, 1]$. Denoting the vessel velocity as $\boldsymbol{x} = \begin{bmatrix} U & r \end{bmatrix}^T \in \mathbb{R}^2$ and the control input as $\boldsymbol{\tau} = \begin{bmatrix} \tau_m & \tau_\delta \end{bmatrix}^T \in \mathbb{R}^2$, we propose the model:

$$\boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}} + \boldsymbol{\sigma}(\boldsymbol{x}) = \boldsymbol{\tau}, \tag{4}$$

where the inertia matrix $\boldsymbol{M}(\boldsymbol{x}) = \operatorname{diag}(m_U(\boldsymbol{x}), m_r(\boldsymbol{x}))$ is diagonal with elements of quantities $\left[\frac{1}{\mathrm{m/s^2}}\ \frac{1}{1/\mathrm{s^2}}\right]$, and $\boldsymbol{\sigma}(\boldsymbol{x}) = \begin{bmatrix} \sigma_U(\boldsymbol{x}) & \sigma_r(\boldsymbol{x}) \end{bmatrix}^T$ is a unit-less damping term. Notice that both are functions of $\boldsymbol{x}$, which allows for a nonlinear model. The reader should also note that centripetal effects are not explicitly included in (4) due to the choice of coordinates.

3 Model identification

Identifying the parameters of (4) require a series of experiments to be performed. In this section, we describe the identification experiments, parameterization of the inertia and damping terms, and the methodology used for parameter identification.

3.1 Vessel platform and hardware

As already mentioned, the vessel used in this work is the Telemetron ASV. It is a dual-use vessel for both manned and unmanned operations, and is equipped with a number of sensors and a proprietary control system. Some of the specifications are summarized in Table 2.

Table 2: Telemetron ASV specifications.

Component	**Description**
Vessel hull	Polarcirkel Sport 845
Length	8.45 m
Width	2.71 m
Weight	1675 kg
Propulsion system	Yamaha 225 HP outboard engine
Motor control	Electro-mechanical actuation of throttle valve
Rudder control	Hydraulic actuation of outboard engine angle with proportional-derivate (PD) feedback control
Navigation system	
Identification experiments	Kongsberg Seatex Seapath 330+
Control experiments	Hemisphere Vector VS330

3.2 Identification experiment design

Since we wish to identify damping and inertia terms, both steady-state and transient information is required. We therefore construct a series of step responses:

- Step changes in τ_m given a series of fixed rudder settings τ_δ, illustrated as orange trajectories in Figure 3.
- Step changes in τ_δ given a series of fixed throttle settings τ_m, illustrated as blue trajectories in Figure 3.

The steps are performed both for increasing and decreasing values to include the effect of hysteresis, and is designed to sample the U/r-space of the vessel as shown in Figure 3. It is assumed that the vessel response is symmetric in yaw, such that it is sufficient to perform experiments only for positive rudder settings (which for the Telemetron ASV result in positive yaw rate). The vessel shall reach steady state between the step changes such that the damping terms can be identified from the steady-state response, while the inertia is identified from the transient response. The motor will be kept in forward gear throughout the entire experiment.

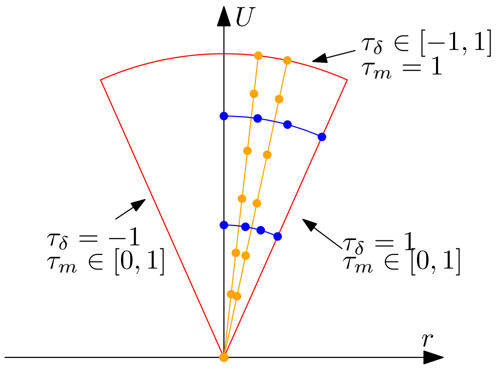


Fig. 3 Expected shape of the vessel velocity space, where the red line is the boundary of the velocity space. The orange and blue lines are examples of step change trajectories for fixed rudder and throttle, respectively. Note that only some trajectories are illustrated. The dots on the trajectories illustrate steady-state points.

The step changes in τ_m are performed as:

1. Start at $\tau_m = 0$. Select $\tau_\delta = 0$.
2. Step τ_m stepwise from 0 to 1 in steps of 0.1, letting the vessel SOG and ROT reach steady state before the next step is applied. Let the vessel do at least one full turn after reaching steady state, to be able to minimize the effect of external disturbances through averaging.
3. Step τ_m stepwise from 1 to 0, in the same fashion as in step 2.
4. Repeat step 2 and 3 with the next rudder setting.

Step changes in τ_δ are performed by interchanging τ_m and τ_δ. Identification experiments were carried out in the Trondheimsfjord 17th and 18th of December 2015.

3.3 Measurement extraction

To identify parameters for $\boldsymbol{M}(\boldsymbol{x})$ and $\boldsymbol{\sigma}(\boldsymbol{x})$, we need measurements of σ_U, σ_r, m_U and m_r for different vessel states $\boldsymbol{x}$.

3.3.1 Extraction of damping data

When the vessel is in steady state, the model (4) gives the relation:

$$\dot{\boldsymbol{x}} = \boldsymbol{0} \rightarrow \boldsymbol{\sigma}(\boldsymbol{x}) = \boldsymbol{\tau}, \tag{5}$$

hence measurements of the damping term can be taken simply as the control input when the vessel is at steady state. To reduce the influence of external forces, the vessel state is averaged to extract measurements for σ_U and σ_r. This is shown for one of the fixed rudder settings in Figure 4. We observed that the motor response is greatly reduced for $\tau_m > 0.6$, hence measurements with $\tau_m > 0.6$ are omitted.

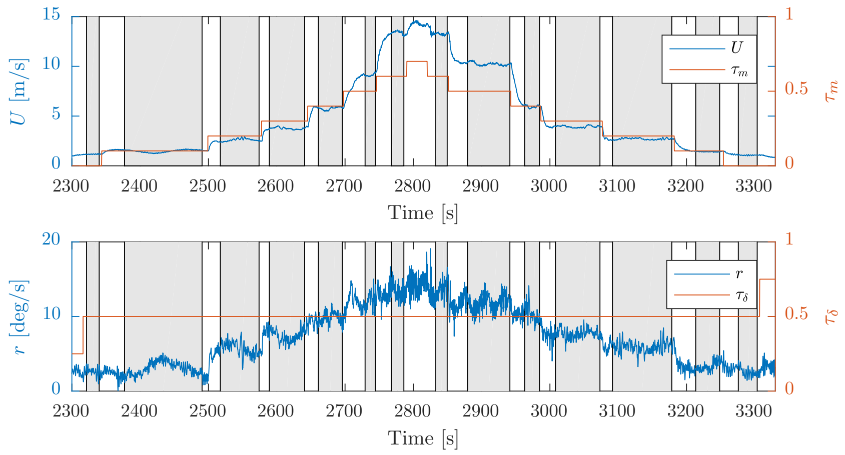


Fig. 4: Vessel response with a fixed rudder setting. The gray patches mark steady state regions.

By averaging the steady-state regions, we generate a set of N_σ measurements $\mathscr{D}_{\boldsymbol{\sigma}} = \{\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{N_\sigma}\}, \{\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2, \ldots, \boldsymbol{\sigma}_{N_\sigma}\}, \{\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_{N_\sigma}\}\}$, which can be used for identifying parameters for the damping term. The damping measurements, with mirrored values for negative rudder settings, are shown in Figure 5.

3.3.2 Extraction of inertia data

To extract measurements for m_U and m_r, we have N_m step changes, and we create an estimate of the vessel response using N_m local first-order linear models. We approximate the SOG and ROT dynamics as SISO systems, hence for the i-th step, the linear approximation of the vessel SOG can be written as:

$$m_{U_i} \Delta \dot{U}_i + k_i \Delta U_i = \Delta \tau_{m_i}, \tag{6}$$

where the inertia m_{U_i} is assumed to be constant during the step, $k_i = \frac{\sigma_{U_i}^+ - \sigma_{U_i}^-}{U_i^+ - U_i^-}$, where $(\cdot)^-$ and $(\cdot)^+$ denotes the value prior to and after the step, is a linearized damping term, $\Delta U_i = U - U_i^-$ and $\Delta \tau_{m_i} = \tau_m - \tau_{m_i}^-$. The only unknown in (6) is the inertia m_{U_i}, hence we can find a suitable inertia m_{U_i} by simulating (6) for a set of possible inertias and selecting the inertia with the smallest squared estimation error, as shown in Figure 6. The measurement is taken as $(\boldsymbol{x}, m_U) = \left(\left(\frac{U_i^+ + U_i^-}{2}, \frac{r_i^+ + r_i^-}{2}\right), m_{U_i}\right)$. The same approach is employed for identifying inertia for ROT.

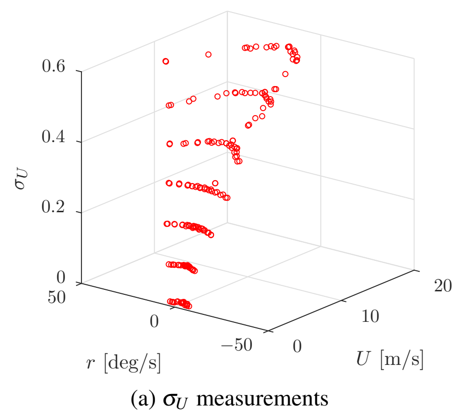


Fig. 5: Damping term measurements from averaging of steady-state responses. Measurements for negative rudder settings are obtained by mirroring the data.

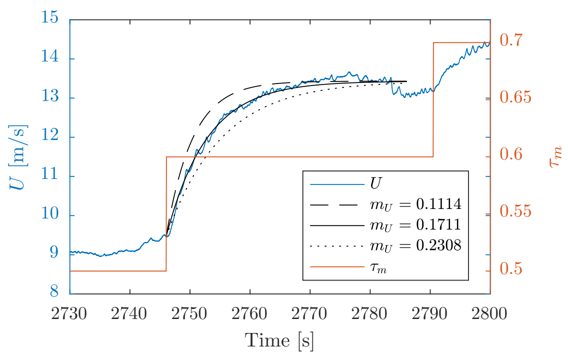


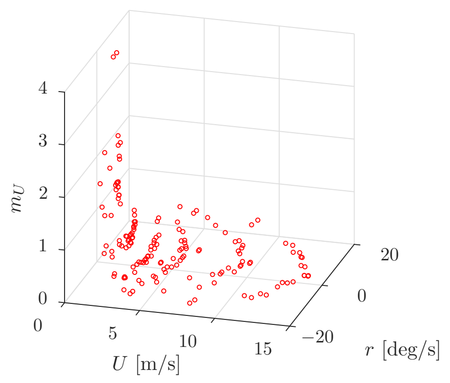
Fig. 6 Inertia measurement extraction for a step in the SOG. It is clear that $m_U = 0.1711$ is the best fit.

It should be noted that, in contrast to identifying damping, we obtain two sets of measurements $\mathscr{D}_{m_U} = \left\{\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{N_{m_U}}\}, \{m_{U_1}, m_{U_2}, \ldots, m_{U_{N_{m_U}}}\}\right\}$ and $\mathscr{D}_{m_r} = \left\{\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{N_{m_r}}\}, \{m_{r_1}, m_{r_2}, \ldots, m_{r_{N_{m_r}}}\}\right\}$ containing N_{m_U} and N_{m_r} measurements respectively. The inertia measurements are shown in Figure 7.

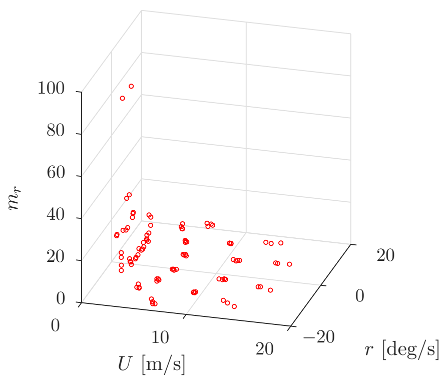
3.3.3 Data preprocessing

Before the measurements are used for parameter identification, some preprocessing is required:

- Damping measurements with $\tau_\delta = 0$ should result in zero yaw rate. Even though we average the steady-state response, some offset will be present. Hence, all



(a) m_U measurements



(b) m_r measurements

Fig. 7: Inertia term measurements. Measurements for negative rudder settings are obtained by mirroring the data.

measurements $(U, r, \sigma_U, \sigma_r) \in \mathscr{D}_{\boldsymbol{\sigma}}$ with $\tau_\delta = \sigma_r = 0$ should be be modified as $(U, r, \sigma_U, \sigma_r) = (U, 0, \sigma_U, \sigma_r)$.

- Since the domains of U and r are different, the measurements should be normalized. We have applied zero-mean and unit variance normalization individually for each measurement set $\mathscr{D}_{\boldsymbol{\sigma}}, \mathscr{D}_{m_U}$ and $\mathscr{D}_{m_r}$.

3.4 Parameter identification

This section describes identification of the parameters of (4) based on the measurement sets $\mathscr{D}_{\boldsymbol{\sigma}}, \mathscr{D}_{m_U}$ and $\mathscr{D}_{m_r}$.

3.4.1 Linear regression

For identification of model parameters, we use linear regression [2]. This requires that the terms in (4) are linear in the parameters, e.g. that the damping and inertia terms can be written as:

$$\begin{aligned} \sigma_U(\boldsymbol{x}) &= \boldsymbol{\phi}_\sigma(\boldsymbol{x})^T \boldsymbol{\beta}_{\sigma_U}, & \sigma_r(\boldsymbol{x}) &= \boldsymbol{\phi}_\sigma(\boldsymbol{x})^T \boldsymbol{\beta}_{\sigma_r} \\ m_U(\boldsymbol{x}) &= \boldsymbol{\phi}_M(\boldsymbol{x})^T \boldsymbol{\beta}_{m_U}, & m_r(\boldsymbol{x}) &= \boldsymbol{\phi}_M(\boldsymbol{x})^T \boldsymbol{\beta}_{m_r}, \end{aligned} \tag{7}$$

where $\boldsymbol{\phi}_\sigma(\boldsymbol{x})$ and $\boldsymbol{\phi}_M(\boldsymbol{x})$ are vectors of basis functions (also called regressors) while $\boldsymbol{\beta}_{\sigma_U}, \boldsymbol{\beta}_{\sigma_r}, \boldsymbol{\beta}_{m_U}$ and $\boldsymbol{\beta}_{m_r}$ are parameter vectors. This generalizes as a function:

$$\hat{y} = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\beta}. \tag{8}$$

For the model (8), one can, given a data set $\{\{\boldsymbol{x}_1,\boldsymbol{x}_2,\dots,\boldsymbol{x}_N\},\{y_1,y_2,\dots,y_N\}\}$ and a parameter vector $\boldsymbol{\beta}$, define the weighted square loss function:

$$\varepsilon = \frac{1}{N}\sum_{i=1}^{N} W_{ii}\left(y_i - \boldsymbol{\phi}(\boldsymbol{x}_i)^T\boldsymbol{\beta}\right)^2, \tag{9}$$

where W_{ii} is a weight for sample i. By defining $\boldsymbol{Y} = \left[y_1\ y_2\ \dots\ y_N\right]^T$ and $\boldsymbol{X} = \left[\boldsymbol{\phi}(\boldsymbol{x}_1)^T\ \boldsymbol{\phi}(\boldsymbol{x}_2)^T\ \dots\ \boldsymbol{\phi}(\boldsymbol{x}_N)^T\right]^T$ one can find the $\boldsymbol{\beta}$ that minimizes (9) as:

$$\boldsymbol{\beta} = (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{Y}, \tag{10}$$

where $\boldsymbol{W} = \mathrm{diag}(W_{11},W_{22},\dots,W_{NN})$. This is known as weighted linear least-squares regression.

A well known issue with linear regression, especially with large parameter vectors, is the problem of overfitting. To reduce this problem, one can penalize large parameter values by adding a regularization term to (9) as:

$$\varepsilon = \frac{1}{N}\sum_{i=1}^{N} W_{ii}\left(y_i - \boldsymbol{\phi}(\boldsymbol{x}_i)^T\boldsymbol{\beta}\right)^2 + \lambda R(\boldsymbol{\beta}), \tag{11}$$

where $\lambda > 0$ is a regularization weight, and the choice of the regularization term $R(\boldsymbol{\beta})$ is problem dependent. We choose ℓ_1-regularization where $R(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1$, also known as lasso, which has the property of driving parameters to zero for sufficiently high values of λ [2]. This penalizes basis functions with low sensitivities to the loss function, and favors sparsity in the parameter vector.

It should be noted that introducing regularization provides one parameter more to the problem, in form of the regularization weight λ. Additionally, there exist no closed form solution to minimizing (11) with respect to $\boldsymbol{\beta}$. However, given a regularization parameter λ, the solution can be found through quadratic programming techniques.

3.4.2 Cross-validation (CV)

For identifying hyperparameters, such as the regularization weight λ, one can use cross-validation (CV). This involves dividing the available data into a training set and a validation set, where the training set is used for solving the parameter estimation while using the validation set for evaluating the loss. Hyperparameters can then be identified by minimizing the loss with respect to the hyperparameters. There exist different methods for dividing the available data, e.g. κ-fold, leave-p-out and leave-one-out (which is a special case of leave-p-out). Leave-one-out CV evaluates all possible combinations of leaving one sample for the validation set, hence for a data set of N samples this will result in N combinations of training and validation sets. We chose to use leave-one-out CV based on this property, while the limited data size ensures computational feasibility.

It should be noted that when performing both positive and negative step changes (see Figure 4), steady-state points with the same $\boldsymbol{\tau}$ will have quite similar (U,r)

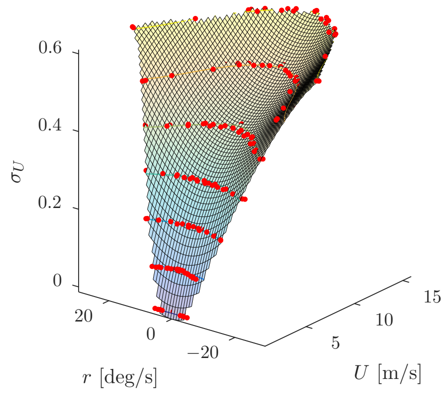
coordinates. Hence, one should handle groups of measurements when dividing the measurements into training and validation data.

3.4.3 Damping term

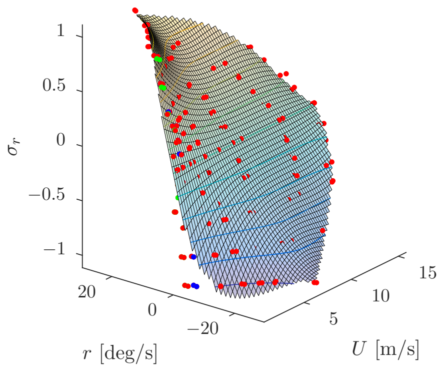
From the structure of the damping measurements in Figure 5, we propose to use polynomial basis functions for the damping term in (4). This is also motivated by [7] where polynomial damping terms are used. The power of the polynomial is chosen as four, which is assumed to be sufficient to capture hydrodynamic damping and actuator dynamics. Hence, the regressor is defined as the 15-element vector:

$$\boldsymbol{\phi}_\sigma(\boldsymbol{x}) = \left[1,\, U,\, r,\, U^2,\, Ur,\, r^2,\, U^3,\, U^2r,\, Ur^2,\, r^3,\, U^4,\, U^3r,\, U^2r^2,\, Ur^3,\, r^4\right]^T. \quad (12)$$

The parameter vectors $\boldsymbol{\beta}_{\sigma_U}$ and $\boldsymbol{\beta}_{\sigma_r}$ are identified by minimizing (11) with respect to $\boldsymbol{\beta}$. The regularization parameter is found as described in Section 3.4.2. A surface plot of the damping function is shown in Figure 8.



(a) $\sigma_U(\boldsymbol{x})$



(b) $\sigma_r(\boldsymbol{x})$

Fig. 8: Polynomial function for the damping term. The scatter points are the measuring points, where red points have weight $W = 1$, blue points have $W = 0.5$ and green points have $W = 0.1$.

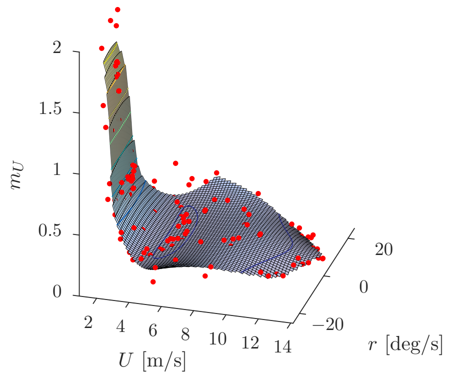
3.4.4 Inertia term

From the structure of the inertia measurements in Figure 7, it is clear that a polynomial model will struggle to fit the data well. We therefore introduce an asymptotic basis function $\tanh(a(U-b))$ in addition to the polynomial terms. The regressor for the inertia terms is hence defined as the 16-element vector:

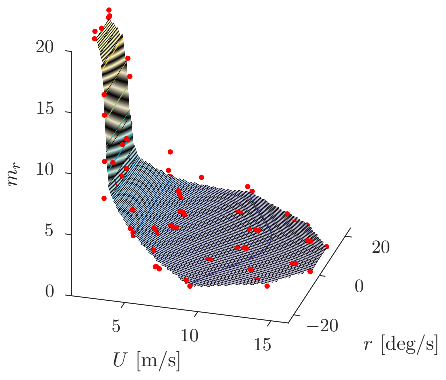
$$\boldsymbol{\phi}_M(\boldsymbol{x}) = \big[1,\, U,\, r,\, U^2,\, Ur,\, r^2,\, U^3,\, U^2r,\, Ur^2,\, r^3,\, U^4,\, U^3r,\, U^2r^2,\, Ur^3,\, r^4,\, \tanh(a(U-b))\big]^T . \tag{13}$$

Notice that the asymptotic basis function introduces two more hyperparameters in the regression problem, namely a and b. To identify these hyperparameters, we again use leave-one-out CV, as described in Section 3.4.2. Notice that we use regularization when we identify these hyperparameters, individually of the linear regression. The motivation for this is that the position of the steep asymptote in the inertial measurement will move with changing ocean currents and external forces. Adding regularization when identifying the hyperparameters increases the robustness of the identified inertia term by adding a cost to choosing high parameter values for the asymptotic term and hence limiting the gradient of the asymptotic term.

The parameter vectors $\boldsymbol{\beta}_{m_U}, \boldsymbol{\beta}_{m_r}$ are, as for the damping term, identified by minimizing (11) with respect to $\boldsymbol{\beta}$. The hyperparameters are identified using CV. It should be noted that we use ℓ_1-regularization when identifying the hyperparameters (a_{m_U}, b_{m_U}) and (a_{m_r}, b_{m_r}).



(a) $m_U(\boldsymbol{x})$



(b) $m_r(\boldsymbol{x})$

Fig. 9: Function for the inertia term. The scatter points are the measuring points.

3.5 Model verification

To qualitatively verify the identified vessel model, we simulate the model with the input sequence from an experiment not used in the model identification and compare the results. The model (4), with damping and inertia parameterization and parameters as identified in Section 3.4, is simulated with the recorded input sequence to obtain the response shown in Figure 10. Based on the comparison, we see that the model captures the dynamics of the vessel, although with slight offsets especially for ROT. The simulated transient response coincides well with the real vessel response.

A design choice for the identification experiments was the assumed shape of the vessel operating space, discussed in Section 3.2 and illustrated in Figure 3. This design choice is verified by estimating the actual vessel operating space. This can be generated by using all the steady-state velocities obtained during the identification, as shown in Figure 11. By comparing the actual and assumed operating spaces, we see that the shapes are very similar.

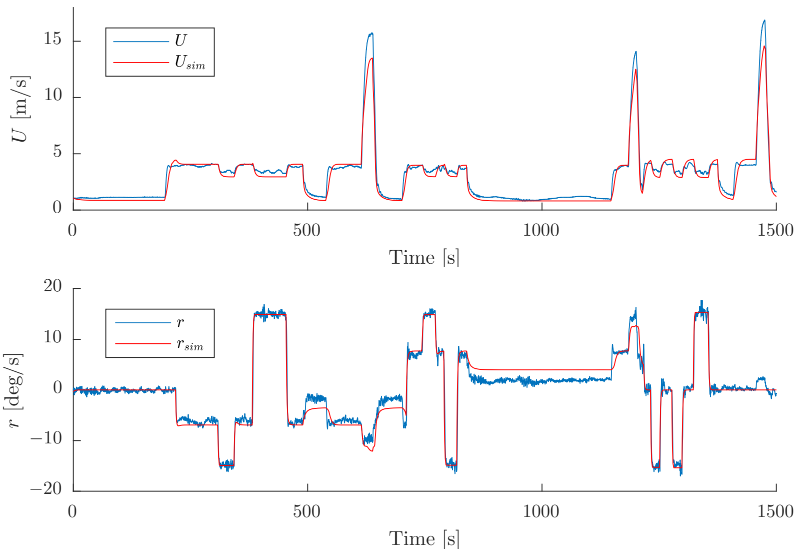
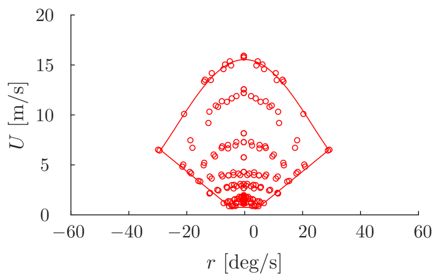


Fig. 10: Real and simulated vessel response. The deviation at high SOG is caused by exiting the valid domain of the identified model.

Fig. 11 Identified steady-state velocities. The red boundary line is estimated by least-squares curve fitting a fourth order polynomial. Note that that $U < 0.75$ m/s is not part of the vessel operating space as ocean current lower-bound the SOG.



4 Controller design

In this section, we design four controllers to be compared through experiments:

1. A proportional-integral feedback (FB) controller.
2. A feedforward (FF) controller.
3. A combined feedforward and feedback (FF-FB) controller.
4. A feedback-linearizing (FBL) controller.

4.1 Controller types

This section describes the controller formulations, and the resulting closed-loop dynamics.

4.1.1 Model uncertainties

The model (4) does not account for modeling uncertainties. For closed-loop analysis, we therefore add an unknown bias term and modify the model as:

$$\boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}} + \boldsymbol{\sigma}(\boldsymbol{x}) = \boldsymbol{\tau} + \boldsymbol{b}, \tag{14}$$

where $\boldsymbol{b}$ is assumed to be slowly varying, hence $\dot{\boldsymbol{b}} \approx \boldsymbol{0}$.

4.1.2 Proportional-integral feedback (FB) controller

The FB controller is a proportional-integral controller with gain scheduling of the proportional gain using the inertia term of the identified model (4):

$$\boldsymbol{\tau}_{FB} = -\boldsymbol{M}(\boldsymbol{x})\boldsymbol{K}_p\tilde{\boldsymbol{x}} - \boldsymbol{K}_i\int_{t_0}^{t}\tilde{\boldsymbol{x}}(\gamma)\mathrm{d}\gamma, \tag{15}$$

where $\boldsymbol{K}_p > 0$ is a diagonal proportional gain matrix, $\boldsymbol{K}_i > 0$ is a diagonal integral gain matrix and $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_d$. By inserting (15) into (14) we derive the error dynamics:

$$\dot{\tilde{\boldsymbol{x}}} = -\boldsymbol{K}_p\tilde{\boldsymbol{x}} + \boldsymbol{M}(\boldsymbol{x})^{-1}\left(\boldsymbol{b} - \boldsymbol{\sigma}(\boldsymbol{x}) - \boldsymbol{K}_i\int_{t_0}^{t}\tilde{\boldsymbol{x}}(\gamma)\mathrm{d}\gamma\right) + \dot{\boldsymbol{x}}_d, \tag{16}$$

where we see that the integrator must compensate for modeling errors and damping. Even if $\boldsymbol{K}_i\int_{t_0}^{t}\tilde{\boldsymbol{x}}(\gamma)\mathrm{d}\gamma = \boldsymbol{b} - \boldsymbol{\sigma}(\boldsymbol{x})$ and $\tilde{\boldsymbol{x}} = \boldsymbol{0}$, we will still not be able to track a changing reference since $\boldsymbol{\sigma}(\boldsymbol{x})$ is changing with $\boldsymbol{x}$, and $\dot{\boldsymbol{x}}_d \neq \boldsymbol{0}$ for a changing reference.

4.1.3 Feedforward (FF) controller

The model-based FF controller feedforwards the desired acceleration and velocity:

$$\boldsymbol{\tau}_{FF} = \boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}}_d + \boldsymbol{\sigma}(\boldsymbol{x}_d). \tag{17}$$

Notice that we use the measured state $\boldsymbol{x}$ when computing the inertia term, and the desired state $\boldsymbol{x}_d$ when computing the damping term. The error dynamics becomes:

$$\dot{\tilde{\boldsymbol{x}}} = \boldsymbol{M}(\boldsymbol{x})^{-1}\left(\boldsymbol{\sigma}(\boldsymbol{x}_d) - \boldsymbol{\sigma}(\boldsymbol{x}) + \boldsymbol{b}\right), \tag{18}$$

which has an equilibrium $\boldsymbol{x} = \boldsymbol{\sigma}^{-1}(\boldsymbol{\sigma}(\boldsymbol{x}_d) + \boldsymbol{b})$, given that $\boldsymbol{\sigma}^{-1}(\cdot)$ is well-defined. Hence, if $\boldsymbol{b} \neq \boldsymbol{0}$ we will have some tracking and steady-state offset.

4.1.4 Combined feedforward and feedback (FF-FB) controller

The FF-FB controller combines the FF and FB controllers as:

$$\boldsymbol{\tau}_{FF\text{-}FB} = \boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}}_d + \boldsymbol{\sigma}(\boldsymbol{x}_d) - \boldsymbol{M}(\boldsymbol{x})\boldsymbol{K}_p\tilde{\boldsymbol{x}} - \boldsymbol{K}_i\int_{t_0}^{t}\tilde{\boldsymbol{x}}(\gamma)\mathrm{d}\gamma. \tag{19}$$

Inserting (19) into (14), we can derive the error dynamics:

$$\dot{\tilde{\boldsymbol{x}}} = -\boldsymbol{K}_p\tilde{\boldsymbol{x}} + \boldsymbol{M}(\boldsymbol{x})^{-1}\left(\boldsymbol{\sigma}(\boldsymbol{x}_d) - \boldsymbol{\sigma}(\boldsymbol{x}) + \boldsymbol{b} - \boldsymbol{K}_i\int_{t_0}^{t}\tilde{\boldsymbol{x}}(\gamma)\mathrm{d}\gamma\right), \tag{20}$$

where one should notice that if the $\boldsymbol{\sigma}(\boldsymbol{x}_d)$ would be substituted with $\boldsymbol{\sigma}(\boldsymbol{x})$ we would have a feedback-linearizing controller. The motivation for using $\boldsymbol{\sigma}(\boldsymbol{x}_d)$ in (17) is to increase the robustness and introduce an extra ”driving” term in addition to the proportional feedback driving the error to zero.

4.1.5 Feedback-linearizing (FBL) controller

The FBL controller is similar to (19), but computes the damping term for the measured velocity:

$$\boldsymbol{\tau}_{FBL} = \boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}}_d + \boldsymbol{\sigma}(\boldsymbol{x}) - \boldsymbol{M}(\boldsymbol{x})\boldsymbol{K}_p\tilde{\boldsymbol{x}} - \boldsymbol{K}_i\int_{t_0}^{t}\tilde{\boldsymbol{x}}(\gamma)\mathrm{d}\gamma, \tag{21}$$

which can cause poor robustness with respect to disturbances and time delays in the control system. The FBL controller is often used for analysis of closed-loop systems due to the simple error dynamics:

$$\dot{\tilde{\boldsymbol{x}}} = -\boldsymbol{K}_p\tilde{\boldsymbol{x}} + \boldsymbol{M}(\boldsymbol{x})^{-1}\left(\boldsymbol{b} - \boldsymbol{K}_i\int_{t_0}^{t}\tilde{\boldsymbol{x}}(\gamma)\mathrm{d}\gamma\right). \tag{22}$$

4.2 Control architecture

The FB, FF and FF-FB controllers in Section 4.1 are realized by enabling the feedback, feedforward and both functions shown in Figure 12, respectively. The FBL controller is realized by combining the feedback and feedforward functions, while computing the feedforward damping as $\boldsymbol{\sigma}(\boldsymbol{x})$ instead of $\boldsymbol{\sigma}(\boldsymbol{x}_d)$.

To increase robustness in the implementation, saturation elements are placed at each output except for the proportional feedback element.

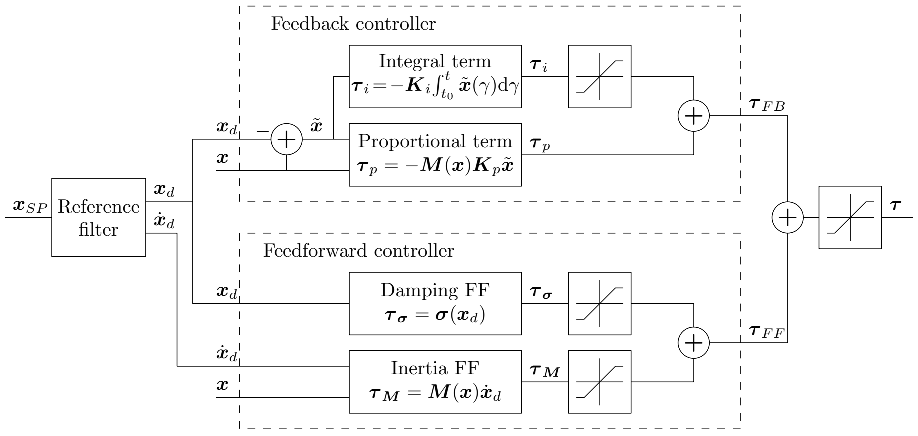


Fig. 12: Control architecture. The different controllers are realized through combinations of the feedback and feedforward functions.

To ensure continuous reference signals $\boldsymbol{x}_d$ and $\dot{\boldsymbol{x}}_d$, we employ a second-order reference filter from a possibly discontinuous, user-specified setpoint signal $\boldsymbol{x}_{SP}$. Additionally, we limit the acceleration such that the reference signals are feasible with respect to the vessel capability. The filter is parameterized as [7]:

$$\begin{bmatrix} \dot{\boldsymbol{x}}_d \\ \ddot{\boldsymbol{x}}_d \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ -\boldsymbol{\Omega}^2 & -2\boldsymbol{\Delta}\boldsymbol{\Omega} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_d \\ \dot{\boldsymbol{x}}_d \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\Omega}^2 \end{bmatrix} \boldsymbol{x}_{SP} \tag{23}$$

while imposing the acceleration limits:

$$\dot{U}_d \in \left[\dot{U}_{d_{\min}}, \dot{U}_{d_{\max}}\right], \quad \dot{r}_d \in \left[\dot{r}_{d_{\min}}, \dot{r}_{d_{\max}}\right]. \tag{24}$$

The relative damping ratio matrix $\boldsymbol{\Delta} > 0$ is chosen as identity to achieve a critically damped system, while the diagonal natural frequency matrix $\boldsymbol{\Omega} > 0$ is a tuning parameter.

5 Motion control experiments

To evaluate the performance of the controllers described in Section 4.1, they were implemented on the Telemetron ASV and tested in the Trondheimsfjord on the 13th and 14th of October 2016. During the first day, the sea state can be characterized as calm, which refer to significant wave heights of 0–0.1 m, while the sea state for the second day can be characterized as slight, which refer to significant wave heights of 0.5–1.25 m [9]. In total, three different scenarios were tested in different sea states.

It should be noted that the time between the model identification and motion control experiments was about 10 months. The top speed of the vessel was reduced from 18 m/s to about 16 m/s, probably caused by algae growth on the hull.

5.1 Tuning parameters

The reference filter was tuned with a natural frequency of $\boldsymbol{\Omega} = \mathrm{diag}(0.4, 1)$ and acceleration constraints $\dot{U}_{\max} = 0.75$ m/s^2, $\dot{U}_{\min} = -0.75$ m/s^2, $\dot{r}_{\max} = 0.1$ rad/s^2 and $\dot{r}_{\min} = -0.1$ rad/s^2. The feedback tuning parameters were selected as shown in Table 3. Unfortunately, an implementation error resulted in a too high integrator gain for the yaw rate feedback controller during the experiments in calm seas.

Table 3: Feedback tuning parameters.

Parameters	**Values**		
	FB	FF-FB	FBL
Sea state - Calm:			
$\boldsymbol{K}_p$	diag(0.15,0.75)	diag(0.15,0.75)	diag(0.15,0.75)
$\boldsymbol{K}_i$	diag(0.015,0.5)	diag(0.015,0.5)	diag(0.015,0.5)
Sea state - Slight:			
$\boldsymbol{K}_p$	diag(0.15,1)	diag(0.1,0.5)	diag(0.1,0.5)
$\boldsymbol{K}_i$	diag(0.01,0.25)	diag(0.0067,0.125)	diag(0.0067,0.125)

5.2 Performance metrics

To compare controller performance, it is beneficial to define suitable performance metrics. To simplify the analysis, it is also beneficial to combine the control inputs and outputs to one input and one output when calculating the metrics. Since the outputs have different units, we define the normalized signals $\bar{U}, \bar{U}_d, \bar{r}$ and $\bar{r}_d$ that are in the interval $[0, 1]$ in the expected operation space of the vessel. A combined error and control input can then be computed as:

$$\bar{e}(t) = \sqrt{(\bar{U}(t) - \bar{U}_d(t))^2 + (\bar{r}(t) - \bar{r}_d(t))^2}, \quad \bar{\tau}(t) = \sqrt{\tau_m^2 + \tau_\delta^2}. \tag{25}$$

Given these signals, we can define the integral of absolute error (IAE):

$$IAE(t) = \int_{t_0}^{t} |\bar{e}(\gamma)| \mathrm{d}\gamma, \tag{26}$$

which penalizes the error linearly with the magnitude and serves as a measure of control precision. A similar metric is the integral of square error (ISE), which penalizes large errors more than small errors.

The integral of absolute differentiated control (IADC) has been used earlier in a combined performance metric in [13], and is defined as:

$$IADC(t) = \int_{t_0}^{t} |\dot{\bar{\tau}}(\gamma)| \mathrm{d}\gamma, \tag{27}$$

which penalizes actuator changes and serves as a measure of actuator wear and tear.

The integral of absolute error times the integral of absolute differentiated control (IAE-ADC) is a combination of IAE and IADC:

$$IAE\text{-}ADC(t) = \int_{t_0}^{t} |\bar{e}(\gamma)| \mathrm{d}\gamma \int_{t_0}^{t} |\dot{\bar{\tau}}(\gamma)| \mathrm{d}\gamma, \tag{28}$$

which serves as a measure of control precision versus wear and tear.

The integral of absolute error times work (IAEW) scales IAE with energy consumption [12]:

$$IAEW(t) = \int_{t_0}^{t} |\bar{e}(\gamma)| \mathrm{d}\gamma \int_{t_0}^{t} P(\gamma) \mathrm{d}\gamma, \tag{29}$$

where $P(t)$ is the mechanical power applied by the engine. IAEW measures control precision versus energy consumption, hence it quantifies the energy efficiency. It is common to model the applied propeller force F as proportional to the square of the propeller speed, hence, $F \propto |n|n$ [7]. The mechanical energy can then be written as:

$$P(t) \propto U(t)|n(t)|n(t). \tag{30}$$

Since we use the metric in a relative comparison, we do not care about any scaling constant and set $P(t) = U(t)|n(t)|n(t)$.

5.3 Experiments in slight seas

All the scenarios were tested in slight seas, and here we present two of the scenarios.

5.3.1 Test 1 - High-speed trajectory tracking with steady states

The first test was intended to test a large portion of the vessel operating space while measuring both steady-state and transient performance. The test is symmetric in U_d and anti-symmetric in r_d. The vessel response in slight seas is shown in Figure 13.

Immediately, we observe that the FBL controller suffers from instability, caused by the feedback term $\boldsymbol{\sigma}(\boldsymbol{x})$ in (21). The oscillatory vessel state causes a dropout of the navigation system, stopping the experiment at $t \approx 194$ s. In general, using sensor measurements in model-based feedforward terms reduces the robustness with respect to time delays, sensor dropouts and noise. Using the reference in the feedforward terms avoids these problems. The FBL controller is not used in the other tests. The FF controller achieves good tracking, but naturally with some steady-state offset. The FB controller achieves poor tracking, while also being largely influenced by disturbances. The FF-FB controller has similar (or better) tracking performance than the FF controller while avoiding steady-state offsets, and at the same time better disturbance rejection than the FB controller. The FF, FF-FB and FBL controllers fail in tracking the first transient due to the control system time delay which causes problems with capturing the steep transient in the inertia term. It might be beneficial to limit the gradient $\nabla_{\boldsymbol{x}} \boldsymbol{M}(\boldsymbol{x})$ or saturating the inertia $\boldsymbol{M}(\boldsymbol{x})$ to avoid this behavior.

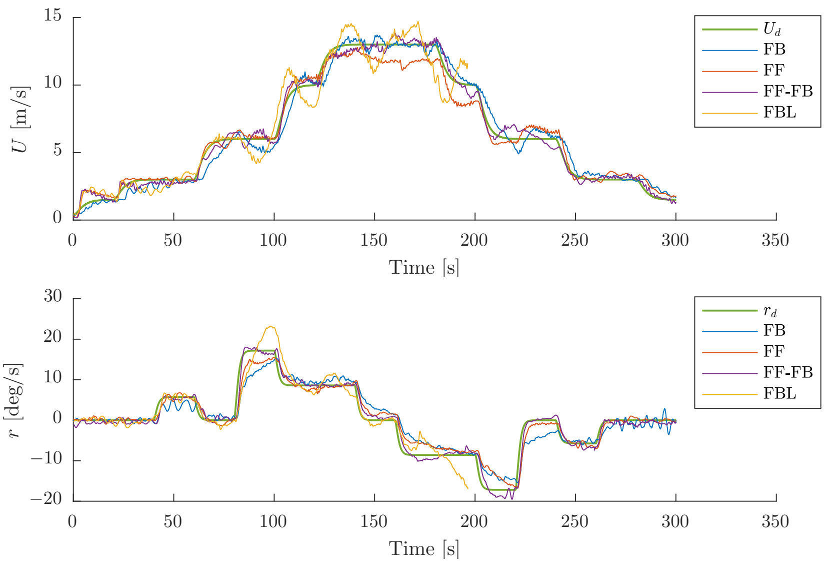
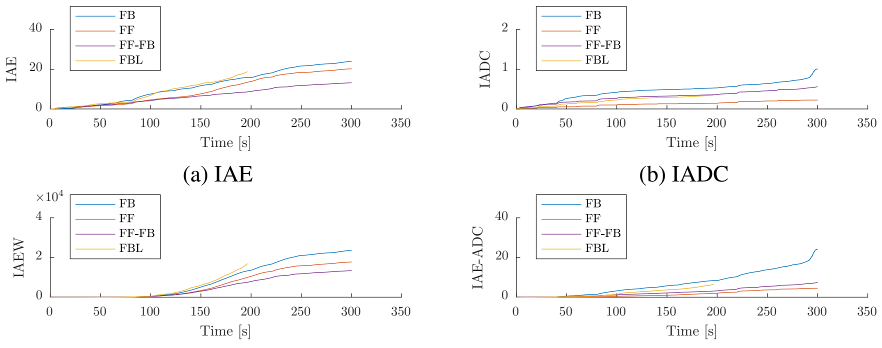


Fig. 13: Test 1 - High-speed trajectory tracking with steady states in slight seas. The feedback linearizing (FBL) controller fails at $t \approx 194$ s due to sensor dropout.



(a) IAE (b) IADC

(c) IAEW (d) IAE-ADC

Fig. 14: Performance metrics for Test 1 in slight seas.

The IAE (Figure 14a) shows that the FF-FB controller has the best control precision, while the FF controller is somewhat better than the FB and FBL controllers. From the IADC (Figure 14b), it is clear that the FB controller is tough on the actuators, while the FBL and FF-FB controllers are comparable. The FF controller is, as expected, the best with respect to wear and tear. From the IAEW (Figure 14c), the FF-FB controller has the best energy efficiency, the FF controller is second best and the FB controller places third. The FBL controller has a bad IAEW due to the oscillatory behavior. The IAE-ADC (Figure 14d) shows the same tendencies as the

IADC, but the FF-FB controller performs better than the FBL controller, and the gap between the FF-FB and FF controllers is smaller.

5.3.2 Test 2 - High-speed trajectory tracking without steady states

The second test was intended to investigate the tracking performance of the controllers. The reference is constantly changing without reaching steady state, and both moderate and high velocities are tested. This test was performed only in slight seas. From Figure 15, we observe that the FB controller again suffers from poor

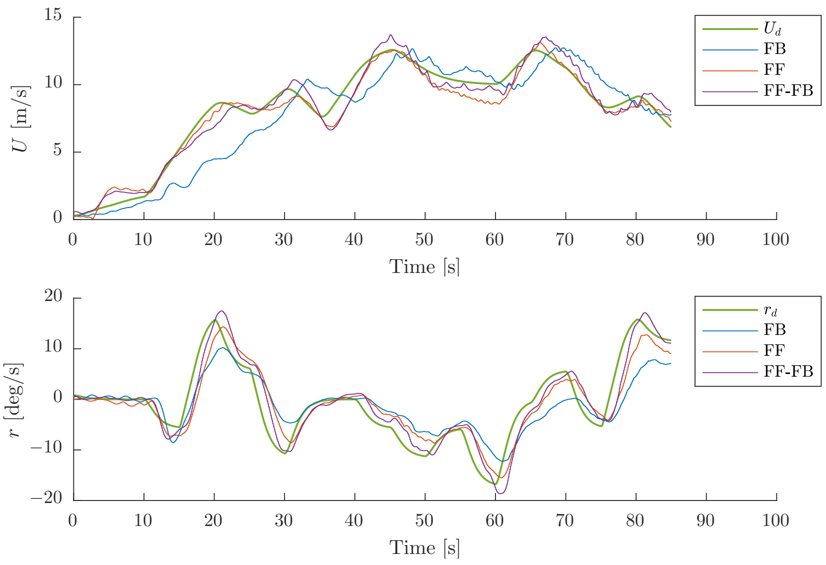


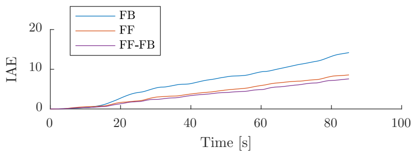
Fig. 15: Test 2 - High speed trajectory tracking without steady states in slight seas. The FF and FF-FB controllers far outperform the FB controller.

tracking and largely fails this test. The FF controller performs remarkably well and, from the time plot, the FF and FF-FB controllers seem to have equal performance.

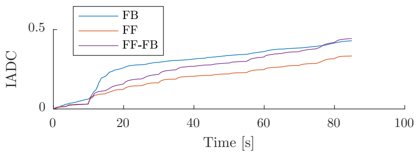
From the performance metrics in Figure 16, the FB controller has the lowest performance, while the FF and FF-FB controllers are quite equal. The FF-FB controller has slightly better control precision (IAE) than the FF controller, at the cost of increased actuator wear and tear (IADC and IAE-ADC).

5.4 *Experiments in calm seas*

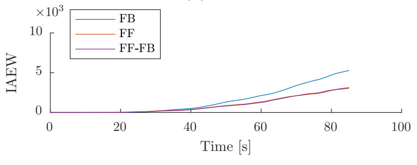
Two of the scenarios were tested in calm seas, and here we present one of them.



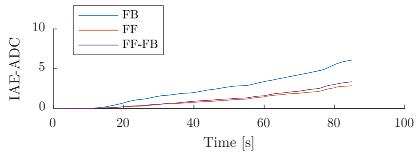
(a) IAE



(b) IADC



(c) IAEW



(d) IAE-ADC

Fig. 16: Performance metrics for Test 2 in slight seas.

5.4.1 Test 3 - Lower-speed trajectory tracking with steady states

The third test was intended to test lower velocities, especially for the yaw rate. The vessel response in calm seas is shown in Figure 17. Note that the integral gain for the yaw rate controller unfortunately was set too high by accident, causing oscillation in the yaw rate.

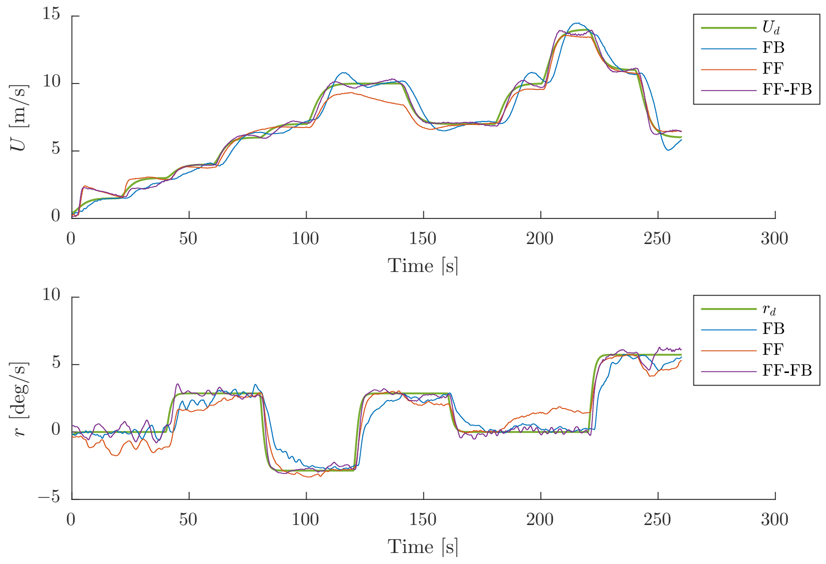
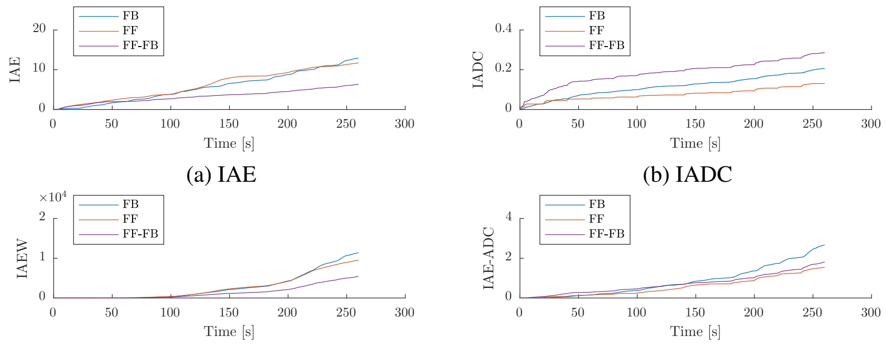


Fig. 17: Test 3 - Lower-speed trajectory tracking with steady states in calm seas. Observe the low amount of noise in the SOG-response compared to Figure 13.

From Figure 17, we observe that the FB controller again suffers from poor tracking, and struggles with steady-state offset in yaw rate (despite the high integrator gain). The FF controller also struggles with steady-state offset, but has superior performance in the transients. The FF-FB controller combines the performance of the FB and FF controllers and provides good tracking and low steady-state offset.



(a) IAE (b) IADC

(c) IAEW (d) IAE-ADC

Fig. 18: Performance metrics for Test 3 in calm seas.

From the performance metrics in Figure 18, we can draw the same conclusions as for Test 1. However, concerning the IADC, the FF-FB controller has the most wear and tear, which is probably caused by the initial oscillatory behavior in yaw rate due to the high integrator gain resulting in high initial condition sensitivity.

5.5 *Motion control experiments summary*

For controller evaluation, it is useful to compare the performance metrics. The final performance metric values for all the tests are presented in Table 4. We observe that:

- The FF and FF-FB controllers have the best performance:
  - The FF controller is best with respect to actuator wear and tear (IADC), also when scaled with the control precision (IAE-ADC).
  - The FF-FB controller is best with respect to control precision (IAE). In all tests except Test 2, it also has the best energy efficiency (IAEW). For Test 2, the FF and FF-FB controllers have near identical energy efficiency.
- The FF-FB controller has the most consistent control precision performance (IAE) for varying environmental conditions.
- The FF and FF-FB controllers have quite similar consistency of energy efficiency (IAEW) for varying environmental conditions.
- The FB controller has the worst metrics in all the tests, except for IADC in Test 2.

Table 4: The performance metrics are normalized for each test, and the controller performing best for each metric in each test is highlighted in bold. C/S refer to calm (C) and slight (S) seas. *For Test 1 in slight seas, the FBL controller did not complete the entire test, hence the S metrics of FBL Test 1 are not comparable.

Test case	**Controller**	**IAE**	**IADC**	**IAE-ADC**	**IAEW**
Test 1	FB	85.8 / 100.0	41.0 / 100.0	35.2 / 100.0	85.6 / 100.0
C/S	FF	71.1 / 84.2	**21.0 / 22.1**	**14.9 / 18.6**	65.6 / 74.9
	FF-FB	**41.5 / 54.9**	55.1 / 55.4	22.9 / 30.5	**42.0 / 56.5**
	FBL*	84.9 / 78.3	45.7 / 33.9	38.8 / 26.5	87.2 / 71.5
Test 2	FB	100.0	96.9	100.0	100.0
S	FF	60.4	**75.3**	**46.9**	**57.5**
	FF-FB	**53.4**	100.0	55.1	59.3
Test 3	FB	88.9 / 100.0	45.2 / 100.0	40.2 / 100.0	89.7 / 100.0
C/S	FF	80.8 / 97.5	**28.7 / 26.4**	**23.2 / 25.8**	75.3 / 83.1
	FF-FB	**43.7 / 45.2**	62.7 / 70.2	27.4 / 31.7	**42.8 / 45.1**

6 Conclusion

In this paper, we have presented a powerful approach to modeling, identification and control of high-speed ASVs operating in the displacement, semi-displacement and planing regions. We have used this approach on a high-speed ASV to successfully identify a control-oriented model of the vessel covering all its operating regions. Furthermore, we have through full-scale motion control experiments compared the performance of four controllers all utilizing the identified model:

- A proportional-integral feedback (FB) controller with gain scheduling.
- A feedforward (FF) controller.
- A combined feedforward and feedback (FF-FB) controller.
- A feedback-linearizing (FBL) controller.

By both qualitative and quantitative comparisons, it is shown that the FF-FB and FF controllers have superior performance over the two others. The FF-FB and FBL controllers are formulated almost identically, but the FF-FB controller has superior robustness and performance over the FBL controller.

From the results, we observe that model-based feedforward control is a powerful tool, which when used correctly will result in outstanding performance. There are, however, pitfalls reducing the robustness with respect to time delays, sensor dropouts and noise. This is the case for the FBL controller, where using the measured vessel velocity in the damping feedforward term causes instability.

Possibilities for further work include:

- Use the SOG and ROT controllers for closed-loop pose control for e.g. path following and target tracking scenarios.
- Use the SOG and ROT controllers and the identified model in combination with the dynamic window algorithm to achieve collision avoidance functionality, continuing the work in [5].

- Use the SOG and ROT controllers for manual velocity control through a joystick.
- Use the modeling approach for automatic and/or recursive model identification.
- Use the identified model to online modify the reference filter acceleration limits.

Acknowledgments

This work was supported by the Research Council of Norway through project number 244116, and through the Centres of Excellence funding scheme with project number 223254. The authors would like to express great gratitude to Maritime Robotics for placing the Telemetron ASV at our disposal, and especially Thomas Ingebretsen for help with implementing the controllers and supporting the experiments. The authors would also like to thank Andreas L. Flåten for valuable discussions on the topic of linear regression and cross-validation.

References

1. Bertram, V. (2008). Unmanned Surface Vehicles - A Survey. Skibsteknisk Selskab, Copenhagen, Denmark
2. Bishop C (2006) Pattern Recognition and Machine Learning. Springer Science + Business Media
3. Breivik M (2010) Topics in Guided Motion Control of Marine Vehicles. PhD thesis, Norwegian University of Science and Technology, Trondheim, Norway
4. Breivik M, Hovstein V E, Fossen T I (2008) Straight-Line Target Tracking for Unmanned Surface Vehicles. Model. Ident. Control. 29:131–149
5. Eriksen B-O H, Breivik M, Pettersen K Y, Wiig M S (2016) A Modified Dynamic Window Algorithm for Horizontal Collision Avoidance for AUVs. Proc. of IEEE CCA. Buenos Aires, Argentina
6. Faltinsen O M (2005) Hydrodynamics of High-Speed Marine Vehicles. Cambridge University Press
7. Fossen T I (2011) Handbook of Marine Craft Hydrodynamics and Motion Control. John Wiley & Sons Ltd
8. Liu Z, Zhang Y, Yu X, Yuan C (2016) Unmanned surface vehicles: An overview of developments and challenges. Annu Rev Control. 41:71–93
9. Prince W G, Bishop R E D (1974) Probabilistic Theory of Ship Dynamics. Chapman and Hall
10. Sonnenburg C R, Woolsey C A (2013) Modeling, Identification, and Control of an Unmanned Surface Vehicle. Journal of Field Robotics. 30(3):371–398.
11. Sonnenburg C R, Gadre A, Horner D, Kragelund S, Marcus A, Stilwell D J Woolsey C A (2010) Control-Oriented Planar Motion Modeling of Unmanned Surface Vehicles. Proc. of OCEANS 2010. Seattle, USA
12. Sørensen M E N, Breivik M (2015) Comparing Nonlinear Adaptive Motion Controllers for Marine Surface Vessels. Proc. of 10th IFAC MCMC. Copenhagen, Denmark
13. Sørensen M E N, Bjørne E S, Breivik M (2016) Performance Comparison of Backstepping-based Adaptive Controllers for Marine Surface Vessels. Proc. of IEEE CCA. Buenos Aires, Argentina

Paper C MPC-based mid-level collision avoidance for ASVs using nonlinear programming

Postprint of **B.-O. H. Eriksen** and M. Breivik, “MPC-based mid-level collision avoidance for ASVs using nonlinear programming”, in *Proc. of the 1st IEEE Conference on Control Technology and Applications (CCTA)*, (Mauna Lani, Hawai’i, USA), 2017, pp. 766–772.

MPC-based Mid-level Collision Avoidance for ASVs using Nonlinear Programming

Bjørn-Olav H. Eriksen, Morten Breivik

***Abstract*— In this paper, we present a mid-level collision avoidance algorithm for autonomous surface vehicles (ASVs) based on model predictive control (MPC) using nonlinear programming. The algorithm enables avoidance of both static and moving obstacles, and following of a desired nominal trajectory if there is no danger of collision. We compare two alternative objective functions, where one is a quadratic function and the other is a nonlinear function designed to produce maneuvers observable for other vessels in compliance with rule 8 of the International Regulations for Preventing Collisions at Sea (COLREGS). The algorithm is implemented in the CASADI framework and uses the IPOPT solver. The performance of the algorithm is evaluated through simulations which show promising results. Furthermore, the algorithm is considered computationally feasible to run in real time. This algorithm serves as a base algorithm for further development in order to ensure full COLREGS compliance.**

I. Introduction

The development and use of autonomous technology in both industry and research is continuously moving forward. In particular, the automotive industry has had a leading role, while the development in the maritime domain has not received similar focus, even though the potential benefits from developing and utilizing autonomous marine technology is great. Employing autonomous surface vehicles (ASVs) for marine operations such as oceanography, bathymetry, passenger and goods transport, marine patrolling, etc. can result in increased safety, widened operational window and reduced costs.

A requirement for employing ASVs in the marine domain is a collision avoidance (COLAV) system. Such a system must be able to plan observable long-term maneuvers and at the same time respond to rapid changes in the environment, such as a nearby high-speed vessel suddenly changing course. Employing observable maneuvers is especially important to establish implicit communication between vessels based on their maneuvers. Performing observable maneuvers is also a requirement of the International Regulations for Preventing Collisions at Sea (COLREGS), which acts as “rules of the road” for marine surface vessels, both manned and unmanned.

COLAV algorithms can in general be divided into reactive and deliberate algorithms. Reactive algorithms utilize currently available sensor data, and employ a minimum of computations. This often produces sub-optimal solutions, but

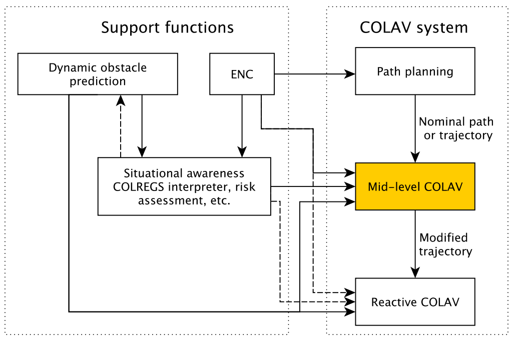

Support functions
COLAV system
Dynamic obstacle prediction
ENC
Path planning
Nominal path or trajectory
Situational awareness COLREGS interpreter, risk assessment, etc.
Mid-level COLAV
Modified trajectory
Reactive COLAV

Fig. 1: Example of a hybrid COLAV architecture with three layers. The middle and top layers are typically deliberate algorithms, while a reactive algorithm is used in the bottom layer. In this architecture, electronic nautical charts (ENC), prediction of moving obstacles and interpretation of COLREGS are handled by separate support functions.

the low computational requirement makes the algorithms capable of responding to rapid changes in the environment. Examples of reactive algorithms include velocity obstacles (VO) [1, 2] and the dynamic window (DW) approach [3–5]. Deliberate algorithms, on the other hand, utilize apriori information (often in the form of aggregated sensor information, maps, etc.) and can encode more sophisticated criteria for the behavior, for instance generating observable maneuvers which are optimal in a global sense. Examples of deliberate algorithms include rapidly-exploring random trees (RRT) [6], graph search algorithms such as A* and D* [7, 8] and constrained nonlinear optimization [9, 10]. A downside of deliberate algorithms is their computational complexity, which results in high computational requirements and possibly challenges for real-time implementation. The solution to this issue is to employ a hybrid architecture, merging reactive and deliberate algorithms [9, 11]. An example of a three-layered hybrid architecture is shown in Fig. 1. In such an architecture, the top layer can perform long-term planning from the start position to the goal position, only considering static obstacles (land, reefs, etc.) while performing optimal planning with respect to arrival time, energy consumption, etc. This would typically be performed offline, but may also be computed online if required. The second layer inputs the desired nominal path or trajectory, and makes local adaptations if necessary. The planning horizon of this layer

Bjørn-Olav H. Eriksen and Morten Breivik are with the Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway. Email: {bjorn-olav.h.eriksen,morten.breivik}@ieee.org

should be long enough to plan avoidance maneuvers, but short enough to ensure computational feasibility. The reactive layer then tries to follow the trajectory specified by the mid-level algorithm, and makes local adaptations if necessary.

In this paper, we focus our attention to deliberate mid-layer COLAV formulated using nonlinear programming (NLP). Constrained optimization in the form of model predictive control (MPC) has already been applied for automotive COLAV [12, 13]. The use of MPC for ASV COLAV has been investigated in [10], however only using brute-force techniques by discretization of the search space in a finite number of control inputs. In this paper, we present an MPC algorithm for mid-level COLAV, implemented using NLP and solved using the IPOPT [14] solver in the CASADI [15] framework. The algorithm is tested with two objective functions, where one generates maneuvers which are compliant with rule 8 of COLREGS.

The paper is structured as follows: Section II presents an ASV model and kinematic constraints. An overview of COLREGS, and some discussion of the most applicable rules to ASVs is given in Section III. In Section IV, we define the mid-level COLAV problem as an optimization problem, while simulation results are shown in Section V. Finally, concluding remarks and possibilities for further work are given in Section VI.

II. ASV MODELING

In the scope of deliberate mid-level COLAV, we propose to use a purely kinematic model. This simplifies the design, since no vehicle kinetic model is required. For underactuated ASVs, a kinematic model can be formulated as [16]:

$$\dot{\boldsymbol{\eta}} = \begin{bmatrix} \cos(\psi) & 0 \\ \sin(\psi) & 0 \\ 0 & 1 \end{bmatrix} \boldsymbol{u}, \tag{1}$$

where $\boldsymbol{\eta} = \begin{bmatrix} N & E & \psi \end{bmatrix}^T \in \mathbb{R}^2 \times S$ is the vehicle pose with N and E representing the north and east position and ψ representing the yaw angle (heading). The vector $\boldsymbol{u} = \begin{bmatrix} U & r \end{bmatrix}^T \in \mathbb{R}^2$ is the vehicle velocity with U and r representing the vehicle speed over ground (SOG) and yaw rate (ROT), respectively. Note that the vehicle is assumed to have zero side-slip, hence the vehicle heading and course are assumed to be aligned. In addition, note that the ocean current is not included in (1).

The argument for neglecting the vehicle kinetics and the ocean current is that the reactive layer will take these into account when following the mid-level trajectory. However, some criteria should be employed to ensure a degree of feasibility with respect to the vessel capabilities. Underactuated ASVs typically employ a rudder for controlling ROT, and uses the forward thrust from the propeller to control SOG. For such a configuration, the actuation moment in yaw is highly dependent on the SOG since the rudder force is dependent on the fluid velocity over the rudder.

In [16], a nonlinear kinetic model for the Maritime Robotics Telemetron ASV, shown in Fig. 2, is identified. The model is of the form:

Fig. 2: The Telemetron ASV. Courtesy of Maritime Robotics.

$$\boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}} + \boldsymbol{\sigma}(\boldsymbol{x}) = \boldsymbol{\tau}, \tag{2}$$

where $\boldsymbol{x} = \begin{bmatrix} U & r \end{bmatrix}^T$, $\boldsymbol{M}(\boldsymbol{x})$ is a velocity-dependent inertia matrix, $\boldsymbol{\sigma}(\boldsymbol{x})$ is a damping vector and $\boldsymbol{\tau}$ is a normalized control input. The inertia matrix and damping vector are formed by a number of polynomial and asymptotic terms [16]. Keeping in mind that the possible SOG and ROT are dependent on each other, we use the model to develop the kinematic limitations:

$$\boldsymbol{x} \in V_s = \left\{ \begin{bmatrix} U & r \end{bmatrix}^T \in \mathbb{R}^2 | U_{\min}(r) \leq U \leq U_{\max}(r) \right. \\ \left. \wedge\, r_{\min} \leq r \leq r_{\max} \right\}, \tag{3}$$

where $r_{\min} < 0$ and $r_{\max} > 0$ are constants, while $U_{\min}(r)$ and $U_{\max}(r)$ are functions of r. We handle the ROT limitation as constant, while letting the SOG limitation capture the dependency between the SOG and ROT limitations.

III. THE INTERNATIONAL REGULATIONS FOR PREVENTING COLLISIONS AT SEA (COLREGS)

COLREGS intends to provide a set of rules which when followed should avoid ship-to-ship collisions. The rules have been revised a number of times, to cope with the advances in technology and the increasing use of the seaways for different activities.

COLREGS is divided in 5 parts, with a total of 38 rules [17]. Part B (Steering and Sailing Rules) considers the rules most relevant for implementing COLAV algorithms. The rules most commonly discussed in the litterature on autonomous COLAV, see e.g. [2], are rules 13-15, which describe the overtaking, head-on, and crossing situations:

Rule 13 Overtaking situation: An overtaking situation occurs when a vessel is approaching another vessel of more than 22.5° abaft her beam. A vessel overtaking is required to keep clear of the vessel being overtaken, such that risk of collision is avoided.

Rule 14 Head on: A head-on situation occurs when two vessels are approaching at reciprocal (or nearly reciprocal) courses. This is usually interpreted as a relative bearing of 180° ± 6°. In such a situation, both vessels are required to do a port turn to avoid collision.

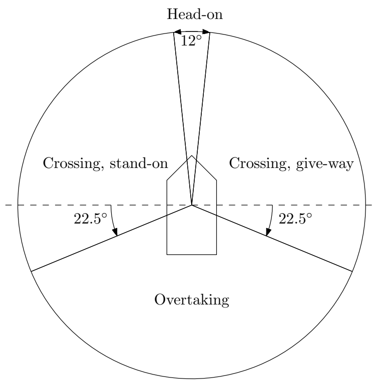


Fig. 3: Graphical interpretation of COLREGS situations.

Rule 15 Crossing: A crossing situation occurs when a vessel is approaching with a relative bearing of between 25° abaft her beam and 6° abaft her bow. In such a situation, the vessel having the other vessel on her starboard side is required to give way to avoid collision. If possible, one should avoid passing in front of the other vessel.

A graphical interpretation of the COLREGS situations is shown in Fig. 3. Formally, the vessel required to avoid collision is named the give-way vessel while the other vessel is named the stand-on vessel.

In addition to rules 13-15, rules 8, 16 and 17 are quite relevant for autonomous COLAV:

Rule 8 Action to avoid collision: This rule requires that actions taken to avoid collision should be large enough to be observable for other ships. An implication for this is that a series of small alterations of course and/or speed should not be applied. If there is sufficient space available, alterations of course should be the preferred action to avoid collision.

Rule 16 Action by give-way vessel: This rule dictates that the give-way vessel should, if possible, take early and substantial action to keep well clear of the stand-on vessel.

Rule 17 Action by stand-on vessel: This rule dictates that the stand-on vessel should keep her speed and course. However, if it is apparent that the give-way vessel is not taking action to avoid collision, the stand-on vessel is required to apply actions to best avoid collision.

Rule 8 is considered in [10] by limiting the possible actions, resulting in distinct observable actions. These rules are, however, seldom considered in the literature. It should be noted that rule 17 actually requires the stand-on vessel to not attempt to avoid collision, before it is apparent that the give-way vessel is not acting to avoid collision. By applying a hybrid architecture, rule 17 can be handled implicitly by implementing the latter part in the reactive algorithm.

IV. MPC-based mid-level COLAV

This section describes the parameterization and implementation of the mid-level COLAV algorithm.

A. Control objective

For the mid-level COLAV algorithm, we want to input a desired nominal path or trajectory which should be followed if there is no danger of collision. A trajectory can for example be a sequence of waypoints together with a desired speed. By assuming that we wish to follow straight lines between the waypoints, we can define a desired nominal trajectory:

$$\boldsymbol{p}_d(t) = \begin{bmatrix} N_d(t) \\ E_d(t) \end{bmatrix}. \tag{4}$$

Consequently, we can formulate the control objective as that the vessel trajectory $\boldsymbol{p}(t) = \begin{bmatrix} N(t) & E(t) \end{bmatrix}^T$ should converge towards $\boldsymbol{p}_d(t)$, while avoiding collisions and obeying COLREGS.

B. Obstacle modeling

A common simplification is to model both moving and static obstacles as circles. We define a static obstacle using a center position and a radius as $O_{s_i} = (\boldsymbol{p}_{s_i}, r_{s_i}) \in \mathbb{R}^2 \times \mathbb{R}^+$. A set of S static obstacles can then be defined as $\mathcal{O}_s = \{O_{s_1}, O_{s_2}, \dots O_{s_S}\}$.

Similarly, a moving obstacle can be defined by a time-varying center position and a radius as $O_{m_i}(t) = (\boldsymbol{p}_{m_i}(t), r_{m_i}) \in \mathbb{R}^2 \times \mathbb{R}^+$, where $\boldsymbol{p}_{m_i} : \mathbb{R} \to \mathbb{R}^2$. A set of M moving obstacles is then defined as $\mathcal{O}_m = \{O_{m_1}(t), O_{m_2}(t), \dots O_{m_M}(t)\}$.

C. Optimization problem construction

To solve the mid-level COLAV problem in Section IV-A as an optimal control problem (OCP), we first define the general OCP:

$$\begin{aligned} \text{minimize} \quad & \phi(\boldsymbol{\eta}(t), \boldsymbol{u}(t)) \\ \text{subject to} \quad & \dot{\boldsymbol{\eta}}(t) = \boldsymbol{F}(\boldsymbol{\eta}(t), \boldsymbol{u}(t)) \\ & \boldsymbol{h}(\boldsymbol{\eta}(t), \boldsymbol{u}(t)) \leq \boldsymbol{0}, \\ & \boldsymbol{\eta}(t_0) = \bar{\boldsymbol{\eta}}_0, \end{aligned} \tag{5}$$

where $\phi : (\mathbb{R}^2 \times S) \times \mathbb{R}^2 \to \mathbb{R}$ is the objective function, $\boldsymbol{\eta}(t)$ is the vehicle pose trajectory, $\boldsymbol{u}(t)$ is the control input trajectory and $\boldsymbol{F}(\boldsymbol{\eta}(t), \boldsymbol{u}(t))$ denotes the kinematic model (1). The function $\boldsymbol{h} : (\mathbb{R}^2 \times S) \times \mathbb{R}^2 \to \mathbb{R}^{n_h}$ forms n_h inequality constraints and $\bar{\boldsymbol{\eta}}_0 \in \mathbb{R}^2 \times S$ is the initial vehicle state.

Even though the continuous OCP in some cases is possible to solve e.g. by using Pontryagin's maximum principle, it is generally practical to define a nonlinear program (NLP) by discretizing (5). The discretized OCP can be formulated using a number of techniques. We choose to use direct multiple shooting, where both the state and control inputs are explicitly defined as decision variables. The NLP with Np prediction steps is:

$$\begin{aligned} \underset{\boldsymbol{w}}{\text{minimize}} \quad & \phi(\boldsymbol{w}) \\ \text{subject to} \quad & \boldsymbol{g}(\boldsymbol{w}) = \boldsymbol{0} \\ & \boldsymbol{h}(\boldsymbol{w}) \leq \boldsymbol{0}, \end{aligned} \tag{6}$$

where $\boldsymbol{w} = \begin{bmatrix}\boldsymbol{\eta}_0^T & \boldsymbol{u}_0^T & \dots & \boldsymbol{\eta}_{Np-1}^T & \boldsymbol{u}_{Np-1}^T & \boldsymbol{\eta}_{Np}^T\end{bmatrix}^T \in \mathbb{R}^{5Np+3}$ is a vector of $5Np+3$ decision variables, and $\boldsymbol{g}(\boldsymbol{w}) \in \mathbb{R}^{n_g}$ is a vector of n_g equality constraints.

We define a quadratic objective function:

$$\phi_1(\boldsymbol{w}, \boldsymbol{p}_{d_{1:Np}}) = \sum_{k=0}^{Np-1} \Big(K_p \left\| \boldsymbol{p}_{k+1} - \boldsymbol{p}_{d_{k+1}} \right\|_2^2 + K_U(U_k - U_d)^2 + K_r(r_k - r_d)^2 \Big), \quad (7)$$

where $K_p, K_U, K_r > 0$ are tuning parameters. The desired SOG U_d and ROT r_d can be derived from the desired nominal path, given by the sequence of desired positions $\boldsymbol{p}_{d_{1:Np}} = \begin{bmatrix}\boldsymbol{p}_{d_1} & \boldsymbol{p}_{d_2} & \cdots & \boldsymbol{p}_{d_{Np}}\end{bmatrix}$.

When using multiple shooting, one must employ shooting constraints to ensure that the control input and vehicle states satisfy the kinematic model (1). We define an integrator function $\boldsymbol{f}(\boldsymbol{\eta}_k, \boldsymbol{u}_k)$ using Runge-Kutta of order 4:

$$\begin{aligned}
\boldsymbol{k}_1 &= \boldsymbol{F}(\boldsymbol{\eta}_k, \boldsymbol{u}_k) \\
\boldsymbol{k}_2 &= \boldsymbol{F}(\boldsymbol{\eta}_k + \frac{h}{2}\boldsymbol{k}_1, \boldsymbol{u}_k) \\
\boldsymbol{k}_3 &= \boldsymbol{F}(\boldsymbol{\eta}_k + \frac{h}{2}\boldsymbol{k}_2, \boldsymbol{u}_k) \qquad (8) \\
\boldsymbol{k}_4 &= \boldsymbol{F}(\boldsymbol{\eta}_k + h\boldsymbol{k}_3, \boldsymbol{u}_k) \\
\boldsymbol{f}(\boldsymbol{\eta}_k, \boldsymbol{u}_k) &= \boldsymbol{\eta}_k + \frac{h}{6}\left(\boldsymbol{k}_1 + \boldsymbol{2}k_2 + \boldsymbol{2}k_3 + \boldsymbol{k}_4\right),
\end{aligned}$$

where h is the discretization time step. Given a state and control input $\boldsymbol{\eta}_k$ and $\boldsymbol{u}_k$, we can now define the vessel state at the next iteration as $\boldsymbol{\eta}_{k+1} = \boldsymbol{f}(\boldsymbol{\eta}_k, \boldsymbol{u}_k)$. We include the shooting constraints in the equality constraints $\boldsymbol{g}(\boldsymbol{w})$ as:

$$\boldsymbol{g}(\boldsymbol{w}) = \begin{bmatrix} \bar{\boldsymbol{\eta}}_0 - \boldsymbol{\eta}_0 \\ \boldsymbol{f}(\boldsymbol{\eta}_0, \boldsymbol{u}_0) - \boldsymbol{\eta}_1 \\ \boldsymbol{f}(\boldsymbol{\eta}_1, \boldsymbol{u}_1) - \boldsymbol{\eta}_2 \\ \vdots \\ \boldsymbol{f}(\boldsymbol{\eta}_{Np-1}, \boldsymbol{u}_{Np-1}) - \boldsymbol{\eta}_{Np} \end{bmatrix}, \qquad (9)$$

resulting in $n_g = 3(Np+1)$ equality constraints.

To guarantee that the resulting vehicle trajectory $\boldsymbol{p}(t)$ avoids collisions, we must make sure that the vehicle trajectory $\boldsymbol{p}(t)$ does not intersect the obstacles in the sets $\mathcal{O}_s$ and $\mathcal{O}_m$. In the set $\mathcal{O}_s$, we have S static obstacles. For each obstacle $O_{s_i} = (\boldsymbol{p}_{s_i}, r_{s_i}) \in \mathcal{O}_s$ we define the function:

$$\boldsymbol{h}_{s_i}(\boldsymbol{p}_{1:Np}) = \begin{bmatrix} r_{s_i}^2 - \left\|\boldsymbol{p}_1 - \boldsymbol{p}_{s_i}\right\|_2^2 \\ r_{s_i}^2 - \left\|\boldsymbol{p}_2 - \boldsymbol{p}_{s_i}\right\|_2^2 \\ \vdots \\ r_{s_i}^2 - \left\|\boldsymbol{p}_{Np} - \boldsymbol{p}_{s_i}\right\|_2^2 \end{bmatrix} \in \mathbb{R}^{Np}. \qquad (10)$$

Similarly, for each moving obstacle $O_{m_i} = (\boldsymbol{p}_{m_i}(t), r_{m_i}) \in \mathcal{O}_m$ we define the function:

$$\boldsymbol{h}_{m_i}(\boldsymbol{p}_{1:Np}, \boldsymbol{t}_{1:Np}) = \begin{bmatrix} r_{m_i}^2 - \left\|\boldsymbol{p}_1 - \boldsymbol{p}_{m_i}(t_1)\right\|_2^2 \\ r_{m_i}^2 - \left\|\boldsymbol{p}_2 - \boldsymbol{p}_{m_i}(t_2)\right\|_2^2 \\ \vdots \\ r_{m_i}^2 - \left\|\boldsymbol{p}_{Np} - \boldsymbol{p}_{m_i}(t_{Np})\right\|_2^2 \end{bmatrix} \in \mathbb{R}^{Np}, \qquad (11)$$

where $\boldsymbol{t}_{1:Np} = \begin{bmatrix} t_1 & t_2 & \dots & t_{Np}\end{bmatrix}^T$ contains the time related to each NLP step. We then define the inequality constraints:

$$\boldsymbol{h}_o(\boldsymbol{w}, \boldsymbol{t}_{1:Np}) = \begin{bmatrix} \boldsymbol{h}_{s_1}(\boldsymbol{p}_{1:Np}) \\ \vdots \\ \boldsymbol{h}_{s_S}(\boldsymbol{p}_{1:Np}) \\ \boldsymbol{h}_{m_1}(\boldsymbol{p}_{1:Np}, \boldsymbol{t}_{1:Np}) \\ \vdots \\ \boldsymbol{h}_{m_M}(\boldsymbol{p}_{1:Np}, \boldsymbol{t}_{1:Np}) \end{bmatrix} \in \mathbb{R}^{(S+M)Np}, \qquad (12)$$

which ensure that the resulting trajectory avoids obstacles.

To ensure compliance with the velocity limitations in (3), we can for each control input $\boldsymbol{u}_i$ define the function:

$$\boldsymbol{h}_{u_i}(\boldsymbol{u}_i) = \begin{bmatrix} U_{\min}(r_i) - U_i \\ -(U_{\max}(r_i) - U_i) \\ r_{\min} - r_i \\ -(r_{\max} - r_i) \end{bmatrix} \in \mathbb{R}^4, \qquad (13)$$

and form the inequality constraints:

$$\boldsymbol{h}_u(\boldsymbol{w}) = \begin{bmatrix} \boldsymbol{h}_{u_0}(\boldsymbol{u}_0) \\ \boldsymbol{h}_{u_1}(\boldsymbol{u}_1) \\ \vdots \\ \boldsymbol{h}_{u_{Np-1}}(\boldsymbol{u}_{Np-1}) \end{bmatrix} \in \mathbb{R}^{4Np}, \qquad (14)$$

which ensure that all control inputs satisfy (3).

Finally, (12) and (14) is combined:

$$\boldsymbol{h}(\boldsymbol{w}, \boldsymbol{t}_{1:Np}) = \begin{bmatrix} \boldsymbol{h}_o(\boldsymbol{w}, \boldsymbol{t}_{1:Np}) \\ \boldsymbol{h}_u(\boldsymbol{w}) \end{bmatrix}, \qquad (15)$$

resulting in $n_h = (4+S+M)Np$ inequality constraints. By requiring the functions $U_{\min}(r)$ and $U_{\max}(r)$ to be C^2 in r, we see that $\boldsymbol{h}(\boldsymbol{w}, \boldsymbol{t}_{1:Np})$ is C^2 in $\boldsymbol{w}$.

D. COLREGS compliance

The NLP formulation (6), (7), (9) and (15) will be able to avoid collision with both moving and static obstacles, but will not necessarily obey the COLREGS rules formulated in Section III.

Implementing these rules in an NLP problem is a challenging task, and in this paper we will only investigate the possibility of obeying rule 8, which requires that the vessel maneuvers should be large enough to be observable to other vessels. This implies that applying a sequence of small alterations in course or speed should be avoided. Naturally, there are a number of ways to avoid this, including:

- Constraining the ROT and SOG derivatives to only allow values with magnitude very close to zero or greater than some threshold.
- Including penalty terms in the objective function which penalize course and speed alteration depending on the time to perform it. Slow alterations would then be penalized more than quick alterations.

Initially, it may seem like a good solution to constrain the ROT and SOG derivatives to only allow maneuvers of a given magnitude, avoiding small alterations. However,

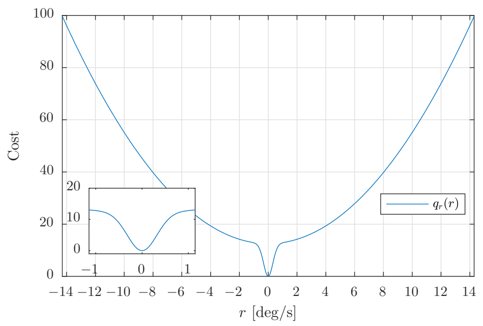


Fig. 4: ROT penalty function with $a_r = 112$, $b_r = 6.25{\cdot}10^{-5}$ and $r_{\max} = 0.25$ rad/s ≈ 14.3 deg/s. The function is non-convex close to the origin, but remains smooth as the zoomed insert shows.

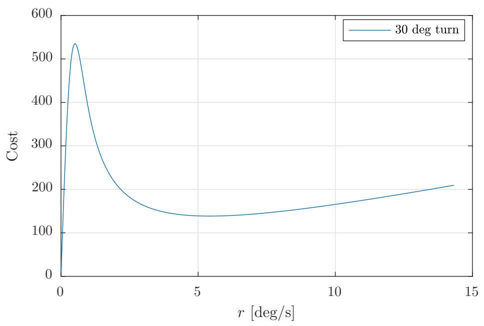


Fig. 5: Cost of performing a 30 deg course alteration, using different ROTs. The peak at $r \approx 0.51$ deg/s is the highest cost, while $r = 5.42$ deg/s is the minimum after the peak.

this would result in a highly non-convex (and even non-connected) search space. Solving an NLP of this form is difficult, and highly dependent on initialization. We therefore investigate the second option with introducing penalty terms in the objective function. Such penalty terms can either be defined on the change of course and SOG $\Delta\psi_k = \psi_{k+1} - \psi_k$ and $\Delta U_k = U_{k+1} - U_k$, for $k = 0 \dots Np-1$, or the ROT and SOG-derivative. Defining the terms on the ROT and SOG-derivative allows for changing the time step length h, without changing the behavior of the penalty terms. Since the SOG-derivative is not part of the NLP (6), we approximate it as $\dot{U}_k \approx \frac{U_{k+1} - U_k}{h}$.

We wish to ensure that the objective function is C^2, hence special considerations must be made when designing the penalty terms. Inspired by the normal distribution, we propose to combine a square and an exponential term as:

$$q(\zeta; a, b) = a\zeta^2 + (1 - \mathrm{e}^{-\frac{\zeta^2}{b}}), \tag{16}$$

where $a, b > 0$ control the shape of the function. We define two functions:

$$\begin{aligned} q_r(r; r_{\max}) &= \frac{100}{q(r_{\max}; a_r, b_r)} q(r; a_r, b_r) \\ q_{\dot{U}}(\dot{U}; \dot{U}_{\max}) &= \frac{100}{q(\dot{U}_{\max}; a_{\dot{U}}, b_{\dot{U}})} q(\dot{U}; a_{\dot{U}}, b_{\dot{U}}), \end{aligned} \tag{17}$$

where the parameters a_r and b_r define the shape of the ROT penalty function $q_r(r)$, and the parameters $a_{\dot{U}}$ and $b_{\dot{U}}$ define the shape of the SOG-derivative penalty term. The parameters $r_{\max} > 0$ and $\dot{U}_{\max} > 0$ are the maximum expected values of r and $\dot{U}$, and are used to scale the functions such that $q_r(r), q_{\dot{U}}(\dot{U}) \in [0, 100]$ for $r \in [-r_{\max}, r_{\max}]$ and $\dot{U} \in [-\dot{U}_{\max}, \dot{U}_{\max}]$.

The ROT penalty function $q_r(r)$ with $a_r = 112$, $b_r = 6.25 \cdot 10^{-5}$ and $r_{\max} = 0.25$ rad/s is shown in Fig. 4. It is obvious that the function is non-convex, but it is C^∞.

Assuming that a course alteration is performed with a constant ROT, the cost of performing a course alteration $\Delta\psi$ using the penalty term can be approximated as $q_r(r)\frac{\Delta\psi}{r}$, where $\frac{\Delta\psi}{r}$ is the time to execute the turn. From Fig. 5 it is clear that for a course alteration of 30 deg, the penalty term motivates choosing a ROT either significantly higher than 0.51 deg/s, or using very long time to execute the turn. Notice that this generalizes to an arbitrary turn, since the cost scales linearly with $\Delta\psi$.

To avoid that the position error in (7) dominates at large errors, we introduce the pseudo-Huber cost function:

$$C(x; \delta) = 2\delta^2 \left(\sqrt{1 + \left(\frac{x}{\delta}\right)^2} - 1 \right), \tag{18}$$

which for small x approximates a quadratic function x^2, and for large x approximates a linear function with slope 2δ [18]. The parameter $\delta > 0$ controls where the function changes from being quadratic to being linear.

Replacing the squared 2-norm position error with the pseudo-Huber cost function and the quadratic ROT and SOG cost with the new penalty terms we get the modified objective function:

$$\begin{aligned} \phi_2(\boldsymbol{w}, \boldsymbol{p}_{d_{1:Np}}) = \sum_{k=0}^{Np-1} \Bigg(\frac{K_p}{2} C \left(\left\| \boldsymbol{p}_{k+1} - \boldsymbol{p}_{d_{k+1}} \right\|_2 ; \delta \right) \\ + K_{\dot{U}} q_{\dot{U}}(\dot{U}_k) + K_r q_r(r_k) \Bigg). \end{aligned} \tag{19}$$

Notice that the desired SOG and ROT are implicitly included in the position error term. We select $\delta = 1$ and scale the position error by $1/2$ to get a slope of 1 for large position errors. This can, however, also be treated as a tuning parameter in the problem.

V. Simulation Results

We simulate a scenario with two static and one moving obstacle to show the resulting behavior of the mid-level COLAV algorithm, given the two objective functions (7) and (19). The mid-level COLAV algorithm is implemented as an MPC algorithm, where only the first step of the optimal solution is implemented.

To define and solve the NLP (6) with (9), (15), and the quadratic and modified objective functions (7) and (19), we

TABLE I: Simulation and tuning parameters.

Param.	Value	Comment
Ns	55	Number of simulation steps
h	10 s	Step size
Np	24	Prediction steps
$U_{\min}$	0 m/s	Minimum SOG
$U_{\max}$	17 m/s	Maximum SOG
Quadratic objective function ϕ_1:		
K_p:	$2/3 \cdot 10^{-4}$ 1/m^2	Position error scaling
K_U:	$1 \cdot 10^{-1}$ s^2/m^2	SOG error scaling
K_r:	$3 \cdot 10^{2}$ 1/rad^2	ROT error scaling
Modified objective function ϕ_2:		
K_p	$4 \cdot 10^{-2}$	Position error scaling
$K_{\dot{U}}$	$6 \cdot 10^{-1}$	SOG-derivative penalty term scaling
K_r:	$5 \cdot 10^{-1}$	ROT penalty term scaling
$[a_{\dot{U}}, b_{\dot{U}}]$	$[8, 2.5 \cdot 10^{-4}]$	SOG-derivative penalty term parameters
$[a_r, b_r]$	$[112, 6.25 \cdot 10^{-5}]$	ROT penalty term parameters

used the CASADI framework (v3.1.0-rc1 for MATLAB) [15] with the IPOPT solver [14]. From the parameterization in Section IV is is clear that the NLP problem is non-convex. IPOPT is however able to handle non-convex optimization.

Simulation and tuning parameters are shown in Table I. The desired nominal trajectory $\boldsymbol{p}_d(t)$ is generated by a set of waypoints and a desired speed along the path $U_d = 10$ m/s. We start the simulation with $\boldsymbol{\eta}_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$, and initialize the first iteration of the MPC with the desired nominal trajectory, and SOG and ROT as zero. Hence:

$$\boldsymbol{w}_0 = \begin{bmatrix} \boldsymbol{\eta}_0^T & \boldsymbol{u}_0^T & \boldsymbol{p}_{d_1}^T & \dots & \boldsymbol{u}_0^T & \boldsymbol{p}_{d_{Np}}^T \end{bmatrix}^T, \tag{20}$$

with $\boldsymbol{u}_0 = \begin{bmatrix} U_d & 0 \end{bmatrix}^T$. Notice that the initial guess is infeasible if the desired nominal trajectory intersects with obstacles. However, IPOPT does not require a feasible initial guess. Later iterations of the MPC is initialized with the solution of the previous iteration.

The simulations are run in MATLAB R2016b, on a 2.8 GHz Intel Core i7 processor. The first iteration of the MPC with the quadratic and modified objective functions takes approximately 250 ms and 750 ms to solve, respectively. Later iterations with warm start take approximately 15–25 ms for the quadratic objective function, and 20–150 ms for the modified objective function. Hence, with a time step of 10 s, we consider the algorithm with the modified objective function as implementable in real time. The NLP with the modified objective function has more local minima than the NLP with the quadratic objective function, which is why the runtime with the modified objective function is less consistent than when using the quadratic objective function. Guaranteeing a maximum computational time for NLPs is difficult, but in the event that the optimization problem is not solved within the required time, the reactive COLAV layer would still keep the vessel on a collision-free path, although possibly getting stuck in a local minima.

Figures 6 and 7 show the resulting vessel trajectory using the quadratic and modified objective functions, respectively. The vessel trajectory is shown in red, while the colored

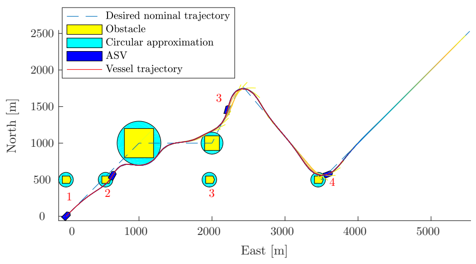


Fig. 6: Simulation result using the quadratic objective function ϕ_1. Note that the size of the ownship (in blue) is overexaggerated for visibility. Hence, what appears to be a collision with the moving obstacle is in fact not a collision.

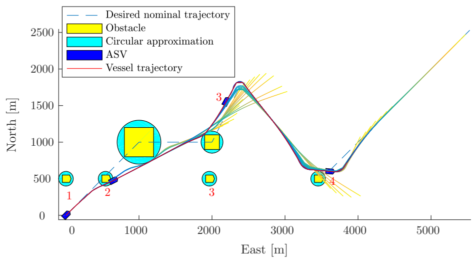


Fig. 7: Simulation result using the modified objective function ϕ_2. Note that the size of the ownship (in blue) is overexaggerated for visibility. Hence, what appears to be a collision with the moving obstacle is in fact not a collision.

lines show the predicted trajectory in each iteration with blue and yellow in the start and end of the predicted trajectory, respectively. Furthermore, the numbers 1–4 mark time instants, where $t_1 = 0$ s, $t_2 = 80$ s, $t_3 = 290$ s and $t_4 = 510$ s. We clearly see that the modified objective function results in a trajectory with clear and observable course changes in contrast to the quadratic objective function. This is also confirmed from the ROT in Fig. 8 which shows the input sequences. We also see that the modified objective function produces a more observable SOG trajectory than the quadratic objective function.

Both objective functions produce collision-free trajectories, and avoid the first encounter with the moving obstacle by passing in front of it at $t \approx 80$ s. An interesting observation is that with respect to COLREGS, our vessel should actually keep the course and speed constant here, and not attempt to avoid collision at an early stage, since our vessel is deemed the stand-on vessel in this situation. In the second encounter with the moving obstacle at $t \approx 500$ s, both algorithms turn to port to avoid collision.

VI. Conclusion and Further Work

We have designed and implemented an NLP-based mid-level COLAV algorithm, enabling avoidance of both static and moving obstacles. The algorithm is simulated with two

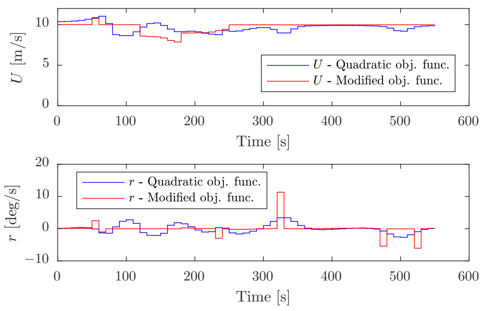


Fig. 8: SOG and ROT trajectories when using the quadratic and modified objective functions.

different objective functions, where one is designed to ensure compliance with rule 8 of COLREGS, thus providing observable course and speed changes. The simulation results are promising and motivate for further development to include other essential COLREGS rules. Based on the simulations, the NLP is considered to be real-time implementable with respect to computational requirements.

Possibilities for further work include:

- Designing objective function terms able to capture rules 13-15 and 17 of COLREGS. These terms can be aided by a support function, see Fig. 1, to encode which rules are applicable to each vessel in a given situation.
- Combining the developed mid-level COLAV algorithm with a reactive algorithm, e.g. continuing the work on the dynamic window algorithm in [5].

ACKNOWLEDGMENT

This work was supported by the Research Council of Norway through project number 244116 and the Centres of Excellence funding scheme with project number 223254.

REFERENCES

[1] P. Fiorini and Z. Shiller, “Motion planning in dynamic environments using velocity obstacles”, *International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[2] Y. Kuwata, M. T. Wolf, D. Zarzhitsky, and T. L. Huntsberger, “Safe maritime autonomous navigation with COLREGS, using velocity obstacles”, *IEEE J. Oceanic Eng.*, vol. 39, no. 1, pp. 110–119, 2014.

[3] D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance”, *IEEE Robotics & Automation Magazine*, vol. 4, pp. 23–33, 1997.

[4] P. Ögren and N. Leonard, “A convergent dynamic window approach to obstacle avoidance”, *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 188–195, 2005.

[5] B.-O. H. Eriksen, M. Breivik, K. Y. Pettersen, and M. S. Wiig, “A modified dynamic window algorithm for horizontal collision avoidance for AUVs”, in *Proc. of IEEE CCA*, Buenos Aires, Argentina, 2016.

[6] S. M. LaValle, “Rapidly-exploring random trees: A new tool for path planning”, Computer Science Dept., Iowa State University, Tech. Rep., 1998.

[7] P. Hart, N. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths”, *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[8] A. Stentz, “The focussed D* algorithm for real-time replanning”, in *Proc. of the International Joint Conference on Artificial Intelligence*, Pittsburgh, Pennsylvania, USA, 1995.

[9] Ø. A. G. Loe, “Collision avoidance for unmanned surface vehicles”, Master’s thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2008.

[10] T. A. Johansen, T. Perez, and A. Cristofaro, “Ship collision avoidance and COLREGS compliance using simulation-based control behavior selection with predictive hazard assessment”, *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3407–3422, 2016.

[11] G. Casalino, A. Turetta, and E. Simetti, “A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields”, in *Proc. of Oceans 2009-Europe*, Bremen, Germany, 2009.

[12] T. Shim, G. Adireddy, and H. Yuan, “Autonomous vehicle collision avoidance system using path planning and model-predictive-control-based active front steering and wheel torque control”, *Proceedings of the Institution of Mechanical Engineers, part D: Journal of Automobile Engineering*, vol. 226, no. 6, pp. 767–778, 2012.

[13] B. Yi, S. Gottschling, J. Ferdinand, N. Simm, F. Bonarens, and C. Stiller, “Real time integrated vehicle dynamics control and trajectory planning with MPC for critical maneuvers”, in *IEEE Intelligent Vehicles Symposium*, Gothenburg, Sweden, 2016.

[14] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming”, *Mathematical Programming*, vol. 106, pp. 25–57, 2005.

[15] J. Andersson, “A general-purpose software framework for dynamic optimization”, PhD thesis, Arenberg Doctoral School, KU Leuven, Heverlee, Belgium, 2013.

[16] B.-O. H. Eriksen and M. Breivik, “Modeling, identification and control of high-speed asvs: Theory and experiments”, in *Sensing and control for autonomous vehicles: Applications to land, water and air vehicles*, T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer, Eds. Cham: Springer International Publishing, 2017, pp. 407–431.

[17] A. N. Cockcroft and J. N. F. Lameijer, *A guide to the collision avoidance rules*. Elsevier, 2004.

[18] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

Paper D Radar-based maritime collision avoidance using dynamic window

Postprint of **B.-O. H. Eriksen**, E. F. Wilthil, A. L. Flåten, E. F. Brekke, and M. Breivik, “Radar-based maritime collision avoidance using dynamic window”, in *Proc. of the 2018 IEEE Aerospace Conference*, (Big Sky, Montana, USA), 2018, pp. 1–9.

Radar-based Maritime Collision Avoidance using Dynamic Window

Bjørn-Olav H. Eriksen **Erik F. Wilthil** **Andreas L. Flåten** **Edmund F. Brekke** **Morten Breivik**

Centre for Autonomous Marine Operations and Systems
Department of Engineering Cybernetics
Norwegian University of Science and Technology (NTNU)
Trondheim, Norway
{bjorn-olav.h.eriksen, morten.breivik}@ieee.org
{erik.wilthil, andreas.flaten, edmund.brekke}@ntnu.no

Abstract—**Collision avoidance systems are a key ingredient in developing autonomous surface vehicles (ASVs). Such systems require real-time information about the environment, which can be obtained from transponder-based systems or exteroceptive sensors located on the ASV. In this paper, we present a closed-loop collision avoidance (COLAV) system using a maritime radar for detecting target ships, implemented on a 26 foot high-speed ASV. The system was validated in full-scale experiments in Trondheimsfjorden, Norway, in May 2017. The probabilistic data association filter (PDAF) is used for tracking target vessels. The output from the PDAF is processed through a least-squares retrodiction procedure in order to provide the COLAV system with sufficiently accurate course estimates. A tracking interface provides estimates of target states to the COLAV system, which is based on the dynamic window (DW) algorithm. DW is a reactive COLAV algorithm originally designed for ground vehicles, and we therefore make a number of modifications to adapt it for use with high-speed ASVs. The closed-loop experiments demonstrated successful COLAV with this system, but also disclosed several challenges arising from both the DW algorithm and the tracking system, motivating for further work.**

TABLE OF CONTENTS

1. INTRODUCTION

The cost efficiency and safety of marine operations such as seabed surveying, surveillance, passenger and goods transport have potential for improvement by moving in the direction of more automatic and autonomous operations. An enabling technology for this to happen is autonomous collision avoidance (COLAV). For autonomous surface vehicles (ASVs), COLAV is a complex task involving challenges with perception, planning and regulations.

COLAV algorithms may in general be divided into reactive and deliberate approaches. Reactive algorithms consider a limited amount of information, often just currently available sensor data, making the algorithms computationally cheap at the cost of possibly producing sub-optimal behavior. The dynamic window (DW) algorithm [13], [10] and the velocity obstacle (VO) algorithm [16] are examples of reactive COLAV algorithms. Deliberate algorithms usually use greater amounts of information and plan in a longer temporal setting, making the algorithms more optimal in a global sense at the cost of increased computational requirements. To be able to both react to sudden changes in the environment and performing meaningful long-term maneuvers, reactive and deliberate algorithms can be combined in a hybrid architecture [18].

ASVs are in general required to follow the International Regulations for Preventing Collisions at Sea (COLREGS), which act as “rules of the road” for avoiding collisions at sea [4]. Reactive COLAV is intended as a “last line of defense” in close-quarter situations, where deliberate algorithms fail in resolving the situation. In such situations, it may be necessary to violate COLREGS to avoid collision. In fact, COLREGS require a vessel to violate the rules if collision cannot be avoided without violating the rules. Hence, we do not consider COLREGS in these experiments.

For perception, a transponder-based communication system may be used. An example of such a system is the automatic identification system (AIS), which is used to transmit the position and velocity of a vessel to other vessels. Passenger ships and vessels with gross tonnage over 300 are obliged to carry AIS transponders. AIS is being used for navigation at sea, and can obviously provide useful information about other vessels’ whereabouts and intentions to a COLAV system. However, since AIS depends on satellite navigation and data input from the user, it may contain inaccurate or invalid data [14]. The information is also restricted to vessels equipped with AIS transponders. Although lower-cost transponders for smaller vessels (AIS class B) exist, many vessels such as leisure craft and kayaks are not equipped with AIS. This also extends to other objects in water such as navigational aids and debris. Transmitted information may also be faulty, either by accidental errors such as transmission failures or through intentional actions or negligence by the crew [14].

Other means of perception include exteroceptive sensors such as radars, lidars and cameras. The main advantage of these sensors are that they do not depend on other ships to transmit information, but instead transmit a signal and wait for a return, or passively capture information from the environment. This makes detection of kayaks and other small obstacles possible, but the measurements may be affected by clutter. The sensors may also have limited range, which means that objects can be very close before they are detected. This is usually not an issue for radars. There are several reports of use of exteroceptive sensors for maritime COLAV. The complementary properties of different sensors are thoroughly discussed in [6], with experimental validation. In [20], a maritime broadband radar intended for smaller recreational vessels is used for tracking in the joint probabilistic data association (JPDA) and interacting multiple model (IMM) framework.

Many target tracking models parametrize the target states as north and east position and velocity, while COLAV methods often use the total speed and course of the target. This may lead to fluctuations in these values, and the course angle in particular can fluctuate more than it should.

There are several solutions that can be applied to obtain better course estimates. One could argue that course and speed, instead of the linear world frame-parameterized velocity vector, should be used in the state vector. This would lead to a nonlinear process model. It is not clear that such a model would give improved performance. More refined models such as the Best-Norton model [2] require additional tuning to work well. Current research by the authors is exploring the utility of particle filtering for heading estimation [11]. Another approach would be to use multiple models within an IMM framework as in [20]. However, IMM is known to have limited effect when the so-called maneuvering index [15] is low. This can often be the case in maritime COLAV, when ships move relatively slowly compared to the measurement noise. In these cases, it may be feasible to smooth the target estimates using a form of retrodiction [5].

In this article, we report on a full-scale experiment where the DW algorithm was used in conjunction with a probabilistic data association filter (PDAF) radar-tracking filter to perform autonomous COLAV. A number of modifications to the DW algorithm in order to adapt the algorithm for use with high-speed ASVs are presented. We highlight a number of challenges arising from the results, motivating for further work.

The rest of the paper is structured as follows: Section 2 describes the platform used for the experiments. Section 3 describes the tracking system. Section 4 describes the DW algorithm, and the modifications applied to it. Section 5 describes the interface between the tracking and COLAV systems, while Section 6 show the results. Finally, concluding remarks and possibilities for further work are presented in Section 7.

2. EXPERIMENTAL PLATFORM

The vessel used in the experiments described in this paper is the dual-use Maritime Robotics Telemetron ASV, shown in Fig. 1. The ASV was equipped with a high grade navigation system from Kongsberg Seatex, supplemented by real-time GNSS corrections (CPOS) from the Norwegian mapping authority [22]. Conservative performance estimates for the navigation system are given in Table 1. Given the performance of the Seapath navigation system, navigation uncertainty is of minor impact to the target tracking system, and will be neglected. This is supported by simulation studies in [23] and [3] which indicated that navigation uncertainty would have to be significantly higher to have noticeable impact. The vessel tracking and control system is implemented in the robot operating system (ROS) [19], and the vessel actuators are interfaced via a proprietary interface developed by Maritime Robotics. A summary of the vessel parameters is given in Table 1.

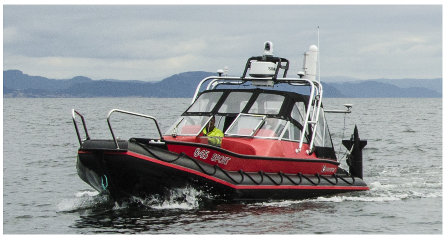


Figure 1. The Telemetron ASV, a dual-use vessel for both manned and unmanned operations. Courtesy of Maritime Robotics.

Table 1. Telemetron ASV specifications.

Component	**Description**
Vessel hull	Polarcirkel Sport 845
Length	8.45 m
Width	2.71 m
Weight	1675 kg
Propulsion system	Yamaha 225 HP outboard engine
Motor control	Electro-mechanical actuation of throttle valve
Rudder control	Hydraulic actuation of outboard engine angle with proportional-derivate (PD) feedback control on engine angle
Navigation system	Kongsberg Seatex Seapath 330+
Heading/roll/pitch accuracy	0.1° RMS
Position accuracy	0.1 m RMS
Radar	Simrad Broadband 4G$^{\text{TM}}$ Radar
Processing platform	Intel® i7 3.4 GHz CPU, running Ubuntu 16.04 Linux

The Telemetron vessel operates at speeds of up to $18\ \mathrm{m/s}$, which makes modeling and control of the vessel difficult since it operates in both the displacement, semi-displacement and planing regions [12]. In [8], a control-oriented model of the Telemetron ASV was developed, together with experimental validation of vessel controllers based on the vessel model. The model is defined in 2DOF using the speed over ground (SOG) and rate of turn (ROT) as states, denoted as U $[\mathrm{m/s}]$ and r $[\mathrm{rad/s}]$, respectively. The DW algorithm specifies a desired velocity which the vessel should follow. We therefore employ a velocity controller for SOG and ROT combining a proportional integral (PI) feedback controller with model-based feedforward of a desired velocity and acceleration shown to have good performance for the Telemetron ASV [8].

Table 2. Tracking module parameters used in the experiments.

Parameter	Value	Comment
T_r	2.4 s	Radar revolution time
q	$0.5\ \mathrm{m/s^2}$	Process noise standard deviation
r	$6.86\ \mathrm{m}$	Measurement noise standard deviation
N_s	5	Retrodiction window length

3. Target tracking: Theory and implementation

The core algorithm of the target tracking system is the probabilistic data association filter (PDAF), which is a single target tracking method which calculates the association probabilities for each measurement in the validation region of the target of interest [1]. It is not a multitarget method, but a parallel bank of PDAFs will be able to handle targets that are sufficiently temporally and/or spatially spaced. This covers the majority of the situations encountered in commercial ship traffic, and is thus chosen for its simplicity. The PDAF requires point measurements with known measurement noise covariance. The radar listed in Table 1 provides an array of spokes consisting of resolution cells with intensity information. The radar data must be processed in a detection, projection and clustering pipeline before it can be used in the PDAF. This pipeline is described in detail in [24]. The target motion and measurement model is the nearly constant velocity (NCV) model [17] with position measurements, given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{F}(T_r)\boldsymbol{x}_k + \boldsymbol{v}_k, \quad p(\boldsymbol{v}_k) = \mathcal{N}(\boldsymbol{v}_k; 0, \boldsymbol{Q}(T_r)), \tag{1}$$

$$\boldsymbol{z}_k = \boldsymbol{H}\boldsymbol{x}_k + \boldsymbol{w}_k, \quad p(\boldsymbol{w}_k) = \mathcal{N}(\boldsymbol{w}_k; 0, r^2\boldsymbol{I}), \tag{2}$$

where $\boldsymbol{x} = [p_N \quad v_N \quad p_E \quad v_E]^T$ is the state vector, containing position and velocity in north and east directions and $\boldsymbol{v}$ is the process noise. The variable $\boldsymbol{z}$ is the position measurement, while $\boldsymbol{w}$ is the measurement noise with covariance $r^2\boldsymbol{I}$ where $\boldsymbol{I}$ is the identity matrix. The matrix $\boldsymbol{H}$ extracts the position elements of the state vector, while the matrices $\boldsymbol{F}(T_r)$ and $\boldsymbol{Q}(T_r)$ are given as

$$\boldsymbol{F}(T_r) = \begin{bmatrix} 1 & T_r & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_r \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

$$\boldsymbol{Q}(T_r) = q^2 \begin{bmatrix} T_r^4/4 & T_r^3/2 & 0 & 0 \\ T_r^3/2 & T_r^2 & 0 & 0 \\ 0 & 0 & T_r^4/4 & T_r^3/2 \\ 0 & 0 & T_r^3/2 & T_r^2 \end{bmatrix}. \tag{4}$$

In addition to the radar sampling time T_r, which is given by the radar revolution time, the NCV model is parametrized by the covariance of the white noise acceleration q^2. This simple model reduces the risk of overfitting and tailoring the model to a specific target. A suitable value for the white noise acceleration standard deviation q was determinted to be $0.5\ \mathrm{m/s^2}$ through an analysis of covariance consistency, by considering AIS data from several vessels, including the Telemetron ASV [24]. This is a fairly large process noise value, which ensures good resilience against track-loss but also inevitably leads to some fluctuations in the course estimates, as shown in Fig. 6. The course angle is the angle of the velocity vector, measured clockwise from straight north, and is calculated as $\arctan_2(v_E, v_N)$ where $\arctan_2$ is the four-quadrant arctangent function.

Improved target course estimation

Several methods for improved course estimation was discussed in section 1. In this case, the target maneuvering index is found to be 0.42 from the values in Table 2, which falls below the limit of where the IMM estimator is preferred over the regular Kalman filter estimator [15]. Since the assumed target dynamics are moderately slow, the speed and course estimates will be improved using a retrodiction procedure.

Assume that the target, for the last N_s scans, have been moving in a straight line. The motion model for these steps can be written

$$\boldsymbol{x}_{k+1} = \boldsymbol{F}(T_r)\boldsymbol{x}_k, \quad k \in [K^\dagger, K], \tag{5}$$

where K is the latest timestamp of the estimate of the target, and $K^\dagger = K - N_s + 1$ is the start of the retrodiction interval. On this form, the motion of the target for the last N_s scans are parametrized by a constant velocity and an initial position $\boldsymbol{x}_{K^\dagger}$. The estimate from the PDAF is written as

$$\hat{\boldsymbol{x}}_{k|k} = \boldsymbol{F}(t_k - t_{K^\dagger})\boldsymbol{x}_{K^\dagger} + \boldsymbol{e}_k, \;\; p(\boldsymbol{e}_k) = \mathcal{N}(\boldsymbol{e}_k; 0, \boldsymbol{P}_{k|k}) \tag{6}$$

where $\hat{\boldsymbol{x}}_{k|k}$ and $\boldsymbol{P}_{k|k}$ is the posterior estimate of the PDAF. The retrodicted estimate of $\boldsymbol{x}_{K^\dagger}$ can be calculated by a standard least-squares calculation,

$$\bar{\boldsymbol{x}}_{K^\dagger} = (\boldsymbol{F}_r^T\boldsymbol{P}_r\boldsymbol{F}_r)^{-1}\boldsymbol{F}_r^T\boldsymbol{P}_r\hat{\boldsymbol{x}}_r, \tag{7}$$

where

$$\hat{\boldsymbol{x}}_r = \begin{bmatrix} \hat{\boldsymbol{x}}_{K^\dagger|K^\dagger} \\ \hat{\boldsymbol{x}}_{K^\dagger+1|K^\dagger+1} \\ \vdots \\ \hat{\boldsymbol{x}}_{K|K} \end{bmatrix}, \boldsymbol{F}_r = \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{F}(t_{K^\dagger+1} - t_{K^\dagger}) \\ \vdots \\ \boldsymbol{F}(t_K - t_{K^\dagger}) \end{bmatrix}, \tag{8}$$

$$\boldsymbol{P}_r = \mathrm{diag}(\boldsymbol{P}_{K^\dagger|K^\dagger}, \boldsymbol{P}_{K^\dagger+1|K^\dagger+1} \ldots, \boldsymbol{P}_{K|K}), \tag{9}$$

The retrodicted estimate of $\boldsymbol{x}_{K^\dagger}$ can then be used to calculate the estimate of the target at following timesteps, using (5).

We emphasize that the retrodiction is only used as a post-processing step for the output to the collision avoidance method, and not as an integral part of the tracking and prediction. Nevertheless, the estimates of the target used in the COLAV-algorithm will be affected if the target maneuvers. There are two main factors that will affect the safety of the system. The first is the time it takes to detect the maneuver in the retrodiction method, and the second is the safety regions of the ASV. The maneuver detection will depend on the retrodiction time. With the tracking system parameters shown in Table 2, this amounts to about 12 seconds. This corresponds to 60 meters with a target velocity of $5\ \mathrm{m/s}$. This means that the collision and safety regions described in Section 4 are significantly larger than the distance covered by the target during a turn. Should this not be the case, the ASV can take precautionary measures such as increasing the collision or safety regions, or decreasing its velocity.

4. Dynamic window: Theory and implementation

The dynamic window (DW) algorithm was introduced as a COLAV algorithm for indoor ground robots in 1997 [13]. The algorithm originally assumes that the vehicle is subject to constant acceleration limits, and predicts future trajectories using straight lines and circles. ASVs have nonlinear dynamics, which result in time-varying acceleration constraints. Moreover, the dynamics of an ASV is far more complex than that of an indoor ground robot, rendering the original prediction approach inaccurate. A modified DW algorithm for autonomous underwater vehicles (AUVs) moving in the horizontal plane is presented in [10], proposing a solution to these issues. This algorithm searches for feasible velocity pairs consisting of a desired surge speed u and yaw rate r, and chooses the optimal velocity pair based on an objective function. A set of feasible velocities is created by joining three search spaces, which then is sampled and used for predicting vessel trajectories with a prediction horizon T_p. The dynamic window contains velocities reachable during a time window T while respecting actuator rate saturations, and is defined as:

$$V_d = \Big\{(u,r) \in \mathbb{R}\times\mathbb{R} \Big| u \in [u^* + \dot{u}_{\min}T, u^* + \dot{u}_{\max}T] \\ \wedge r \in [r^* + \dot{r}_{\min}T, r^* + \dot{r}_{\max}T]\Big\}, \quad (10)$$

where u^* and r^* are the current surge speed and yaw rate, and $u_{\min}, u_{\max}$ and $r_{\min}, r_{\max}$ are time-varying acceleration limits. The set of possible velocities is defined as:

$$V_s = \Big\{(u,r) \in \mathbb{R}\times\mathbb{R} \Big| g(u,r) \geq 0\Big\}, \quad (11)$$

where $g(u,r)$ is positive for velocities that are possible to reach with respect to actuator magnitude saturations.

Two regions are defined around the obstacles, namely the collision and safety regions. The collision region is a circle, which if entered is treated as a collision. The safety region is a new circle outside of the collision region, which is allowed (but not desirable) to enter, hence acting as a safety margin. The set of admissible velocities only include velocities which allow the vehicle to stop before entering the collision region, and is defined as:

$$V_a = \Big\{(u,r) \in \mathbb{R}\times\mathbb{R} \Big| u \leq \sqrt{2\rho'(u,r)|\dot{u}_{\min}|} \\ \wedge |r| \leq \left\{ \begin{array}{ll} \sqrt{2\rho'(u,r)|\dot{r}_{\max}|} & ,r<0 \\ \sqrt{2\rho'(u,r)|\dot{r}_{\min}|} & ,r \geq 0 \end{array} \right\}, \quad (12)$$

where $\rho'(u,r)$ represents the along-trajectory distance to the collision region at the next time instant the algorithm is run, given the velocity pair (u,r). Finally, the optimal velocity pair is selected through maximizing an objective function:

$$(u_d, r_d) = \underset{(u,r)\in V_r}{\text{argmax}}\, G(u,r;u_d',r_d'), \quad (13)$$

where $V_r = V_d \cap V_s \cap V_a$, and $G(u,r;u_d',r_d')$ is defined as:

$$G(u,r;u_d',r_d') = \alpha\cdot\text{yawrate}(u,r,r_d') + \beta\cdot\text{dist}(u,r) \\ + \gamma\cdot\text{velocity}(u,r,u_d'), \quad (14)$$

where u_d' and r_d' are inputs to the algorithm, generated by a line of sight (LOS) guidance system [12], and $\alpha,\beta,\gamma > 0$ are tuning parameters. The yawrate$(\cdot)$ and velocity$(\cdot)$ terms assign value to choosing a velocity pair close to the desired velocity (u_d', r_d'), while the dist$(\cdot)$ term motivate the algorithm to keep distance to obstacles based on the along-trajectory distance to the safety region. These are defined as:

$$\text{yawrate}(u,r,r_d') = 1 - \frac{|r_d' - r|}{\max\limits_{r\in V_r}(|r_d' - r|)}, \quad (15)$$

$$\text{velocity}(u,r,u_d') = 1 - \frac{|u_d' - u|}{\max\limits_{u\in V_r}(|u_d' - u|)}, \quad (16)$$

$$\text{dist}(u,r) = \frac{\bar{\rho}(u,r)}{\frac{1}{T_p}\int_0^{T_p} \|\boldsymbol{\chi}(u,r,t)\|_2\,\mathrm{d}t}, \quad (17)$$

where $\boldsymbol{\chi}(u,r,t)$ is the predicted vessel body velocity and $\bar{\rho}(u,r)$ represents the along-trajectory distance to the safety region, both given the velocity pair (u,r).

The maximization in (13) is performed discretely by uniformly sampling the dynamic window V_d, and removing the velocity pairs which are not elements of V_s and V_a. In this process, predicted trajectories for the velocity pairs are generated using a model of the vehicle closed-loop error dynamics. See [10] for more details on the modified DW algorithm.

There are several differences between the AUV application in [10] and the ASV platform described in Section 2:

1. The model of the Telemetron ASV is formulated in 2DOF using SOG and ROT, instead of 3DOF using surge, sway and yaw as in [10].
2. The control system requires a continuously differentiable velocity trajectory to employ acceleration feed-forward.
3. The Telemetron ASV is not well captured by the AUV model used in [10].

The first issue is handled by redefining the DW algorithm for SOG and ROT. This requires a way to model the sideslip of the vessel, to be able to simulate the vessel kinematics [8]. For these experiments, we assume that the sideslip is small enough to be neglected. During the identification experiments in [8], we observed that the sideslip stays below $10°$ when operating the ASV at moderate speeds without extreme maneuvers.

The second issue is tackled by changing the way we generate the velocity pairs. Instead of sampling the dynamic window, we define a dynamic acceleration window:

$$A_d = \Big\{(\dot{U},\dot{r}) \in \mathbb{R}\times\mathbb{R} \Big| \dot{U} \in [\dot{U}_{\min}, \dot{U}_{\max}] \\ \wedge \dot{r} \in [\dot{r}_{\min}, \dot{r}_{\max}]\Big\}, \quad (18)$$

which we sample to obtain a list of acceleration pairs. We then define an acceleration trajectory as a piecewise linear trajectory. For a SOG acceleration sample $\dot{U}'$, the trajectory is defined as:

$$\dot{U}(t) = \left\{ \begin{array}{ll} \dot{U}_0 + \frac{\dot{U}'-\dot{U}_0}{T_{act}}t & ,0 \leq t < T_{act} \\ \dot{U}' & ,T_{act} \leq t < T_{acc} \\ \dot{U}' - \frac{\dot{U}'}{T_{act}}(t - T_{acc}) & ,T_{acc} \leq t < T_{acc} + T_{act} \\ 0 & ,otherwise, \end{array} \right. \quad (19)$$

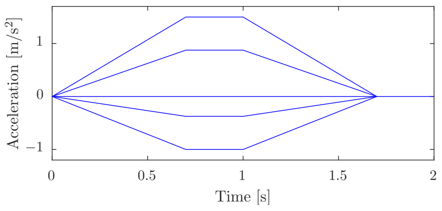


Figure 2. SOG acceleration trajectories for 5 samples in A_d.

where T_{act} is the time allowed for changing the actuator settings and T_{acc} is the time for acceleration, which should be equal to the sample time of the algorithm to allow for continuous accelerations. A collection of 5 SOG acceleration trajectories with $\dot{U}_0 = 0\,\mathrm{m/s^2}$, $T_{act} = 0.7$ s and $T_{acc} = 1$ s is shown in Fig. 2. Trajectories for ROT acceleration is obtained in a similar way.

The third issue is handled by simply using a kinematic model to obtain velocity and position trajectories given the acceleration trajectories. This relies on the velocity controller being able to track the desired trajectory, without too much bias introduced from external disturbances. Based on the experimental validation of the velocity controllers in [8], we believe that they will be able to quite closely track the desired velocity, and therefore justify this approach. In addition, the DW algorithm will be run at a quite high rate, which reduces the impact of external disturbances.

In addition, it has been found that using only the distance to the safety region in the distance term (17) of the objective function (14) may lead to the vessel being trapped in the safety region [21], [7], essentially disabling the COLAV aspect of the algorithm. To improve on this, an alternative distance term including the amount of a trajectory residing inside the safety region is proposed in [21]. This distance term does, however, rely on a part of the trajectory residing outside of the safety region, which may not be the case if the safety region is large. We therefore interchange the dist(·) term with a more complex one similar to the one in [21], but also including the distance to the collision region:

$$\mathrm{dist}(U,r) = \kappa \frac{\bar{\rho}(U,r) + \rho(U,r)}{\frac{1}{T}\int_0^T U \mathrm{d}t} + (1-\kappa)\frac{\sum_{n=1}^{N} \frac{\lambda(U,r,n)}{\sqrt{n}}}{\sum_{n=1}^{N} \frac{1}{\sqrt{n}}}. \tag{20}$$

Here, the function $\lambda(U,r,n)$ is 1 if point n of the predicted trajectory resides inside the safety region, and 0 otherwise, while $\kappa \in (0,1)$ is a tuning parameter. The variables $\bar{\rho}(U,r)$ and $\rho(U,r)$ are the along-trajectory distance to the safety and collision regions, respectively, both given the velocity pair (U,r). For more details, see [10] and [21].

In the experiments we ran the DW algorithm every second using 225 m and 350 m as the collision and safety region sizes, respectively. Further tuning parameters are shown in Table 3.

5. Target tracking and collision avoidance interface

Successfully closing the loop between the target tracking and COLAV systems requires that these two modules communicate in some way. The target tracking module has to provide the COLAV module with estimated target tracks in some manner. This can be posed as a software design problem, with many possible solutions. One way of enabling communication would be to implement the target tracking and COLAV functionality in one integrated module, handling communication implicitly within this module by sharing state. This is flexible and has the advantage of requiring very little explicit design, but at the same time makes the implementation less modular, harder to test and more difficult to develop and maintain as a team. An explicit interface has the additional flexibility of abstracting which sensor/sensors the target estimates originate from, and what kind of COLAV functionality the estimates are used for.

Within the ROS software framework, there are two main ways of communicating between separate software modules. The first is an asynchronous publish-subscribe mechanism, and the second is a synchronous request-response mechanism. We have chosen to use the latter model, such that the COLAV module requests a list of all known targets at specific times, i.e. a list of target trajectories that are discretized in time. This places the burden of track management, prediction and possibly interpolation in the target tracking module. All these functions are arguably naturally encapsulated within a target tracking framework. The interface is shown in Fig. 3.

Table 3. COLAV tuning parameters for the experiments.

Parameter	Value	Comment
T_p	30 s	Planning horizon length
T_{act}	0.7 s	Actuator ramp time
T_{acc}	1.0 s	Acceleration time
α	1.0	Yawrate function weight
β	4.0	Distance function weight
γ	1.0	Velocity function weight

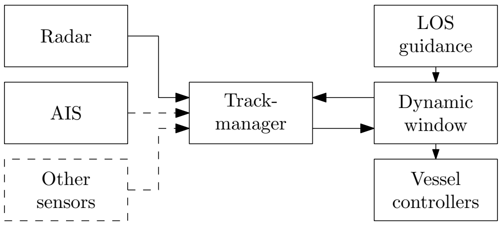


Figure 3. Diagram illustrating the target tracking and COLAV interface.

6. Closed-loop collision avoidance experiments

The combined tracking and COLAV system was tested in the Trondheimsfjord on the 15th to 19th of May 2017.

Scenario description

During the experiments, we tested a number of head-on situations with a target vessel under our control. The target vessel was a 17 foot motorboat constructed in glass fibre and equipped with a radar reflector to improve the visibility on the radar, as shown in Fig. 4. The target vessel was also equipped with an emulated AIS transponder using an

Figure 4. The 17 foot long target vessel, equipped with a radar reflector to improve radar visibility.

Android phone to transmit the vessel position, course and speed in the AIS format over the mobile network to the processing platform onboard the ASV. We also performed some COLAV experiments using the emulated AIS, but in the experiments presented here we only used it for ground truth. The experiments were performed with a constant desired SOG of $4\ \mathrm{m/s}$, while the guidance system attempts to follow a straight line towards the initial position of the target vessel. The target vessel was manually steered at a speed of approximately $5\ \mathrm{m/s}$, attempting to keep a constant course towards the initial position of the ASV. The scenarios were initiated with a distance of at least $900\ \mathrm{m}$ between the ASV and the target vessel.

Experimental results

The first experiment was performed using the PDAF tracking states for vessel prediction. From Fig. 5, we see that the ASV traveled too close to the target vessel, finally entering the collision region, which caused the experiment to be aborted for safety reasons. The reason for this was the large variations in course and speed estimates of the target ship, which when used in an NCV model results in large variations in the predicted future trajectory of the target vessel. This further results in that the ASV travels closer than it should to the target ship, as the DW algorithm in some iterations believes that the target vessel moves in another direction than the actual direction of travel. Finally, the target vessel is so close that there is no option to avoid entering the collision region, failing the experiment. It is quite apparent from the estimated target course in Fig. 6 that the predicted target trajectory will fluctuate a lot. This is also confirmed by Fig. 7 showing the distance at closest point of approach (DCPA) for the target vessel, noting that this is dependent on which trajectory the DW algorithm chooses. One should also note that when entering the collision region, all velocity pairs are considered inadmissible, in practice deactivating the DW algorithm.

The second experiment was performed using the retrodicted tracking states for vessel prediction. In this case, the ASV avoided the collision region, rendering the experiment as a

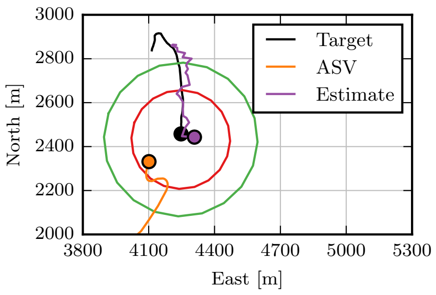


Figure 5. Experiment 1: North-east trajectory of the ASV and the target vessel, together with the PDAF position estimate used in the experiment. The experiment was aborted since the ASV traveled too close to the target vessel, moving into the collision region shown in red. The green circle show the safety region.

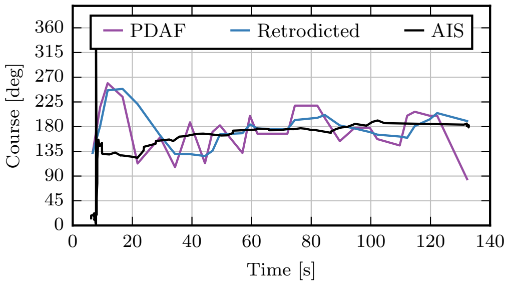


Figure 6. Experiment 1: Course estimate of the target vessel, with and without retrodiction. In this experiment, the PDAF estimate was used for predicting the target trajectory.

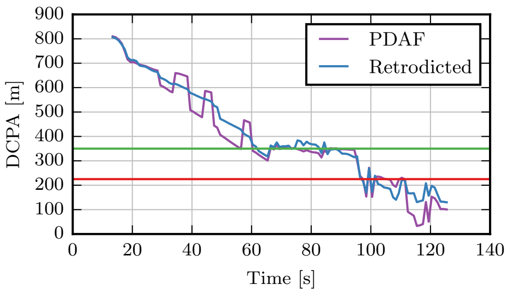


Figure 7. Experiment 1: DCPA for the target vessel. The collision and safety regions are shown in red and green, respectively.

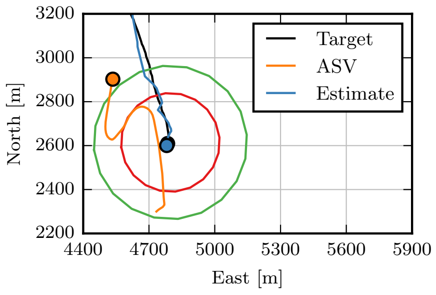


Figure 8. Experiment 2: North-east trajectory of the ASV and the target vessel, together with the retrodicted position estimate used in the experiment. The collision and safety regions are shown in red and green, respectively.

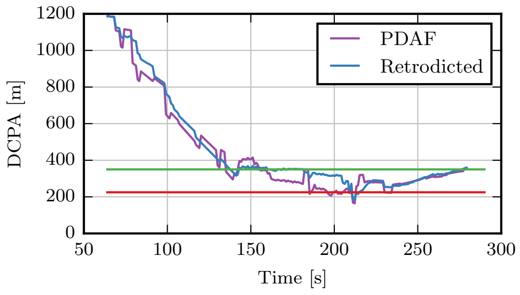


Figure 10. Experiment 2: DCPA for the target vessel. The collision and safety regions are shown in red and green, respectively.

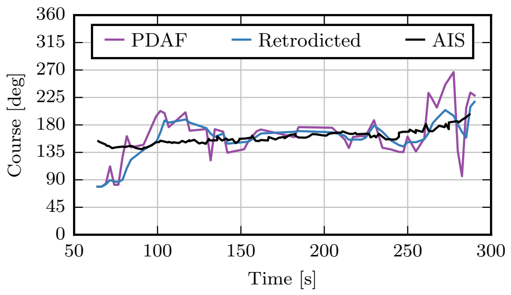


Figure 9. Experiment 2: Course estimate of the target vessel, with and without retrodiction. In this experiment, the retrodicted estimate was used for predicting the target trajectory.

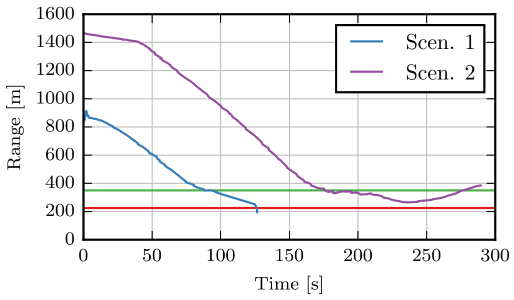


Figure 11. Distance to the target vessel for both experiments, with the collision and safety regions shown with red and green lines.

success. From Fig. 8, we see that the ASV successfully avoids the target vessel, before returning towards the path specified by the LOS guidance system. Figures 9 and 10 show the target course estimate and DCPA for the second experiment, showing the same trends as for the first experiment.

Fig. 11 shows the distance between the ASV and the target vessel for both experiments. It is clear that using a retrodicted target course and speed estimate improves the performance of the closed-loop COLAV system.

7. Conclusion

We have experimentally tested a closed-loop COLAV system consisting of a radar-based tracking system using PDAF and a COLAV system based on the DW algorithm. The system successfully avoided collision with a target vessel when using a retrodiction filter to generate smooth estimates of the target vessel course and speed, but failed when not applying such a filter. This highlights the requirements for the input to the COLAV system. The tracking system should hence be able to provide smooth estimates of the target course and speed, but the COLAV system should also be able to handle inaccurate estimates of target vessels to increase the robustness of the closed-loop system. In practice, the DW algorithm became deactivated when the ASV entered the collision region, since this causes all velocity pairs to be considered inadmissible. The COLAV system should steer the ASV to avoid this situation, but variations in the tracking estimates can still put the ASV in this position, revealing an important shortcoming of the DW algorithm.

Future work will investigate the potential for improvements in speed and course estimates through multiple model filtering, fixed-lag smoothing techniques and/or nonlinear motion models. The results also motivate for modifications to the DW algorithm, or the development of a new algorithm more robust to variations in the tracking estimates. An attractive topic is also to combine the reactive COLAV system with the deliberate COLAV algorithm in [9].

Acknowledgments

This work was supported by the Research Council of Norway through project number 244116 and the Centres of Excellence funding scheme with project number 223254. The authors would like to express great gratitude to Kongsberg Maritime and Maritime Robotics for providing high-grade navigation technology and the Telemetron vessel at our disposal for the experiments.

References

[1] Y. Bar-Shalom and X.-R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*. YBS Publishing, 1995.

[2] R. A. Best and J. Norton, "A new model and efficient tracker for a target with curvilinear motion," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, no. 3, pp. 1030–1037, 1997.

[3] E. F. Brekke and E. F. Wilthil, "Suboptimal Kalman filters for target tracking with navigation uncertainty in one dimension," in *2017 IEEE Aerospace Conference*, Big Sky, MT, USA, 2017, pp. 1–11.

[4] A. N. Cockcroft and J. N. F. Lameijer, *A Guide to the Collision Avoidance Rules*. Elsevier, 2004.

[5] O. E. Drummond, "Target tracking with retrodicted discrete probabilities," in *Signal and Data Processing of Small Targets*, vol. 3163. International Society for Optics and Photonics, 1997, pp. 249–269.

[6] L. Elkins, D. Sellers, and W. R. Monach, "The autonomous maritime navigation (AMN) project: Field tests, autonomous and cooperative behaviors, data fusion, sensors, and vehicles," *Journal of Field Robotics*, vol. 27, no. 6, pp. 790–818, 2010.

[7] B.-O. H. Eriksen, "Horizontal collision avoidance for autonomous underwater vehicles," Master's thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2015.

[8] B.-O. H. Eriksen and M. Breivik, "Modeling, identification and control of high-speed ASVs: Theory and experiments," in *Sensing and Control for Autonomous Vehicles: Applications to Land, Water and Air Vehicles*, T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer, Eds. Springer International Publishing, 2017, pp. 407–431.

[9] ——, "MPC-based mid-level collision avoidance for ASVs using nonlinear programming," in *Proc. of IEEE CCTA*, Kohala Coast, Hawai'i, USA, 2017.

[10] B.-O. H. Eriksen, M. Breivik, K. Y. Pettersen, and M. S. Wiig, "A modified dynamic window algorithm for horizontal collision avoidance for AUVs," in *Proc. of IEEE CCA*, Buenos Aires, Argentina, 2016.

[11] A. L. Flåten and E. F. Brekke, "Rao-blackwellized particle filter for turn rate estimation," in *2017 IEEE Aerospace Conference*, Big Sky, MT, USA, 2017, pp. 1–7.

[12] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons Ltd, 2011.

[13] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, pp. 23–33, 1997.

[14] A. Harati-Mokhtari, A. Wall, P. Brooks, and J. Wang, "Automatic identification system (AIS): Data reliability and human error implications," *Journal of Navigation*, vol. 60, no. 3, pp. 373–389, 2007.

[15] T. Kirubarajan and Y. Bar-Shalom, "Kalman filter versus IMM estimator: when do we need the latter?" *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1452–1457, 2003.

[16] Y. Kuwata, M. T. Wolf, D. Zarzhitsky, and T. L. Huntsberger, "Safe maritime autonomous navigation with COLREGS, using velocity obstacles," *IEEE J. Oceanic Eng.*, vol. 39, no. 1, pp. 110–119, 2014.

[17] X.-R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I. Dynamic models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333–1364, 2003.

[18] Ø. A. G. Loe, "Collision avoidance for unmanned surface vehicles," Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2008.

[19] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source robot operating system," in *Proc. of the IEEE ICRA Workshop on Open Source Robotics*, Kobe, Japan, 2009.

[20] M. Schuster, M. Blaich, and J. Reuter, "Collision avoidance for vessels using a low-cost radar sensor," in *Proc. of the 19th IFAC World Congress*, Cape Town, South Africa, 2014, pp. 9673–9678.

[21] E. Serigstad, "Hybrid collision avoidance for autonomous surface vessels," Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2017.

[22] The Norwegian Mapping Authority, "CPOS: Positioning," https://kartverket.no/posisjonstjenester/cpos/, accessed: 2017-10-13.

[23] E. F. Wilthil and E. F. Brekke, "Compensation of navigation uncertainty for target tracking on a moving platform," in *19th International Conference on Information Fusion (FUSION)*, Heidelberg, Germany, 2016, pp. 1616–1621.

[24] E. F. Wilthil, A. L. Flåten, and E. F. Brekke, "A target tracking system for ASV collision avoidance based on the PDAF," in *Sensing and Control for Autonomous Vehicles: Applications to Land, Water and Air Vehicles*, T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer, Eds. Springer International Publishing, 2017, pp. 269–288.

Biography

__Bjørn-Olav H. Eriksen__ is a PhD candidate at the Department of Engineering Cybernetics at NTNU, where he also obtained his MSc in Engineering Cybernetics in 2015. He specializes in collision avoidance for autonomous surface vehicles, currently working in the project "Sensor fusion and collision avoidance for autonomous surface vehicles".

Erik F. Wilthil *is a PhD candidate at the Department of Engineering Cybernetics at NTNU, where he also obtained his MSc in Engineering Cybernetics in 2015. He specializes in target tracking and navigation for autonomous surface vehicles, currently working in the project "Sensor fusion and collision avoidance for autonomous surface vehicles".*

Andreas L. Flåten *is a PhD candidate at the Department of Engineering Cybernetics at NTNU, where he also obtained his MSc in Engineering Cybernetics in 2015. He specializes in multisensor fusion for collision avoidance, currently working in the project "Sensor fusion and collision avoidance for autonomous surface vehicles".*

Edmund F. Brekke *has an MSc (2005) in Industrial Mathematics and a PhD (2010) in Engineering Cybernetics, both awarded by NTNU. His PhD research covered methods for target tracking using active sonar. This research was conducted at the University Graduate Center at Kjeller in collaboration with Kongsberg Maritime. After his PhD studies, Brekke worked as a postdoctoral research fellow at the Acoustic Research Lab at NUS in Singapore, before becoming an Associate Professor in Sensor Fusion at NTNU in 2014. Brekke's main research interests lie in Bayesian estimation, with applications in target tracking and autonomous navigation. Brekke is project manager of the competence-building research project "Sensor fusion and collision avoidance for autonomous surface vehicles" funded by the Research Council of Norway, Kongsberg Maritime, DNV GL and Maritime Robotics.*

Morten Breivik *has an MSc (2003) and a PhD (2010) in Engineering Cybernetics from NTNU. He is currently Head of NTNU's Department of Engineering Cybernetics, and is also an associate researcher at the Centre for Autonomous Marine Operations and Systems (NTNU AMOS). Breivik has previously worked as an assistant professor and researcher at NTNU, a scientific advisor for Maritime Robotics, and as a principal engineer and department manager in applied cybernetics at Kongsberg Maritime. His research interests include nonlinear and adaptive motion control of unmanned vehicles in general and autonomous ships in particular. Breivik is also currently a member of the Norwegian Board of Technology.*

Paper E A model-based speed and course controller for high-speed ASVs

Postprint of **B.-O. H. Eriksen** and M. Breivik, "A model-based speed and course controller for high-speed ASVs", in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*, (Opatija, Croatia), 2018, pp. 317–322.

A Model-Based Speed and Course Controller for High-Speed ASVs

Bjørn-Olav H. Eriksen [*] **Morten Breivik** [*]

[*] *Centre for Autonomous Marine Operations and Systems*
Department of Engineering Cybernetics
Norwegian University of Science and Technology (NTNU)
NO-7491 Trondheim, Norway
E-mail: {bjorn-olav.h.eriksen, morten.breivik}@ieee.org

Abstract: Well-performing low-level vessel controllers are a necessity for marine motion control applications such as path following, trajectory tracking and collision avoidance for autonomous surface vehicles (ASVs). Designing such controllers are especially challenging for high-speed vessels operating in both the displacement, semi-displacement and planing regions. In this article, we build on a powerful framework for modeling, identification and control of high-speed ASVs in order to develop a model-based speed and course controller with high performance. The controller is shown to outperform a gain-scheduled proportional-integral feedback controller in full-scale experiments in the Trondheimsfjord, Norway. The controllers are compared both qualitatively and quantitatively using suitable performance metrics.

Keywords: Model-based control, autonomous surface vehicles, high-speed motion control

1. INTRODUCTION

Kinematic control applications such as path following and trajectory tracking (Fossen, 2011) rely on high-performance low-level vessel controllers. The same is the case for autonomy-enabling collision avoidance (COLAV) functionality (Kuwata et al., 2014; Eriksen and Breivik, 2017b; Eriksen et al., 2018). Hence, to be able to employ autonomous surface vehicles (ASVs) for a range of marine control applications, precise and robust low-level motion controllers are required.

In general, ASVs are small and agile vessels capable of operating at high speeds. Such vessels often operate in both the displacement, semi-displacement and planing regions. In the displacement region, the hydrostatic pressure mainly carries the vessel weight. When increasing the vessel speed, hydrodynamic effects will increase and eventually carry the majority of the vessel weight. When the hydrodynamic pressure dominates the hydrostatic pressure, the vessel operates in the planing region, and between the displacement and planing regions is the semi-displacement region (Fossen, 2011). Most modeling approaches assume that the vessel only operates in the displacement region, which makes modeling of high-speed ASVs challenging. This again makes it difficult to develop high-performance low-level motion controllers utilizing mathematical models of the vessel.

In (Breivik et al., 2008), an ASV speed and course controller using steady-state feedforward in speed is developed and used in full-scale trajectory tracking experiments with an ASV moving beyond the displacement region. The controller is extended with steady-state feedforward in yaw rate and used for formation control in (Breivik, 2010).

In this paper, we build upon a powerful approach for modeling, identification and control of high-speed ASVs operating in both the displacement, semi-displacement and planing regions (Eriksen and Breivik, 2017a). In particular, a vessel speed and yaw-rate controller employing model-based feedforward terms is developed in (Eriksen and Breivik, 2017a) and shown to outperform other controllers. Here, we extend this controller to also control the vessel course, which enables the vessel velocity to be controlled precisely. As such, it is important to note the difference between heading (yaw angle) and course, where the former relates to the direction which the vessel bow is pointing while the course is the direction which the vessel is traveling. When the vessel is maneuvering or under influence of external disturbances, the heading and course will in most cases not be aligned. The performance of the new model-based speed and course controller is evaluated through full-scale experiments in the Trondheimsfjord, Norway.

The rest of the paper is structured as follows: Section 2 presents the Telemetron ASV and gives a summary of the modeling and identification approach developed in (Eriksen and Breivik, 2017a). In Section 3, we develop the new speed and course controller, while Section 4 presents the results from the full-scale experiments. Finally, Section 5 concludes the paper and suggests some further work.

2. ASV PLATFORM AND MODELING

The vessel considered in this paper is the Maritime Robotics' Telemetron ASV, shown in Figure 1. This is a dual-use vessel, designed for both manned and unmanned operations. The vessel is 8.45 m long and capable of speeds up to 18 m/s. See Table 1 for more specifications.

Fig. 1. The dual-use Telemetron ASV, which is 8.45 m long and capable of speeds up to 18 m/s. Courtesy of Maritime Robotics.

Most ASVs are modeled using the well-known 3 degrees of freedom (DOF) model (Fossen, 2011):

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\nu} \tag{1a}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau} + \boldsymbol{\tau}_{\text{environment}}, \tag{1b}$$

where $\boldsymbol{\eta} = [N \ E \ \psi]^T$ is the vessel pose, $\boldsymbol{\nu} = [u \ v \ r]^T$ is the vessel body velocity, while $\boldsymbol{\tau}$ and $\boldsymbol{\tau}_{\text{environment}}$ are the control and environmental forces and moments, respectively. The matrix $\boldsymbol{R}(\psi)$ is a rotation matrix, while $\boldsymbol{M}$, $\boldsymbol{C}(\boldsymbol{\nu})$ and $\boldsymbol{D}(\boldsymbol{\nu})$ are the inertia, Coriolis/centripetal and damping matrices, respectively. Although the model (1) is frequently used for modeling vessels operating at high speeds, the model is developed under the assumption that the vessel operates in the displacement region. The operating region of a vessel can be approximated by computing the Froude number (Faltinsen, 2005). Using this number, it can be shown that an ASV with submerged length of 8 m exits the displacement region at around 3.54 m/s, while entering the planing region at approximately 8.86 m/s (Eriksen and Breivik, 2017a). Hence, an alternative model to (1) is required to develop a motion control system for a vessel like the Telemetron ASV.

2.1 Control-oriented modeling of high-speed ASVs

In (Eriksen and Breivik, 2017a), a data-driven control-oriented modeling approach for high-speed ASVs is proposed. Most ASVs are underactuated, making it impossible to independently control surge, sway and yaw simultaneously. One usually chooses to control the surge and yaw motion, leaving the sway motion uncontrolled. Therefore, a 2DOF model using the vessel speed over ground (SOG) $U = \sqrt{u^2 + v^2}$ and rate of turn (ROT) r as states is suitable for control purposes. The kinematics of the vessel using SOG and ROT can be described as:

$$\dot{\boldsymbol{\eta}} = \begin{bmatrix} \cos(\chi) & 0 \\ \sin(\chi) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ r \end{bmatrix} \tag{2}$$
$$\dot{\chi} = r + \dot{\beta},$$

where r is the vessel ROT and $\chi = \psi + \beta$ is the vessel course with β being the vessel sideslip.

To model the vessel dynamics, we define a state vector $\boldsymbol{x} = [U \ r]^T$ and use a normalized 2DOF non-first principles model to model the vessel dynamics:

$$\boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}} + \boldsymbol{\sigma}(\boldsymbol{x}) = \boldsymbol{\tau}, \tag{3}$$

Table 1. Telemetron ASV specifications.

Component	**Description**
Vessel hull	Polarcirkel Sport 845
Length	8.45 m
Width	2.71 m
Weight	1675 kg
Propulsion system	Yamaha 225 HP outboard engine
Motor control	Electro-mechanical actuation of throttle valve
Rudder control	Hydraulic actuation of outboard engine angle with proportional-derivate (PD) feedback control
Navigation system	Kongsberg Seatex Seapath 330+
Processing platform	Intel® i7 3.4 GHz CPU, running Ubuntu 16.04 Linux

where $\boldsymbol{M}(\boldsymbol{x}) = \operatorname{diag}(m_U(\boldsymbol{x}), m_r(\boldsymbol{x}))$ is a diagonal state-dependent inertia matrix and $\boldsymbol{\sigma}(\boldsymbol{x}) = [\sigma_U(\boldsymbol{x}) \ \sigma_r(\boldsymbol{x})]^T$ is a vector of damping terms, both being nonlinear in terms of the state vector $\boldsymbol{x}$. The vector $\boldsymbol{\tau} = [\tau_m \ \tau_\delta]^T$ is a normalized control input, where $\tau_m \in [0,1]$ and $\tau_\delta \in [-1,1]$ are the motor throttle and rudder control input, respectively.

The terms of the inertia matrix $\boldsymbol{M}(\boldsymbol{x})$ and damping term $\boldsymbol{\sigma}(\boldsymbol{x})$ are defined using high-order polynomial functions:

$$\begin{aligned} m_U(\boldsymbol{x}) &= \boldsymbol{\phi}_m(\boldsymbol{x})^T\boldsymbol{\beta}_{m_U}, & \sigma_U(\boldsymbol{x}) &= \boldsymbol{\phi}_\sigma(\boldsymbol{x})^T\boldsymbol{\beta}_{\sigma_U} \\ m_r(\boldsymbol{x}) &= \boldsymbol{\phi}_m(\boldsymbol{x})^T\boldsymbol{\beta}_{m_r}, & \sigma_r(\boldsymbol{x}) &= \boldsymbol{\phi}_\sigma(\boldsymbol{x})^T\boldsymbol{\beta}_{\sigma_r}, \end{aligned} \tag{4}$$

where $\boldsymbol{\phi}_m(\boldsymbol{x})$ and $\boldsymbol{\phi}_\sigma(\boldsymbol{x})$ are vectors of basis functions, called regressors, while $\boldsymbol{\beta}_{m_U}, \boldsymbol{\beta}_{m_r}, \boldsymbol{\beta}_{\sigma_U}$ and $\boldsymbol{\beta}_{\sigma_r}$ are parameter vectors. Notice that the terms are linear in the parameters, which allows the use of linear regression (Bishop, 2006) to find the optimal parameter vectors. The regressors are defined as a fourth-order polynomial function for the damping:

$$\begin{aligned} \boldsymbol{\phi}_\sigma(\boldsymbol{x}) = \big[&1,\ U,\ r,\ U^2,\ Ur,\ r^2,\ U^3,\ U^2r,\ Ur^2, \\ &r^3,\ U^4,\ U^3r,\ U^2r^2,\ Ur^3,\ r^4\big]^T, \end{aligned} \tag{5}$$

and as the same fourth-order polynomial plus an asymptotic function for the inertia:

$$\begin{aligned} \boldsymbol{\phi}_M(\boldsymbol{x}) = \big[&1,\ U,\ r,\ U^2,\ Ur,\ r^2,\ U^3, \\ &U^2r,\ Ur^2,\ r^3,\ U^4,\ U^3r, \\ &U^2r^2,\ Ur^3,\ r^4,\ \tanh(a(U-b))\big]^T, \end{aligned} \tag{6}$$

where a and b are parameters controlling the asymptotic term. The order of the polynomials is chosen sufficiently high to capture hydrodynamic damping and actuator dynamics. The asymptotic term in the inertia regressor is motivated by experiments showing a large decrease in the inertia for increasing SOG at low speeds.

2.2 Parameter identification

As mentioned, the linearity of parameters in the terms (4) allows for the use of linear regression to identify optimal parameter values. For a model on the form:

$$y = \boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\beta}, \tag{7}$$

one can, given a set of N data points $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$ and $\{y_1, y_2, \dots, y_N\}$, define a weighted square loss function:

$$\epsilon = \frac{1}{N}\sum_{i=1}^{N} W_{ii}\left(y_i - \boldsymbol{\phi}(\boldsymbol{x}_i)^T\hat{\boldsymbol{\beta}}\right)^2, \tag{8}$$

with $\hat{\boldsymbol{\beta}}$ being an estimate of the true parameter vector $\boldsymbol{\beta}$, and W_{ii} being a weight on data point i. Defining $\boldsymbol{Y} = [y_1\ y_2\ \dots\ y_N]^T$, $\boldsymbol{X} = [\boldsymbol{\phi}(\boldsymbol{x}_1)\ \boldsymbol{\phi}(\boldsymbol{x}_2)\ \dots\ \boldsymbol{\phi}(\boldsymbol{x}_N)]^T$ and $\boldsymbol{W} = \operatorname{diag}(W_{11}, W_{22}, \dots, W_{NN})$, we can find the parameter vector $\hat{\boldsymbol{\beta}}$ which minimizes (8) using linear regression as:

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{Y}. \tag{9}$$

A well-known problem with linear regression is the problem of overfitting. To reduce this problem, one can introduce regularization, which penalizes parameter vectors with large parameter values (Bishop, 2006). The loss function (8) is then reformulated as:

$$\epsilon = \frac{1}{N}\sum_{i=1}^{N} W_{ii}\left(y_i - \boldsymbol{\phi}(\boldsymbol{x}_i)^T\hat{\boldsymbol{\beta}}\right)^2 + \lambda R(\hat{\boldsymbol{\beta}}), \tag{10}$$

where $\lambda > 0$ is a regularization parameter and $R(\hat{\boldsymbol{\beta}})$ is a regularization function. In (Eriksen and Breivik, 2017a), ℓ_1 regularization is used, since it favors sparse parameter vectors. Hence, the regularization function is defined as $R(\hat{\boldsymbol{\beta}}) = \left\|\hat{\boldsymbol{\beta}}\right\|_1$.

In (Eriksen and Breivik, 2017a), a number of vessel experiments were conducted in order to obtain measurements of the vessel inertia and damping. These experiments were based on series of step responses in the motor throttle τ_m and rudder angle τ_δ. Based on the experiments, three datasets of measurements were obtained:

$$\mathcal{D}_{\boldsymbol{\sigma}} = \left\{\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{N_\sigma}\}, \{\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2, \dots, \boldsymbol{\sigma}_{N_\sigma}\}\right\}, \tag{11}$$

$$\mathcal{D}_{m_U} = \Big\{\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{N_{m_U}}\}, \{m_{U_1}, m_{U_2}, \dots, m_{U_{N_{m_U}}}\}\Big\}, \tag{12}$$

$$\mathcal{D}_{m_r} = \Big\{\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{N_{m_r}}\}, \{m_{r_1}, m_{r_2}, \dots, m_{r_{N_{m_r}}}\}\Big\}. \tag{13}$$

The set $\mathcal{D}_{\boldsymbol{\sigma}}$ contains vessel states and damping measurements, $\mathcal{D}_{m_U}$ contains vessel states and measurements of SOG inertia while $\mathcal{D}_{m_r}$ contains vessel states and measurements of ROT inertia. Using the data in (11)–(13), the parameter vectors $\boldsymbol{\beta}_{m_U}, \boldsymbol{\beta}_{m_r}, \boldsymbol{\beta}_{\sigma_U}$ and $\boldsymbol{\beta}_{\sigma_r}$ in (4) are identified using linear regression with ℓ_1 regularization. Figures 2 and 3 show the resulting model terms.

Figure 4 shows a comparison between real and simulated vessel responses for the same control input sequence. The real vessel response in Figure 4 was not used in the identification process, hence the comparison can be used to qualitatively verify the identified model. There are some deviations between the real and simulated responses, but the model is considered accurate enough for control purposes.

Interested readers are referred to (Eriksen and Breivik, 2017a) for more details on the modeling and identification framework.

3. ASV SPEED AND COURSE CONTROLLER

In (Eriksen and Breivik, 2017a), the model (3) is used for control of the vessel SOG and ROT using model-based

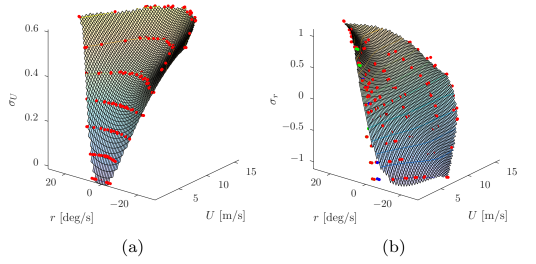


Fig. 2. Surface plot of the damping term $\boldsymbol{\sigma}(\boldsymbol{x})$. The scatter points are the data points in $\mathcal{D}_{\boldsymbol{\sigma}}$ where red, blue and green points have weights $W = 1$, $W = 0.5$ and $W = 0.1$, respectively (Eriksen and Breivik, 2017a).

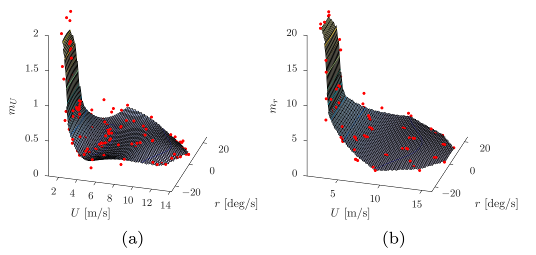


Fig. 3. Surface plot of the inertia terms in $\boldsymbol{M}(\boldsymbol{x})$. The scatter points are the data points in $\mathcal{D}_{m_U}$ and $\mathcal{D}_{m_r}$ (Eriksen and Breivik, 2017a).

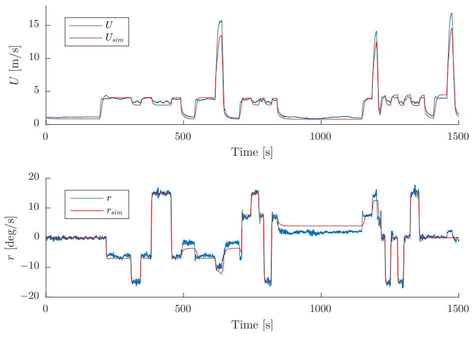


Fig. 4. Comparison of real and simulated vessel responses. The deviation at high SOG is caused by leaving the valid domain of the identified model (Eriksen and Breivik, 2017a).

feedforward terms. A controller named the feedforward feedback (FF-FB) controller is suggested as:

$$\boldsymbol{\tau}_{\text{FF-FB}} = \boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}}_d + \boldsymbol{\sigma}(\boldsymbol{x}_d) - \boldsymbol{M}(\boldsymbol{x})\boldsymbol{K}_p\tilde{\boldsymbol{x}} - \boldsymbol{K}_i\int_{t_0}^{t}\tilde{\boldsymbol{x}}(\gamma)\mathrm{d}\gamma, \tag{14}$$

where $\boldsymbol{x}_d = [U_d\ r_d]^T$ is a vector of desired SOG and ROT, $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_d$ is the control error, $\boldsymbol{K}_p > 0$ is a diagonal proportional gain matrix and $\boldsymbol{K}_i > 0$ is a diagonal integral gain matrix. The controller (14) is shown to have significantly better performance than a gain-scheduled proportional-integral (PI) feedback controller, a pure feedforward controller and a feedback-linearizing controller.

In many applications, it is desirable to control the vessel kinematics, for instance by controlling the ASV course. Similar to the design of vessel SOG and ROT controllers in (Eriksen and Breivik, 2017a), utilizing model-based feedforward terms should increase the closed-loop performance of a controller also for the vessel kinematics. Hence, we propose to extend the FF-FB controller (14) with feedback terms for the vessel course χ, defining the feedforward-feedback course (FF-FB-C) controller as:

$$\begin{aligned}\boldsymbol{\tau}_{\text{FF-FB-C}} &= \boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}}_d + \boldsymbol{\sigma}(\boldsymbol{x}_d) \\ &\quad - \boldsymbol{M}(\boldsymbol{x})\bar{\boldsymbol{K}}_p\tilde{\boldsymbol{\zeta}} - \bar{\boldsymbol{K}}_i\int_{t_0}^{t}\tilde{\boldsymbol{\zeta}}_1(\gamma)\mathrm{d}\gamma, \quad (15)\end{aligned}$$

where $\tilde{\boldsymbol{\zeta}}$ and $\tilde{\boldsymbol{\zeta}}_1$ are the error terms:

$$\begin{aligned}\tilde{\boldsymbol{\zeta}} &= \begin{bmatrix}\boldsymbol{x}-\boldsymbol{x}_d \\ \Upsilon(\chi-\chi_d)\end{bmatrix} = \begin{bmatrix}\tilde{U} \\ \tilde{r} \\ \tilde{\chi}\end{bmatrix} \\ \tilde{\boldsymbol{\zeta}}_1 &= \begin{bmatrix}\tilde{U} \\ \tilde{\chi}\end{bmatrix},\end{aligned} \quad (16)$$

with χ_d being the desired vessel course and $\Upsilon : \mathbb{R} \to S^1$ mapping an angle to the domain $[-\pi, \pi)$. The matrices $\bar{\boldsymbol{K}}_p$ and $\bar{\boldsymbol{K}}_i$ contain positive proportional and integral gains, respectively:

$$\begin{aligned}\bar{\boldsymbol{K}}_p &= \begin{bmatrix}k_{p_U} & 0 & 0 \\ 0 & k_{p_r} & k_{p_\chi}\end{bmatrix} \\ \bar{\boldsymbol{K}}_i &= \begin{bmatrix}k_{i_U} & 0 \\ 0 & k_{i_\chi}\end{bmatrix}.\end{aligned} \quad (17)$$

Through (2), the relationship between the course and ROT is stated as $r = \dot{\chi} - \dot{\beta}$, where the sideslip β enters the equation. Currently, we do not have a sideslip model of the Telemetron ASV. However, we have seen in experiments that at moderate speeds the sideslip is sufficiently constant such that $\dot{\beta} \approx 0$ can be assumed without major implications. We therefore simplify the relation by assuming constant sideslip and defining the desired ROT as:

$$\begin{aligned}r_d &= \dot{\chi}_d \\ \dot{r}_d &= \ddot{\chi}_d.\end{aligned} \quad (18)$$

Hence, assuming a constant or slowly-varying sideslip, the controller (15) is a speed and course controller which is able to precisely control the vessel velocity, which is required for kinematic control applications.

4. EXPERIMENTAL RESULTS

The controller performance is tested through full-scale experiments in the Trondheimsfjord in Norway on the 10th of October 2017 using the Telemetron ASV. The sea state was considered as slight, which refers to significant wave heights of 0.5–1.25 m (Prince and Bishop, 1974).

The FF-FB-C controller is compared to a PI feedback controller with gain scheduling:

$$\boldsymbol{\tau}_{\text{FB-C}} = -\boldsymbol{M}(\boldsymbol{x})\bar{\boldsymbol{K}}_p\tilde{\boldsymbol{\zeta}} - \bar{\boldsymbol{K}}_i\int_{t_0}^{t}\tilde{\boldsymbol{\zeta}}_1(\gamma)\mathrm{d}\gamma, \quad (19)$$

which is named the feedback course (FB-C) controller. Notice that the FB-C controller is obtained by removing the feedforward terms of (15). The same controller parameters were used for both controllers:

$$\begin{aligned}k_{p_U} &= 0.6 \quad & k_{i_U} &= 0.01 \\ k_{p_\chi} &= 0.15 \quad & k_{i_\chi} &= 0.015 \\ k_{p_r} &= 0.35.\end{aligned} \quad (20)$$

To generate the derivatives required for the controllers, the desired SOG trajectory $U_d(t)$ must be continuously differentiable, while the desired course trajectory $\chi_d(t)$ must be twice continuously differentiable. To ensure this, a second order reference filter is used to generate $U_d(t)$, while a third order reference filter is used to generate $\chi_d(t)$.

4.1 Performance metrics

To quantitatively evaluate the performance of the controllers, performance metrics are useful. For simplicity in analyzing the performance, we wish to combine both speed and course errors in the metrics. Since these have different units, we must introduce some sort of weighted sum (Eriksen and Breivik, 2017a). To do this, we define the normalized signals $\bar{U}, \bar{U}_d, \bar{\chi}, \bar{\chi}_d \in [0, 1]$ for the expected operational space of the vessel. For the Telemetron ASV, the expected operating space is up to 18 m/s for SOG, and 2π rad for course since it resides in S^1. We then define a combined error term and control input as:

$$\bar{e}(t) = \left\| \begin{bmatrix}\bar{U}(t) - \bar{U}_d(t) \\ \bar{\chi}(t) - \bar{\chi}_d(t)\end{bmatrix} \right\|_2 \quad (21)$$

and

$$\bar{\tau}(t) = \|\boldsymbol{\tau}\|_2\,. \quad (22)$$

Different performance metrics are used to evaluate different qualities. The integral of absolute error (IAE) penalizes deviation from the desired speed and course:

$$IAE(t) = \int_{t_0}^{t}\bar{e}(\gamma)\mathrm{d}\gamma, \quad (23)$$

and serves as a measure of control precision. The integral of absolute differentiated control (IADC) has previously been used as a part of a combined performance metric in (Sørensen et al., 2016):

$$IADC(t) = \int_{t_0}^{t}|\dot{\bar{\tau}}(\gamma)|\mathrm{d}\gamma, \quad (24)$$

and serves as a measure of wear and tear on the actuators. The integral of absolute error times the integral of absolute differentiated control (IAE-ADC) combines IAE and IADC, and is computed as:

$$IAE\text{-}ADC(t) = \int_{t_0}^{t}\bar{e}(\gamma)\mathrm{d}\gamma\int_{t_0}^{t}|\dot{\bar{\tau}}(\gamma)|\mathrm{d}\gamma, \quad (25)$$

and is a measure of control precision scaled by wear and tear (Eriksen and Breivik, 2017a). Finally, the integral of absolute error times work (IAEW) scales IAE with the energy consumption (Sørensen and Breivik, 2015), and is computed as:

$$IAEW(t) = \int_{t_0}^{t}\bar{e}(\gamma)\mathrm{d}\gamma\int_{t_0}^{t}P(\gamma)\mathrm{d}\gamma, \quad (26)$$

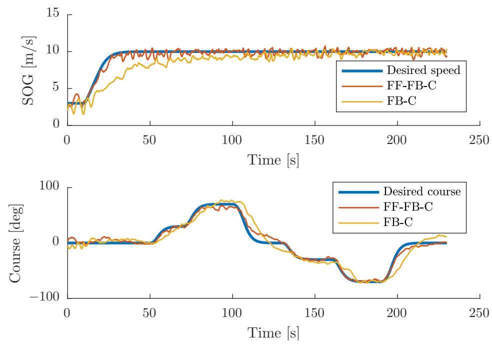
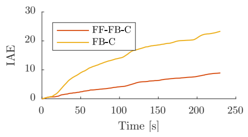
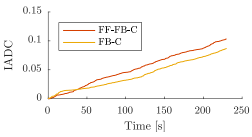


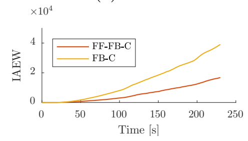
Fig. 5. Test 1: High-speed trajectory with constant SOG and steps and steady states in course.



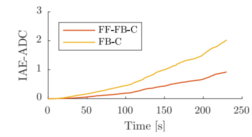
(a) IAE



(b) IADC



(c) IAEW



(d) IAE-ADC

Fig. 6. Performance metrics for Test 1. The FF-FB-C controller clearly outperforms the FB-C controller on all metrics except the IADC.

where $P(t)$ is the mechanical power applied by the vessel motor. Hence, IAEW serves as a measure of the energy efficiency. In this work, we approximate the mechanical power as linear with the motor throttle, hence $P \propto \tau_m$.

4.2 Full-scale experiments

In this section, we present experimental results from two test scenarios:

- Test 1: A high-speed trajectory with constant SOG and step changes in course, with steady states in between.
- Test 2: A high-speed trajectory with step changes and steady states in both SOG and course.

Test 1 For the first test, we have a high-speed trajectory with constant SOG, while attempting to follow a course trajectory consisting of both steps and steady states. The resulting trajectories are shown in Figure 5. It is clear that the FF-FB-C controller follows the desired SOG trajectory better than the FB-C controller, which also is the case for the course. In particular, the FF-FB-C controller has a better transient response. The FB-C controller struggles to follow a changing reference, while also having problems with overshoots, which is a natural result of using integral terms to stabilize the controlled variables.

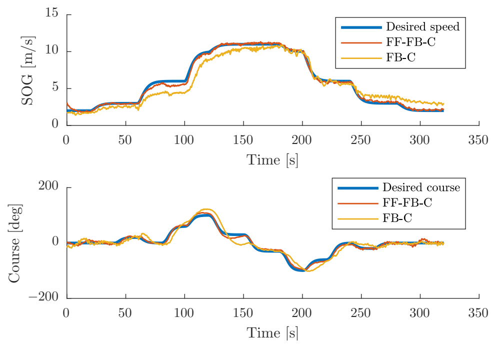
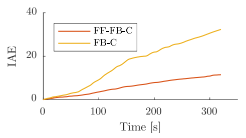
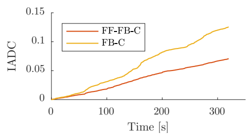


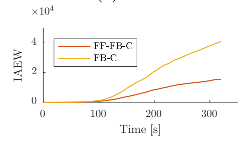
Fig. 7. Test 2: High-speed trajectory with steps and steady states in both SOG and course.



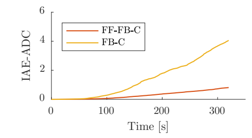
(a) IAE



(b) IADC



(c) IAEW



(d) IAE-ADC

Fig. 8. Performance metrics for Test 2. The FF-FB-C controller clearly outperforms the FB-C controller on all the metrics.

From the performance metrics in Figure 6, we see that the FF-FB-C controller has the best performance. In particular, the IAE is much lower with the FF-FB-C controller than the FB-C controller. The control usage (IADC) is slightly higher for the FF-FB-C controller, but when scaled with the control performance (IAE-ADC), the FF-FB-C outperforms the FB-C controller. Also, the energy efficiency (IAEW) is much better with the FF-FB-C controller than with the FB-C controller.

Test 2 The second test consists of steps and steady states in both SOG and course, testing the performance both at relatively low and high speeds. The trajectories are shown in Figure 7, while the performance metrics are shown i Figure 8. Again, it is clear that the FF-FB-C controller has much better transient response than the FB-C controller, both in SOG and course. The FF-FB-C controller also has the best steady-state performance. In this scenario, all the performance metrics are in favor of the FF-FB-C controller. Notice in particular the control precision (IAE) and energy efficiency (IAEW) which are 2–3 times better with the FF-FB-C controller than with the FB-C controller.

The performance metrics for both tests are summarized in Table 2.

Table 2. Performance metrics for both controllers in both tests. The controller with the best performance for each metric in each test is highlighted in bold.

Test case	**Controller**	**IAE**	**IADC**	**IAE-ADC**	**IAEW**
Test 1	FF-FB-C	**8.9**	$1.4\cdot10^{-1}$	**1.3**	$\mathbf{1.7\cdot10^{4}}$
	FB-C	$2.3\cdot10^{1}$	$\mathbf{1.2\cdot10^{-1}}$	2.8	$3.9\cdot10^{4}$
Test 2	FF-FB-C	$\mathbf{1.1\cdot10^{1}}$	$\mathbf{7.1\cdot10^{-2}}$	$\mathbf{8.1\cdot10^{-1}}$	$\mathbf{1.5\cdot10^{4}}$
	FB-C	$3.2\cdot10^{1}$	$1.3\cdot10^{-1}$	4.1	$4.1\cdot10^{4}$

5. CONCLUSION AND FURTHER WORK

We have proposed a SOG and course controller for high-speed ASVs operating in the displacement, semi-displacement and planing regions. The controller employs model-based feedforward terms, based on the FF-FB velocity controller developed in (Eriksen and Breivik, 2017a). Through full-scale experiments in the Trondheimsford, Norway, the controller is compared to a PI feedback controller with gain scheduling. From the experiments, it is clear that using model-based feedforward terms in combination with feedback terms greatly improve the control performance compared to using a pure feedback controller. The performance is particularly improved for time-varying references.

The proposed speed and course controller has subsequently been successfully used for full-scale closed-loop COLAV experiments during the autumn of 2017 (Eriksen and Breivik, 2018). In future work, we would also like to use the proposed controller for other kinematic control applications, such as for example path following and trajectory tracking.

ACKNOWLEDGEMENTS

This work was supported by the Research Council of Norway through project number 244116 and the Centres of Excellence funding scheme with project number 223254. The authors would like to express great gratitude to Kongsberg Maritime and Maritime Robotics for providing high-grade navigation technology and the Telemetron ASV at our disposal for the experiments.

REFERENCES

Bishop, C.M. (2006). *Pattern Recognition and Machine Learning*. Springer Science + Business Media.

Breivik, M. (2010). *Topics in Guided Motion Control of Marine Vehicles*. Ph.D. thesis, Norwegian University og Science and Technology (NTNU), Trondheim, Norway.

Breivik, M., Hovstein, V.E., and Fossen, T.I. (2008). Straight-line target tracking for unmanned surface vehicles. *Modeling, Identification and Control*, 29(4), 131–149.

Eriksen, B.-O.H. and Breivik, M. (2017a). *Modeling, Identification and Control of High-Speed ASVs: Theory and Experiments*, 407–431. Springer International Publishing, Cham.

Eriksen, B.-O.H. and Breivik, M. (2017b). MPC-based mid-level collision avoidance for ASVs using nonlinear programming. In *Proc. of IEEE CCTA*. Kohala Coast, Hawai'i, USA.

Eriksen, B.-O.H. and Breivik, M. (2018). The branching course MPC algorithm for maritime collision avoidance. *Journal of Field Robotics - To be submitted.*

Eriksen, B.-O.H., Wilthil, E.F., Flåten, A.L., Brekke, E.F., and Breivik, M. (2018). Radar-based maritime collision avoidance using dynamic window. In *Proc. of IEEE Aerospace*. Big Sky, MT, USA.

Faltinsen, O.M. (2005). *Hydrodynamics of High-Speed Marine Vehicles*. Cambridge University Press.

Fossen, T.I. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons Ltd.

Kuwata, Y., Wolf, M.T., Zarzhitsky, D., and Huntsberger, T.L. (2014). Safe maritime autonomous navigation with COLREGS, using velocity obstacles. *IEEE J. Oceanic Eng.*, 39(1), 110–119.

Prince, W.G. and Bishop, R.E.D. (1974). *Probabilistic Theory of Ship Dynamics*. Chapman and Hall.

Sørensen, M.E.N., Bjørne, E.S., and Breivik, M. (2016). Performance comparison of backstepping-based adaptive controllers for marine surface vessels. In *Proc. of IEEE CCA*. Buenos Aires, Argentina.

Sørensen, M.E.N. and Breivik, M. (2015). Comparing nonlinear adaptive motion controllers for marine surface vessels. In *Proc. of the 10th IFAC MCMC*. Copenhagen, Denmark.

Paper F The branching-course MPC algorithm for maritime collision avoidance

Preprint of **B.-O. H. Eriksen**, M. Breivik, E. F. Wilthil, A. L. Flåten, and E. F. Brekke, “The branching-course MPC algorithm for maritime collision avoidance”, 2019, Accepted for publication in Journal of Field Robotics, available at https://arxiv.org/abs/1907.00039.

The Branching-Course MPC Algorithm for Maritime Collision Avoidance

Bjørn-Olav H. Eriksen Morten Breivik Erik F. Wilthil Andreas L. Flåten Edmund F. Brekke
Centre for Autonomous Marine Operations and Systems
Department of Engineering Cybernetics
Norwegian University of Science and Technology (NTNU)
Trondheim, Norway
{bjorn-olav.h.eriksen, morten.breivik}@ieee.org
{erik.wilthil, andreas.flaten, edmund.brekke}@ntnu.no

Abstract

This article presents a new algorithm for short-term maritime collision avoidance (COLAV) named the branching-course MPC (BC-MPC) algorithm. The algorithm is designed to be robust with respect to noise on obstacle estimates, which is a significant source of disturbance when using exteroceptive sensors such as e.g. radars for obstacle detection and tracking. Exteroceptive sensors do not require vessel-to-vessel communication, which enables COLAV toward vessels not equipped with e.g. automatic identification system (AIS) transponders, in addition to increasing the robustness with respect to faulty information which may be provided by other vessels. The BC-MPC algorithm is compliant with rules 8, 13 and 17 of the International Regulations for Preventing Collisions at Sea (COLREGs), and favors maneuvers following rules 14 and 15. Specifically, the algorithm can ignore the specific maneuvering regulations of rules 14 and 15, which may be required in situations where rule 17 revokes a stand-on obligation. The algorithm is experimentally validated in several full-scale experiments in the Trondheimsfjord in 2017 using a radar-based system for obstacle detection and tracking. To complement the experimental results, we present simulations where the BC-MPC algorithm is tested in more complex scenarios involving multiple obstacles and several simultaneously active COLREGs rules. The COLAV experiments and simulations show good performance.

1 Introduction

Today's society moves rapidly towards an increased level of automation. The development of autonomous cars is spearheading this trend, as exemplified by the efforts made by e.g. Google and Uber. In recent years, autonomy has also become a hot topic in the maritime domain with research on autonomous passenger and goods transport, seabed surveying and military applications. An example of this is the Yara Birkeland project in Norway, where an autonomous electrically-powered cargo ship will replace approximately 40000 diesel-powered truck journeys of fertilizer per year (Kongsberg Maritime, 2018). Reduced cost, increased efficiency and reduced environmental impact may be the most obvious benefits of autonomy at sea, but the potential for increased safety is not to be overlooked since reports state that in excess of 75% of maritime accidents are caused by human errors (Chauvin, 2011; Levander, 2017). A prerequisite for employing autonomous surface vehicles (ASVs) in environments where other vessels may be present is, however, that the ASVs have robust collision avoidance (COLAV) systems. Such COLAV systems must make the ASVs, as other vessels, follow the International Regulations for Preventing Collisions at Sea (COLREGs) which contains a set of rules on how vessels should behave in situations where there is a risk of collision with another vessel (Cockcroft and Lameijer, 2004). However, COLREGs is written for human interpretation with few quantitative rules, which makes it challenging to develop algorithms capturing the intention of COLREGs by machine decision-making.

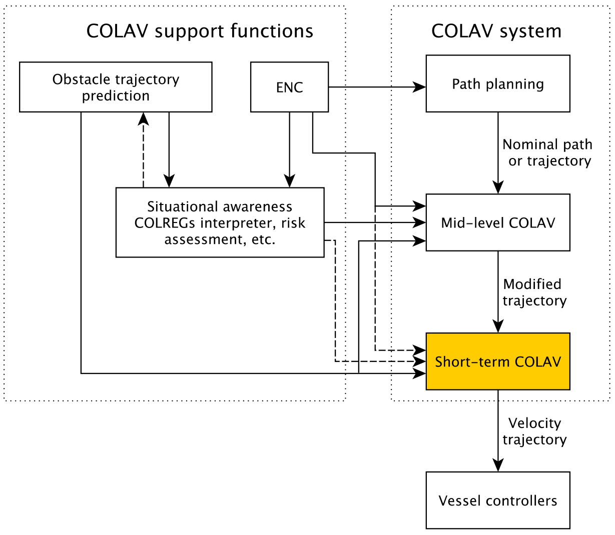


Figure 1: A hybrid COLAV architecture with three levels. The support functions provide relevant information for the COLAV algorithms, including obstacle trajectories, static obstacles from electronic nautical charts (ENC) and situational awareness in the form of COLREGs situations. The short-term layer does not currently utilize information from ENC or situational awareness.

COLAV algorithms have typically been divided into reactive and deliberate algorithms. Reactive algorithms are characterized by considering a limited amount of information, originally only currently available sensor information (Tan et al., 2004), and employing little motion planning in a short time frame. This makes reactive algorithms computationally cheap, and able to react to sudden changes in the environment. Examples include vessels making sudden unpredicted maneuvers, late detection of obstacles, etc. However, since reactive algorithms consider a limited amount of information and employ little motion planning, they tend to make suboptimal choices in complex situations which makes them sensitive to local minima. Examples of reactive algorithms are the velocity obstacles (VO) (Fiorini and Shiller, 1998; Kuwata et al., 2014) and the dynamic window (DW) (Fox et al., 1997) algorithms. Deliberate algorithms consider more information and plan for a longer time frame, which results in more optimal choices at the cost of increased computational requirements. Examples of deliberate algorithms include the A* (Hart et al., 1968) and the rapidly exploring random tree (RRT) (LaValle, 1998) algorithms.

The previously clear border between reactive and deliberate algorithms have become somewhat artificial since few algorithms only utilize currently available sensor information. However, the idea that the reactive algorithms are capable of responding quickly to changes in the environment and the deliberate algorithms are capable of performing optimal motion planning in a longer time frame is still relevant. We therefore choose to rather use the terms “short-term” and “long-term” algorithms to distinguish the algorithms. In a practical COLAV system, both short-term and long-term algorithms are useful. For long time frames, all available information should be included, while one may use a less detailed vessel model for planning. For short-term COLAV, one can include less spatial and temporal information but may need to use a more detailed model of the vessel to ensure dynamically feasible maneuvers. By combining short-term and long-term algorithms in a hybrid architecture, the benefits of both algorithms can be combined, ensuring both responsiveness, feasibility and optimality. An example of a hybrid architecture with three COLAV levels is shown in Fig. 1. The topmost level, named path planning, is intended to produce a nominal path or trajectory from the initial position to the goal. The spatial and temporal distance between the initial and goal positions may be large, allowing only for a limited complexity in this algorithm. For instance, moving obstacles could be neglected at this level. The mid-level COLAV algorithm tries to follow this nominal path or trajectory, while at the same time performing COLAV with respect to all obstacles, characterized as a long-term COLAV algorithm. COLREGs is a natural part of this level, since it may be complex to decide the appropriate action with respect to COLREGs. The mid-level algorithm produces a modified trajectory which is passed to the short-term COLAV layer. This layer performs short-time COLAV making sure to avoid obstacles performing sudden maneuvers or which are detected too late to be handled by the mid-level algorithm, while also ensuring that the maneuvers are feasible with respect to the dynamic constraints of the vessel. The short-term layer can also act as a backup solution to avoid collisions in cases where the mid-level algorithm fails to produce feasible trajectories, for instance due to time constraints or numerical issues (Eriksen and Breivik, 2017b). Furthermore, the short-term layer should be able to avoid collision in emergency situations, e.g. when obstacles does not maneuver in accordance with COLREGs.

COLAV algorithms depend on information about obstacle position, speed and course in order to be able to avoid collisions. One possible source of such information is using automatic identification system (AIS) transponders. AIS is a vessel-to-vessel communication system where vessels transmit their current position and velocity to other vessels carrying AIS transponders (IMO, 2018). Passenger ships and vessels with a gross tonnage of over 300 are required to carry AIS transponders. This is of course valuable information when it comes to navigation and COLAV at sea. However, AIS transponders usually rely on satellite navigation and data inputs from the user, which results in the possibility of transmitting inaccurate or invalid data (Harati-Mokhtari et al., 2007). Also, vessels or objects not equipped with AIS transponders will not be detected. A more robust approach to obtain information about the environment is to employ exteroceptive sensors, which have the advantage of not relying on any infrastructure or collaboration with the obstacles in order to detect them. A commonly used exteroceptive sensor at sea is radar. However, the data from a radar usually includes a fair amount of noise, which makes this sensor more complex and difficult to work with than AIS (Eriksen et al., 2018). On-board radars have been used for full-scale COLAV experiments based on the A* algorithm in (Schuster et al., 2014), and using a modified version of the DW algorithm in (Eriksen et al., 2018). In (Elkins et al., 2010; Kuwata et al., 2014), other exteroceptive sensors such as cameras and lidar are used for COLAV.

Model predictive control (MPC) has for a long time been a well-known and proven tool for motion planning and COLAV for e.g. ground and automotive robots (Ögren and Leonard, 2005; Keller et al., 2015; Gray et al., 2013), aerospace applications (Kuwata and How, 2011) and underwater vehicles (Caldwell et al., 2010). In the later years, MPC has also been applied for COLAV in the maritime domain, both using sample-based approaches where one considers a finite space of control inputs (Švec et al., 2013; Johansen et al., 2016; Hagen et al., 2018) and conventional gradient-based search algorithms (Abdelaal and Hahn, 2016; Eriksen and Breivik, 2017b). None of these algorithms does, however, consider the amounts of noise which we expect to encounter using a radar-based tracking system. Gradient-based algorithms have the benefit of exploring the entire control input space, but the complexity of the COLAV problem can make it difficult to guarantee that a feasible solution will be found within the time requirements (Eriksen and Breivik, 2017b). This makes sample-based approaches well suited for short-term COLAV. In (Benjamin et al., 2006; Benjamin et al., 2010), a protocol-based COLAV algorithm using interval programming is presented. The algorithm optimizes over multiple functions considering different behaviors, e.g. waypoint following and adherence to different parts of COLREGs, by combining them in an objective function with adaptive weights. The algorithm does, however, use vessel-to-vessel communication in order to obtain obstacle information, and is not necessarily well suited for use with exteroceptive sensors.

1.1 The International Regulations for Preventing Collisions at Sea

COLREGs regulate how vessels should behave in situations where there exists a risk of collision. There are in total 38 rules, where rules 8 and 13–17 are the most relevant ones for designing COLAV algorithms for ASVs, although the rest must also must be addressed in a COLREGs-compliant system. Rules 8 and 13–17 can be summarized as:

Rule 8: This rule requires, among other things, that maneuvers applied in situations where a risk of collision exists should be large enough to be readily observable for other vessels. Small consecutive maneuvers should hence be avoided.

Rule 13: In an overtaking situation, where a vessel is approaching another from an angle of more than $22.5°$ abaft the other vessel's beam, the overtaking vessel is required to keep out of the way of the overtaken vessel. The

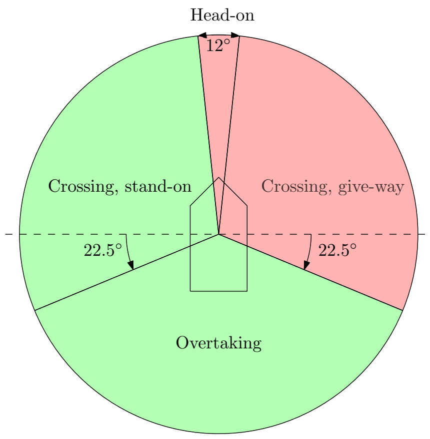


Figure 2: Graphical illustration of COLREGs regions as seen from the ownship. The light red regions show areas where the ownship is required to maneuver, while the light green region show areas where the ownship should keep the current speed and course.

overtaking vessel is allowed to pass on either side. However, in a case where the overtaken vessel is required to avoid collision with another vessel it may be required to make a starboard maneuver. To avoid blocking the path of the overtaken vessel in such a situation, we consider it as most suitable to overtake a vessel on her port side.

Rule 14: In a head-on situation, where two vessels approaches each other on reciprocal or nearly reciprocal courses (a margin of ± 6 ° is often used), both vessels are required to do starboard maneuvers and pass the other vessel on her port side.

Rule 15: This rule handles crossing situations, where a vessel is approaching another vessel from the side, but not in the regions considered as a head on or overtaking situation. The vessel with the other vessel on her starboard side is deemed the give-way vessel, while the other is deemed the stand-on vessel. The preferred give-way maneuver is to do a starboard turn and pass behind the stand-on vessel.

Rule 16: This rule defines the action for the give-way vessel. It requires that the give-way vessel performs early and substantial action to avoid collision.

Rule 17: This rule defines the action for the stand-on vessel. It requires that the stand-on vessel keep her current speed and course, while the give-way vessel maneuvers in order to avoid collision. However, if the give-way vessel fails in her duty of avoiding collision, the stand-on vessel is required to maneuver such as best aids to avoid collision. If this occurs in a crossing situation, the stand-on vessel should avoid maneuvering to port if possible.

Fig. 2 shows a graphical illustration of the situations given by rules 13–15. The interested reader is referred to (Cockcroft and Lameijer, 2004) for more details on the COLREGs rules.

1.2 Contributions

The authors of this article have focused on short-term and reactive COLAV for ASVs for the last few years, starting with a modified version of the DW algorithm designed for use with autonomous underwater vehicles (AUVs) (Eriksen et al., 2016). This algorithm was adapted for use with high-speed ASVs, and tested in conjunction with a radar-based tracking system (Wilthil et al., 2017) successfully demonstrating closed-loop radar-based COLAV in full-scale experiments (Eriksen et al., 2018). However, the experiments revealed challenges with using radar-based tracking systems for COLAV, especially noisy estimates of obstacle speed and course caused problems. The DW algorithm is not particularly robust with respect to such noise, causing the vessel to repeatedly change the planned maneuver. In addition, the DW algorithm assumes the ASV to keep a constant turn rate for the entire prediction horizon. This does not resemble the way vessels usually maneuver at sea, where one usually performs a corrective maneuver by changing the course and/or speed, followed by keeping the speed and course constant. These issues motivate us to develop a new short-term COLAV algorithm which is less sensitive to noisy obstacle estimates while also producing more "maritime-like" maneuvers.

In this article, we therefore present a new algorithm for short-term COLAV named the branching-course MPC (BC-MPC) algorithm. This algorithm is based on sample-based MPC and is designed to be robust with respect to noisy obstacle estimates, which is an important consideration when using radar-based tracking systems for providing obstacle estimates. In contrast to sample-based MPC algorithms previously applied to ASVs, the BC-MPC algorithm considers a sequence of maneuvers, enabling the algorithm to plan more complex trajectories than just a single avoidance maneuver. Furthermore, the BC-MPC algorithm complies with rules 8, 13 and 17 of COLREGs, while favoring maneuvers complying with rules 14 and 15. In cases where the algorithm chooses to ignore the maneuvering aspects of rules 14 and 15, which can be required when rule 17 revokes a stand-on obligation, the maneuvers have increased clearance to obstacles. The term "COLREGs-compliance" is often abused in the literature by using it for algorithms only complying with parts of COLREGs. With this in mind, we consider the algorithm as being partly COLREGs compliant, and well suited to handle the short-term aspects in a COLREGs-compliant hybrid COLAV architecture. The algorithm is implemented on an under-actuatuated ASV and validated through several full-scale closed-loop COLAV experiments using a radar-based tracking system for providing estimates of obstacle course, speed and position. To complement the experimental results, we present simulation results where the algorithm is tested in multi-obstacle scenarios where multiple COLREGs rules apply simultaneously.

1.3 Outline

The rest of the article is structured as follows: Section 2 describes modeling and control of ASVs, Section 3 presents the BC-MPC algorithm, while Section 4 contains results from the full-scale closed-loop COLAV experiments. To complement the experimental results, we present simulation results of more complex scenarios in Section 5, including scenarios with multiple and maneuvering obstacles. Finally, Section 6 concludes the article and presents possibilities for further work.

2 ASV modeling and control

The vessel of interest in this work is the Telemetron ASV shown in Fig. 3, which is owned and operated by Maritime Robotics (MR). The vessel is $8.45\ \mathrm{m}$ long, and uses a single steerable outboard engine for propulsion, which makes the vessel underactuated.

2.1 ASV modeling

ASVs are in general small and agile vessels, capable of operating at high speeds. At low speeds the hydrostatic pressure mainly carries the weight of the vessel, and it operates in the displacement region. When the vessel speed increases, the hydrodynamic pressure increases, eventually dominating over the hydrostatic pressure. At this point, we are in the planing region. In between the displacement and planing region we have the semi-displacement region. The Telemetron ASV is a high-speed vessel, capable of speeds up to $18\ \mathrm{m/s}$, which combined with the vessel length of $8.45\ \mathrm{m}$ makes for a vessel operating in the displacement, semi-displacement and planing regions (Fossen, 2011; Faltinsen, 2005).

The conventional approach to modeling ASVs is by using the 3DOF model (Fossen, 2011):

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\nu} \tag{1a}$$

Figure 3: The Telemetron ASV, designed for both manned and unmanned operations. Courtesy of Maritime Robotics.

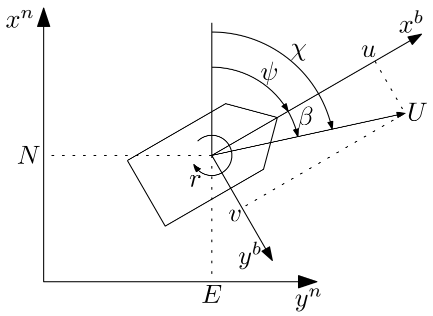


Figure 4: Vessel variables. The superscripts $(\cdot)^n$ and $(\cdot)^b$ denote the NED and body reference frames (Fossen, 2011), respectively. The variables N, E and ψ represent the vessel pose, u, v and r represent the body-fixed vessel velocity and U is the vessel speed over ground. The course χ is the sum of the heading ψ and the sideslip β.

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}, \tag{1b}$$

where $\boldsymbol{\eta} = \begin{bmatrix} N & E & \psi \end{bmatrix}^T$ is the vessel pose in an earth-fixed North-East-Down (NED) reference frame, $\boldsymbol{\nu} = \begin{bmatrix} u & v & r \end{bmatrix}^T$ is the vessel velocity and $\boldsymbol{\tau} = \begin{bmatrix} X & Y & N \end{bmatrix}^T$ is a vector of forces and torque, both given in the body-fixed reference frame. See Fig. 4 for an illustration of the variables. The matrix $\boldsymbol{R}(\psi)$ is a rotation matrix, while $\boldsymbol{M}$, $\boldsymbol{C}(\boldsymbol{\nu})$ and $\boldsymbol{D}(\boldsymbol{\nu})$ are the mass, Coriolis and centripetal and damping matrices, respectively.

There exist many versions of the model (1) (Fossen, 2011), but they require that the vessel operates in the displacement region. For the Telemetron ASV, this would require a maximum operating speed of approximately $3.5\ \mathrm{m/s}$ (Eriksen and Breivik, 2017a). This is quite a big limitation, and we therefore rather use a control-oriented non-first principles model developed for high-speed ASVs (Eriksen and Breivik, 2017a), valid for the displacement, semi-displacement and planing regions:

$$\boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}} + \boldsymbol{\sigma}(\boldsymbol{x}) = \boldsymbol{\tau}, \tag{2}$$

where $\boldsymbol{x} = \begin{bmatrix} U & r \end{bmatrix}^T$ is the vessel state, with $U = \sqrt{u^2 + v^2}$ being the vessel speed over ground and r being the vessel yaw rate, while $\boldsymbol{\tau} = \begin{bmatrix} \tau_m & \tau_\delta \end{bmatrix}^T$ is a normalized control input. In this article, we also refer to the vessel speed over ground as the vessel speed. The matrix $\boldsymbol{M}(\boldsymbol{x})$ is a diagonal state-dependent inertia matrix with nonlinear terms and $\boldsymbol{\sigma}(\boldsymbol{x}) = \begin{bmatrix}\sigma_U(\boldsymbol{x}) & \sigma_r(\boldsymbol{x})\end{bmatrix}^T$ is a vector of nonlinear damping terms. Notice that the model is in 2DOF, designed for underactuated ASVs, where the speed and course are usually controlled. Using the state variable from (2), the kinematics can be defined as:

$$\begin{aligned}\dot{\boldsymbol{\eta}} &= \begin{bmatrix}\cos(\chi) & 0\\ \sin(\chi) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}U\\ r\end{bmatrix}\\ \dot{\chi} &= r + \dot{\beta},\end{aligned} \tag{3}$$

where χ is the vessel course and β is the sideslip. For more details on the model, see (Eriksen and Breivik, 2017a).

2.2 ASV control design

As shown in Fig. 1, the COLAV system is built on top of the vessel controllers. Hence, the performance of the COLAV system can be limited by the performance of the vessel controllers. It is therefore beneficial to use high-performance vessel controllers ensuring that the maneuvers that the COLAV system specifies are properly executed, not limiting the performance of the COLAV system.

The model (2) can be used in control design, particularly using it for model-based feedforward in speed and yaw rate is shown to provide good performance (Eriksen and Breivik, 2017a). A controller named the feedforward feedback (FF-FB) controller is presented in (Eriksen and Breivik, 2017a), which combines model-based feedforward terms with a gain-scheduled proportional-integral (PI) feedback controller for controlling the vessel speed and yaw rate. For the BC-MPC algorithm, we need a controller capable of following a speed and course trajectory. The FF-FB controller has proven to have high performance in experiments (Eriksen et al., 2018; Eriksen and Breivik, 2017a), so we therefore extend the FF-FB controller to include course control:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}}_d + \boldsymbol{\sigma}(\boldsymbol{x}_d) - \boldsymbol{M}(\boldsymbol{x})\boldsymbol{K}_p\tilde{\boldsymbol{\zeta}} - \boldsymbol{K}_i\int_{t_0}^{t}\tilde{\boldsymbol{\zeta}}_1(\gamma)\mathrm{d}\gamma, \tag{4}$$

where $\boldsymbol{x}_d = \begin{bmatrix}U_d & r_d\end{bmatrix}^T$, $\boldsymbol{K}_p > 0$ is a matrix of proportional gains, $\boldsymbol{K}_i > 0$ is a diagonal matrix of integral gains, and:

$$\tilde{\boldsymbol{\zeta}} = \begin{bmatrix}\tilde{U}\\ \tilde{r}\\ \tilde{\chi}\end{bmatrix}, \quad \tilde{\boldsymbol{\zeta}}_1 = \begin{bmatrix}\tilde{U}\\ \tilde{\chi}\end{bmatrix}, \tag{5}$$

where $\tilde{U} = U - U_d$, $\tilde{r} = r - r_d$ and $\tilde{\chi} = \Upsilon(\chi - \chi_d)$ are the speed, yaw rate and course errors, respectively. The function $\Upsilon : \mathbb{R} \to S^1$ maps an angle to the domain $[-\pi, \pi)$.

In the control law (4), we use the desired yaw rate r_d and its derivative $\dot{r}_d$. Through (3), the relation between the course and yaw rate is stated as $r = \dot{\chi} - \dot{\beta}$, where the derivative of the sideslip enters the equation. At this stage, we do not have a sideslip model of the Telemetron ASV. However, we have seen in experiments that at moderate speeds the sideslip is sufficiently constant to be neglected without major implications. We therefore simplify the relation by assuming constant sideslip and defining the desired yaw rate and its derivative as:

$$\begin{aligned}r_d &= \dot{\chi}_d\\ \dot{r}_d &= \ddot{\chi}_d.\end{aligned} \tag{6}$$

The interested reader is referred to (Eriksen and Breivik, 2018) for more details on the speed and course controller.

3 The BC-MPC algorithm

The BC-MPC algorithm is intended to avoid collisions with moving obstacles while respecting the dynamic constraints of the vessel in order to ensure feasible maneuvers, which is ideal for short-term COLAV. The algorithm is based on

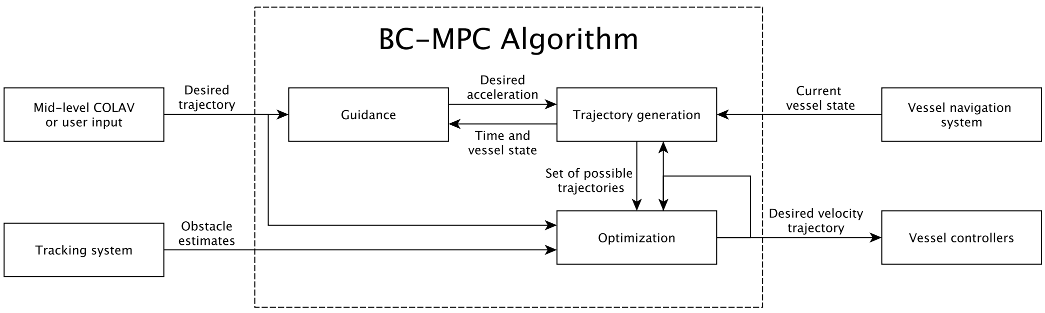


Figure 5: BC-MPC algorithm overview. The algorithm inputs a desired trajectory from a mid-level COLAV algorithm or an operator, obstacle estimates from a tracking system and the current vessel state from a navigation system, and outputs a desired velocity trajectory for the vessel controllers.

model predictive control (MPC), and plans vessel-feasible trajectories with multiple maneuvers where only the first maneuver is executed. The trajectories have continuous acceleration, which is beneficial for vessel controllers utilizing model-based feedforward terms, such as (4). To fit well with tracking systems based on exteroceptive sensors, such as e.g. radars, the algorithm is designed to be robust with respect to noisy obstacle estimates. Furthermore, the algorithm is designed with the short-term perspective of COLREGs in mind, namely situations where the stand-on requirement may need to be ignored in order to avoid collision in compliance with rule 17. The algorithm is also modular, so it can easily be tailored for different applications.

The BC-MPC algorithm can be described by two steps, which will be explained in detail in the following sections:

1. Generate a search space consisting of feasible trajectories with respect to the dynamic constraints of the vessel.

2. Discretize the search space and compute an objective function value on the trajectories. The optimal trajectory is then selected as the one with the lowest objective function value.

The BC-MPC algorithm architecture is shown in Fig. 5. The algorithm inputs a desired trajectory, which can originate from either another COLAV algorithm or directly from a user. The guidance function receives the desired trajectory, and computes a desired acceleration given a vessel state and time specified by the trajectory generation. The trajectory generation block creates a set of possible vessel trajectories, given an initial vessel state, initial desired velocity and a desired acceleration from the guidance function. A tracking system provides obstacle estimates, which are used to calculate a part of the objective function. The optimization block computes the optimal trajectory based on an objective function, and outputs this as a desired velocity trajectory to the vessel controller (4).

3.1 Trajectory generation

The search space consists of a number of trajectories, each consisting of a sequence of sub-trajectories each containing one maneuver. Having multiple maneuvers in each trajectory enables the algorithm to consider complex scenarios which may require a time-limited speed and/or course change, before selecting a new speed and/or course. In addition, it will allow the algorithm to consider a complete avoidance situation, consisting of an evasive maneuver and a plan for converging back to the desired trajectory. Each trajectory is defined by a desired velocity trajectory containing a speed and course trajectory with continuous acceleration, and feedback-corrected predicted pose and velocity trajectories.

3.1.1 Trajectory generation: A single step

As mentioned, each trajectory consists of a sequence of maneuvers, resulting in trajectories that branches out from each other. Hence, the trajectory generation can be divided in repeatable steps. At each step, a set of sub-trajectories, each containing one maneuver, are computed given an initial vessel configuration, initial time and some step-specific parameters:

- The number of speed maneuvers N_U
- The number of course maneuvers N_χ
- The time allowed for changing the actuator input, named the ramp time T_{ramp}
- The maneuver time length in speed T_U and course T_χ
- The total step time length T

We start by generating the desired velocity trajectories, which should be feasible with respect to actuator rate and magnitude saturations. To ensure feasibility with respect to the actuator rate saturations, we start from the model (2) by calculating the possible speed and course accelerations given our current configuration as:

$$\begin{aligned}\dot{\boldsymbol{X}}_{\max} &= \boldsymbol{M}^{-1}\left(\boldsymbol{\tau}_{\max} - \boldsymbol{\sigma}(\boldsymbol{X}_0)\right)\\ \dot{\boldsymbol{X}}_{\min} &= \boldsymbol{M}^{-1}\left(\boldsymbol{\tau}_{\min} - \boldsymbol{\sigma}(\boldsymbol{X}_0)\right),\end{aligned} \tag{7}$$

where $\dot{\boldsymbol{X}}_{\max} = \begin{bmatrix}\dot{U}_{\max} & \dot{r}_{\max}\end{bmatrix}^T$, $\dot{\boldsymbol{X}}_{\min} = \begin{bmatrix}\dot{U}_{\min} & \dot{r}_{\min}\end{bmatrix}^T$, $\boldsymbol{X}_0$ is the current vessel velocity and:

$$\begin{aligned}\boldsymbol{\tau}_{\max} &= \text{sat}\left(\boldsymbol{\tau}_0 + T_{\text{ramp}}\dot{\boldsymbol{\tau}}_{\max}, \boldsymbol{\tau}_{\min}, \boldsymbol{\tau}_{\max}\right)\\ \boldsymbol{\tau}_{\min} &= \text{sat}\left(\boldsymbol{\tau}_0 + T_{\text{ramp}}\dot{\boldsymbol{\tau}}_{\min}, \boldsymbol{\tau}_{\min}, \boldsymbol{\tau}_{\max}\right),\end{aligned} \tag{8}$$

where $T_{\text{ramp}} > 0$ is the ramp time, $\boldsymbol{\tau}_0$ is the current control input, $\boldsymbol{\tau}_{\max}$ and $\boldsymbol{\tau}_{\min}$ are the maximum and minimum control input, respectively, and $\dot{\boldsymbol{\tau}}_{\max}$ and $\dot{\boldsymbol{\tau}}_{\min}$ are the maximum and minimum control input rate of change, respectively. The saturation function sat$(\boldsymbol{a}, \boldsymbol{a}_{\min}, \boldsymbol{a}_{\max})$ is defined as sat : $\mathbb{R}^K \times \mathbb{R}^K \times \mathbb{R}^K \to \mathbb{R}^K$ with:

$$a_i^* = \begin{cases} a_{\min,i} & , a_i < a_{\min,i}\\ a_{\max,i} & , a_i > a_{\max,i}\\ a_i & , \text{otherwise,}\end{cases} \tag{9}$$

for $\boldsymbol{a}^* = \text{sat}(\boldsymbol{a}, \boldsymbol{a}_{\min}, \boldsymbol{a}_{\max})$, $i \in \{1, 2, \ldots, K\}$ and $(\cdot)_i$ denoting element i of a vector. Following this, we create a set of possible accelerations as:

$$A_d = \left\{(\dot{U}, \dot{r}) \in \mathbb{R} \times \mathbb{R} \middle| \dot{U} \in [\dot{U}_{\min}, \dot{U}_{\max}], \dot{r} \in [\dot{r}_{\min}, \dot{r}_{\max}]\right\}. \tag{10}$$

The set of possible accelerations is then sampled uniformly to create a discrete set of candidate maneuver accelerations:

$$\begin{aligned}\dot{\boldsymbol{U}}_{\text{samples}} &= \left\{\dot{U}_1, \dot{U}_2, \ldots, \dot{U}_{N_U}\right\}\\ \dot{\boldsymbol{r}}_{\text{samples}} &= \left\{\dot{r}_1, \dot{r}_2, \ldots, \dot{r}_{N_\chi}\right\},\end{aligned} \tag{11}$$

where $\dot{U}_i$, $i \in [1, N_U]$ are speed acceleration samples and $\dot{r}_i$, $i \in [1, N_\chi]$ are course acceleration samples. To be able to include a specific maneuver in the search space, which can be beneficial e.g. to converge to a specific desired trajectory, we allow to modify some of the sampled accelerations if a desired acceleration $(\dot{U}_d', \dot{r}_d')$ is inside the set of possible accelerations as follows: If $\dot{U}_d' \in A_d$, we change the closest speed acceleration sample in $\dot{\boldsymbol{U}}_{\text{samples}}$ to $\dot{U}_d'$. Similarly for course, if $\dot{r}_d' \in A_d$, we change the closest course acceleration sample in $\dot{\boldsymbol{r}}_{\text{samples}}$ to $\dot{r}_d'$. Following this, we create a set of candidate maneuver accelerations by combining the speed and course candidate maneuvers as

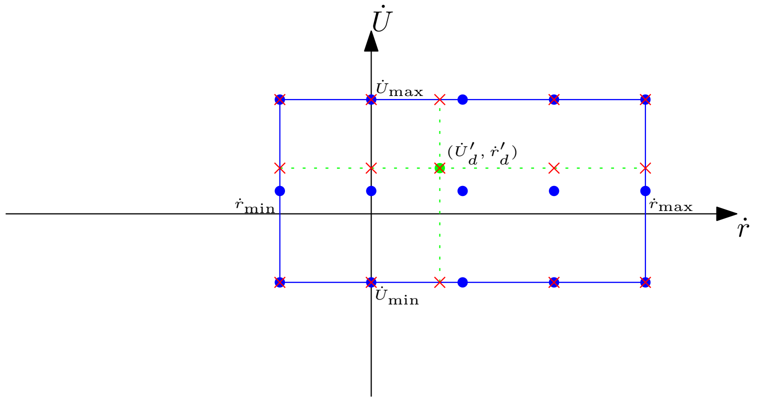


Figure 6: Set of possible accelerations shown with the blue line, with initial samples shown as blue circles. The desired acceleration $(\dot{U}'_d, \dot{r}'_d)$ is shown as a green circle, while the final samples are shown as red crosses.

$\dot{\boldsymbol{U}}_{\text{samples}} \times \dot{\boldsymbol{r}}_{\text{samples}}$. This concept is illustrated in Fig. 6, where A_d is sampled with $N_U = 3$ speed samples and $N_\chi = 5$ course samples.

Given the acceleration samples, we create a set of N_U motion primitives for speed based on the piecewise-linear speed acceleration trajectories:

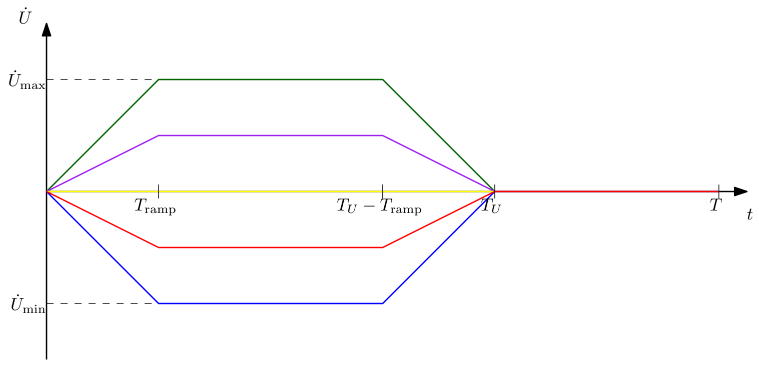
$$\dot{U}_{d,i}(t) = \begin{cases} k_{U,i}t & , 0 \leq t < T_{\text{ramp}} \\ \dot{U}_i & , T_{\text{ramp}} \leq t < T_U - T_{\text{ramp}} \\ \dot{U}_i - k_{U,i}(t - (T_U - T_{\text{ramp}})) & , T_U - T_{\text{ramp}} \leq t < T_U \\ 0, & , T_U \leq t \leq T, \end{cases} \tag{12}$$

where $k_{U,i} = \frac{\dot{U}_i}{T_{\text{ramp}}}$, $\dot{U}_i$ is the sampled acceleration for speed motion primitive $i \in [1, N_U]$, $T_U > 0$ is the speed maneuver length and $T > 0$ is the total trajectory length. Similarly, we define N_χ course motion primitives by the piecewise-linear course acceleration trajectories:

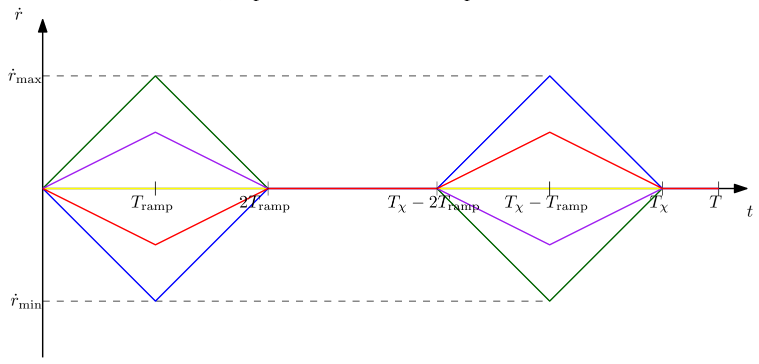
$$\dot{r}_{d,i}(t) = \begin{cases} k_{r,i}t & , 0 \leq t < T_{\text{ramp}} \\ 2\dot{r}_i - k_{r,i}t & , T_{\text{ramp}} \leq t < 2T_{\text{ramp}} \\ 0 & , 2T_{\text{ramp}} \leq t < T_\chi - 2T_{\text{ramp}} \\ -k_{r,i}(t - (T_\chi - 2T_{\text{ramp}})) & , T_\chi - 2T_{\text{ramp}} \leq t < T_\chi - T_{\text{ramp}} \\ -2\dot{r}_i + k_{r,i}(t - (T_\chi - T_{\text{ramp}})) & , T_\chi - T_{\text{ramp}} \leq t < T_\chi \\ 0 & , 2T_\chi \leq t < T, \end{cases} \tag{13}$$

where $k_{r,i} = \frac{\dot{r}_i}{T_{\text{ramp}}}$, $\dot{r}_i$ is the sampled acceleration for course motion primitive $i \in [1, N_\chi]$ and $T_\chi > 0$ is the course maneuver length. For notational simplicity and without loss of generality, we assumed zero initial time $t_0 = 0$ in (12) and (13). The acceleration trajectories and parameters for $N_U = 5$ speed motion primitives and $N_\chi = 5$ course motion primitives are illustrated in Fig. 7. Notice that the integral of the course acceleration maneuvers are zero, hence if the maneuver is initialized with zero yaw rate the maneuver will end with zero yaw rate. The motion primitives (12) and (13) are chosen as linear piecewise functions in order to ensure a continuous acceleration with a minimum complexity.

Based on the acceleration trajectories, we create trajectories for the desired speed, yaw rate and course by integrating



(a) Speed acceleration motion primitives



(b) Course acceleration motion primitives. Note that the integral of each course acceleration trajectory is zero.

Figure 7: Acceleration motion primitives, where T is the step time, T_{ramp} denotes the ramp time while T_U and T_χ are the speed and course maneuver time lengths, respectively.

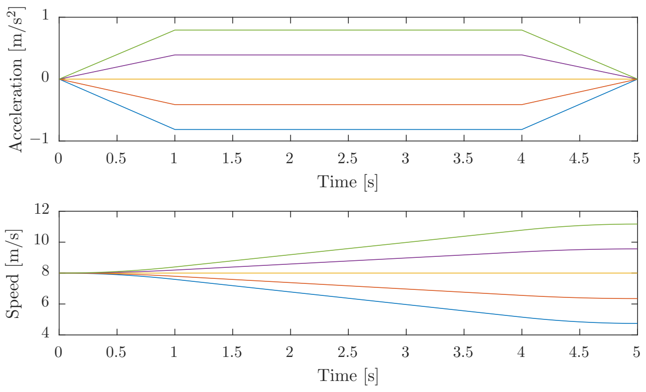


Figure 8: Example of $N_U = 5$ speed trajectories with $T_{\text{ramp}} = 1$ s and $T = T_U = 5$ s. Acceleration in the top plot and speed in the bottom plot.

the expressions (12) and (13) as:

$$\begin{aligned} U_{d,i}(t) &= U_{d,0} + \int_{t_0}^{t} \dot{U}_{d,i}(\gamma)\mathrm{d}\gamma, \quad i \in [1, N_U] \\ r_{d,i}(t) &= r_{d,0} + \int_{t_0}^{t} \dot{r}_{d,i}(\gamma)\mathrm{d}\gamma, \quad i \in [1, N_\chi] \\ \chi_{d,i}(t) &= \chi_{d,0} + \int_{t_0}^{t} r_{d,i}(\gamma)\mathrm{d}\gamma, \quad i \in [1, N_\chi]. \end{aligned} \tag{14}$$

The initial values $U_{d,0}$, $r_{d,0}$ and $\chi_{d,0}$ are taken as the corresponding desired values from the last BC-MPC iteration (or sub-trajectory, if computing trajectories for subsequent maneuvers), such that the desired trajectories passed to the vessel controllers are continuous. This implies that we do not include feedback in the desired trajectories. Furthermore, as in Section 2, the vessel sideslip is neglected. This could, however, be included by using a vessel model including sideslip. A numerical example of 5 speed and 5 course trajectories is shown in figures 8 and 9, where a maneuver length of 5 s is used for both speed and course. Vessels at sea usually maneuver by either keeping a constant speed and course or by performing a speed and/or course change and continuing with this new speed and course for some time. By selecting the initial yaw rate in (14) as $r_{d,0} = 0$ we ensure that maneuvers start and end with constant-course motion, which mimics this behavior while also producing maneuvers that should be readily observable for other vessels, as required by rule 8 of COLREGs.

Following this, we create a union set of the desired velocity trajectories as:

$$\mathcal{U}_d = \{U_{d,1}(t), U_{d,2}(t), \ldots, U_{d,N_U}(t)\} \times \{\chi_{d,1}(t), \chi_{d,2}(t), \ldots, \chi_{d,N_\chi}(t)\}, \tag{15}$$

resulting in a total of $N_U \cdot N_\chi$ desired velocity trajectories. Notice that the speed trajectories in $\mathcal{U}_d$ are continuously differentiable, while the course trajectories are twice continuously differentiable. Velocity trajectories containing infeasible steady-state vessel velocities are removed from $\mathcal{U}_d$ by checking the feasibility using the vessel model (2) together with the actuator saturation constraints.

Given the desired velocity trajectories, we calculate the feedback-corrected pose trajectories. To do this, we first predict the resulting speed and course trajectories, $\bar{U}_i(t)$, $i \in [1, N_U]$ and $\bar{\chi}_i(t)$, $i \in [1, N_\chi]$, respectively. This is done by simulating the closed-loop error dynamics of the vessel and vessel controllers using the desired velocity trajectories as the input. In this article, we approximate the error dynamics using first order linear models, which may seem as quite rough approximations. However, this is justified by noting that the model-based speed and course controller demonstrates very good control performance for the Telemetron ASV, resulting in small control errors (Eriksen and

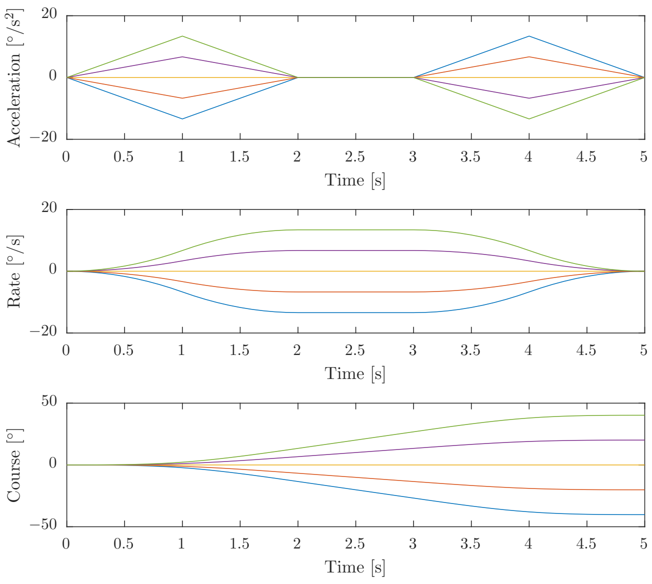


Figure 9: Example of $U_\chi = 5$ course trajectories with $T_{\text{ramp}} = 1$ s and $T = T_\chi = 5$ s. Acceleration in the top plot, rate in the middle and course in the bottom plot.

Breivik, 2018). Furthermore, the control errors are dominated by environmental disturbances, which is difficult to model without increasing the complexity to an unnecessarily high level. The closed-loop error models are given as:

$$\begin{aligned}\dot{\tilde{U}} &= \frac{1}{T_{\tilde{U}}}\tilde{U} \\ \dot{\tilde{\chi}} &= \frac{1}{T_{\tilde{\chi}}}\tilde{\chi},\end{aligned} \tag{16}$$

where $\tilde{U} = \bar{U} - U_d$, $\tilde{\chi} = \bar{\chi} - \chi_d$, and $T_{\tilde{U}} > 0$ and $T_{\tilde{\chi}} > 0$ are time constants. The time constants can be heuristically determined through simulations and experiments. By solving (16), the predicted speed and course trajectories are found as:

$$\begin{aligned}\bar{U}_i(t) &= \tilde{U}_0 \mathrm{e}^{-\frac{1}{T_{\tilde{U}}}(t-t_0)} + U_{d,i}(t), \quad i \in [1, N_U] \\ \bar{\chi}_i(t) &= \tilde{\chi}_0 \mathrm{e}^{-\frac{1}{T_{\tilde{\chi}}}(t-t_0)} + \chi_{d,i}(t), \quad i \in [1, N_\chi],\end{aligned} \tag{17}$$

where $\tilde{U}_0 = U_0 - U_{d,0}$ and $\tilde{\chi}_0 = \chi_0 - \chi_{d,0}$ introduces feedback in the prediction through the current vessel speed and course, U_0 and χ_0, respectively. Similarly as (15), we construct a set of predicted velocity trajectories:

$$\bar{\mathcal{U}} = \{\bar{U}_1(t), \bar{U}_2(t), \ldots, \bar{U}_{N_U}(t)\} \times \{\bar{\chi}_1(t), \bar{\chi}_2(t), \ldots, \bar{\chi}_{N_\chi}(t)\}. \tag{18}$$

Combinations of speed and course trajectories that was considered infeasible when forming $\mathcal{U}_d$ are also removed from $\bar{\mathcal{U}}$. Following this, vessel position trajectories $\bar{\boldsymbol{p}}(t) = \begin{bmatrix}\bar{N}(t) & \bar{E}(t)\end{bmatrix}^T$ are calculated from the predicted velocity trajectories using a kinematic model:

$$\dot{\bar{\boldsymbol{p}}} = \begin{bmatrix}\cos(\bar{\chi}) \\ \sin(\bar{\chi})\end{bmatrix} \bar{U}, \tag{19}$$

which is integrated using the current vessel position as the initial condition. The feedback-corrected predicted vessel pose trajectories are finally combined in the set $\bar{\mathcal{H}}$ as:

$$\bar{\mathcal{H}} = \left\{\bar{\boldsymbol{\eta}}(t; \bar{U}(t), \bar{\chi}(t)) \middle| (\bar{U}(t), \bar{\chi}(t)) \in \bar{\mathcal{U}}\right\}, \tag{20}$$

where $\bar{\boldsymbol{\eta}} = \begin{bmatrix}\bar{N}(t) & \bar{E}(t) & \bar{\chi}(t)\end{bmatrix}^T$.

To summarize, a single step of a trajectory is defined by the set of desired velocity trajectories $\mathcal{U}_d$, the set of predicted velocity trajectories $\bar{\mathcal{U}}$ and the set of set of predicted pose trajectories $\bar{\mathcal{H}}$.

3.1.2 Trajectory generation: The full trajectory generation

A full trajectory consists of multiple sub-trajectories, each containing one maneuver and constructed using the single-step procedure. This naturally forms a tree structure, with nodes representing vessel states and edges representing sub-trajectories. The depth of the tree will be equal to the desired number of maneuvers in each trajectory. The tree is is initialized with the initial state as the root node, which the single-step procedure is performed on, generating a number of sub-trajectories and leaf nodes. Following this, the the single-step procedure is performed on each of the leaf nodes, adding the next sub-trajectory and leaf nodes to the existing trajectories and expanding the tree depth. This procedure is repeated until the tree has the desired depth, resulting in each trajectory having the desired number of maneuvers. Using the same number of speed and course maneuvers at each level would result in the tree growing exponentially with the number of levels. To limit the growth, we therefore allow for choosing a different number of speed and course maneuvers at each level, for instance keeping the speed constant in all levels except the first, only allowing the speed to be changed during the first maneuver of a trajectory.

The remaining parameters can also be chosen differently for each level, and in principle the acceleration trajectories (12) and (13) can also be designed using different structures. However, we choose to use the same acceleration trajectory structure for each level, while also keeping the ramp time and maneuver time lengths constant. This leaves only the step time length and number of speed and course maneuvers as parameters that can change throughout the tree depth. Choosing different step time lengths can be considered as an MPC input blocking scheme, requiring that the step lengths are integer dividable by the algorithm sample time.

A full trajectory generation can hence be defined by the following parameters:

- An initial vessel state including the current desired velocity.
- The number of maneuvers in each trajectory, or levels, defined as $B > 0$.
- The step time at each level $\boldsymbol{T} = \begin{bmatrix}T_1 & T_2 & \dots & T_B\end{bmatrix}$, the ramp time T_{ramp} and the speed and course maneuver lengths T_U and T_χ, respectively.
- The number of speed maneuvers at each level $\boldsymbol{N}_U = \begin{bmatrix}N_{U,1} & N_{U,2} & \dots & N_{U,B}\end{bmatrix}$.
- The number of course maneuvers at each level $\boldsymbol{N}_\chi = \begin{bmatrix}N_{\chi,1} & N_{\chi,2} & \dots & N_{\chi,B}\end{bmatrix}$.

A set of predicted vessel pose trajectories with $B = 3$ levels is shown in Fig. 10. The step time is chosen as $\boldsymbol{T} = \begin{bmatrix}5\text{ s} & 10\text{ s} & 10\text{ s}\end{bmatrix}$, making the trajectories 25 s long in total. The trajectories have 5 course maneuvers at the first level and three in the later levels, while there for illustrational purposes are only one speed maneuver at each step. Hence, $\boldsymbol{N}_U = \begin{bmatrix}1 & 1 & 1\end{bmatrix}$ and $\boldsymbol{N}_\chi = \begin{bmatrix}5 & 3 & 3\end{bmatrix}$. The ramp time and maneuver lengths are chosen as $T_{\text{ramp}} = 1$ s and $T_U = T_\chi = 5$ s, respectively. Notice that the maneuver length of 5 s results in the second and third maneuver having a straight-course segment after the turn, which increases the prediction horizon without increasing the computational load while also increasing the maneuver observability.

Selecting the trajectory generation parameters is a complex task, and it is difficult to provide a general guideline on how to do this. However, we attempt to provide some thoughts and insight on this. Increasing the number of maneuvers in each trajectory B increases how complex solutions the algorithm look for, and also increases the computational requirements. In general, most COLAV situations should be able to solve by a few maneuvers, and selecting three maneuvers will allow the algorithm to plan for maneuvering out from a desired trajectory, turning parallel to the trajectory and returning towards the trajectory. The step time of each maneuver $\boldsymbol{T}$ controls the length of the prediction

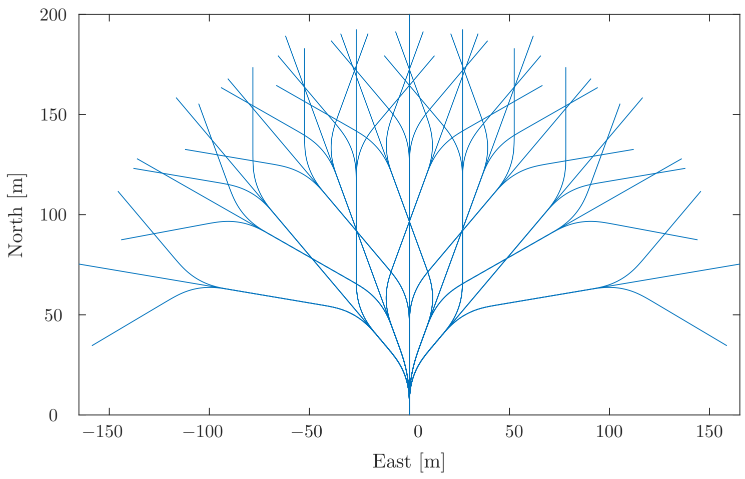


Figure 10: A set of predicted pose trajectories with 3 levels.

horizon, and should be selected long enough to cover the expected situations to be handled. The ramp time T_{ramp}, and the speed and course maneuver lengths T_U and T_χ should be selected such that the vessel of interest is capable of making maneuvers of appropriate magnitude. This implies that vessels with slow dynamics should have longer maneuver times than vessels with fast dynamics. Since the motion primitives are symmetric, the number of speed and course maneuvers $\boldsymbol{N}_U$ and $\boldsymbol{N}_\chi$ should be selected as odd numbers in order to ensure that keeping constant speed and course is included in the search space. The actual number of maneuvers should be selected such that the course and speed deviation between the specific maneuvers are large enough to provide observable maneuvers. Utilizing simulations with a model of the vessel of interest will be highly useful when deciding on the trajectory prediction parameters.

3.1.3 Calculating a desired acceleration

In the single-step trajectory generation, a desired acceleration $(\dot{U}'_d, \dot{r}'_d)$ can be used to include a desired maneuver in the search space. We therefore use a guidance algorithm to ensure that there exists a trajectory in the search space that converges towards the desired trajectory inputted to the BC-MPC algorithm. To achieve this, we use a modified version of a path tracking algorithm ensuring vessel convergence to a curved path (Breivik and Fossen, 2004). The control law is based on line of sight (LOS) guidance (Fossen, 2011), together with defining a desired point on the path which the velocity of is controlled, named the path particle (PP). The desired course is stated as:

$$\chi_{d,LOS} = \chi_{path} + \arctan\left(-\frac{e}{\Delta}\right), \tag{21}$$

where χ_{path} is the path angle at the desired point, e is the cross-track error and $\Delta > 0$ is the lookahead distance. The path particle velocity along the path is stated as:

$$U_{PP} = U\cos(\chi - \chi_{path}) + \gamma_s s, \tag{22}$$

where U is the vessel speed, $\gamma_s > 0$ is a tuning parameter and s is the along-track distance. The guidance scheme is illustrated in Fig. 11. The control law (22) controls the speed along the path U_{PP} as a function of the vessel speed, course and the along-track distance to the path particle, letting the vessel converge towards the path with a constant speed. We rather want to be able to follow a desired trajectory by controlling the vessel speed and course based on the desired trajectory. We therefore fix the path particle at the desired position on the trajectory, given the current time,

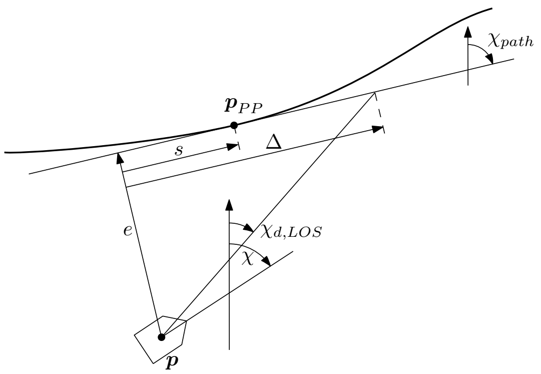


Figure 11: LOS guidance scheme. The path particle propagates along the path with the speed U_{pp}. The vessel course is denoted as χ, while χ_{path} denote the current tangential path course and $\chi_{d,LOS}$ denote the desired course. The variables e, s and Δ are the cross-track error, along-track distance and the lookahead distance, respectively.

and by reformulating (22) we obtain a desired vessel speed given the trajectory velocity:

$$U_{d,LOS} = \begin{cases} \text{sat}\left(\frac{U_t - \gamma_s s}{\cos(\chi - \chi_{path})}, 0, U_{\max,LOS}\right) & \text{if } |\cos(\chi - \chi_{path})| > \epsilon \\ \text{sat}\left(\frac{U_{pp} - \gamma_s s}{\epsilon}, 0, U_{\max,LOS}\right) & \text{else,} \end{cases} \tag{23}$$

where U_t is the trajectory velocity and $\epsilon > 0$ is a small constant to avoid division by zero. The saturation function ensures that the desired vessel speed is in the interval $[0, U_{\max}]$, where $U_{\max} > 0$ is the maximum vessel operating speed. Given a desired speed and course, we compute the desired speed and course acceleration:

$$\begin{aligned} \dot{U}_d' &= \frac{U_{d,LOS} - U_{d,0}}{T_U - T_{\text{ramp}}} \\ \dot{r}_d' &= \frac{\chi_{d,LOS} - \chi_{d,0}}{T_{\text{ramp}}\left(T_\chi - 2T_{\text{ramp}}\right)}, \end{aligned} \tag{24}$$

which are found by solving (14) for the final desired speed and course. Notice that in cases where there is only one speed and/or course maneuver, the corresponding desired acceleration should be selected as zero to keep a constant speed and/or course.

The obvious singularity in (23) when the vessel course is perpendicular to the desired trajectory (and hence $\cos(\chi - \chi_{path} = 0)$ is handled by avoiding division by zero and ensuring that the desired speed is inside the possible operating speed of the vessel, which makes it difficult to guarantee stability and convergence of this guidance scheme. However, the desired acceleration is only used to modify some trajectories in the BC-MPC search space, and will hence not constrain the algorithm to choose a trajectory based on (24). One could employ other schemes, e.g. (Paliotta, 2017) which guarantees convergence to curved trajectories. This does, however, increase the complexity by depending on a detailed 3DOF model of the vessel while also employing a feedback-linearizing controller to control the vessel. It is in general difficult to obtain detailed models of high-speed ASVs, while time delays, sensor noise and modeling uncertainties are shown to cause robustness issues when using feedback-linearizing controllers (Eriksen and Breivik, 2017a). Hence, the simplicity of (21)–(23) is appealing when a guarantee of stability and convergence is not required.

3.2 Selecting the optimal trajectory

Given the set of feasible trajectories, we solve an optimization problem to select the optimal trajectory. We start by defining a cost function to assign a cost to each trajectory:

$$G(\bar{\boldsymbol{\eta}}(t), \boldsymbol{u}_d(t); \boldsymbol{p}_d(t)) = w_{\text{al}}\text{align}(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_d(t)) + w_{\text{av}}\text{avoid}(\bar{\boldsymbol{\eta}}(t)) + w_{\text{t}}\text{tran}(\boldsymbol{u}_d(t)), \tag{25}$$

where $(\bar{\boldsymbol{\eta}}(t), \boldsymbol{u}_d(t))$ is the predicted vessel pose and desired velocity of a candidate trajectory, $\text{align}(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_d(t))$ measures the alignment between the predicted pose trajectory and a desired trajectory $\boldsymbol{p}_d(t)$, $\text{avoid}(\bar{\boldsymbol{\eta}}(t))$ assigns cost to trajectories traversing close to obstacles, while $\text{tran}(\boldsymbol{u}(t))$ introduces transitional cost in the objective function to avoid wobbly behavior. The parameters $w_{\text{al}}, w_{\text{av}}, w_{\text{t}} \geq 0$ are tuning parameters to control the weighting of the different objective terms, which can be selected heuristically by simulating the algorithm to obtain the desired behavior. In general, the avoidance weight $w_{\text{av}} >> w_{\text{al}}$ in order to ensure that the algorithm prioritizes avoiding obstacles over following the desired trajectory, while w_{t} can be tuned to control how responsive, and sensitive to noise, the algorithm will be.

Using (25), we define the optimization problem:

$$\boldsymbol{u}_d^*(t) = \underset{(\bar{\boldsymbol{\eta}}_k(t), \boldsymbol{u}_{d,k}(t)) \in (\bar{\mathcal{H}}, \mathcal{U}_d)}{\text{argmin}} \quad G(\bar{\boldsymbol{\eta}}_k(t), \boldsymbol{u}_{d,k}(t); \boldsymbol{p}_d(t)), \tag{26}$$

where $\boldsymbol{u}_d^*(t)$ is the optimal desired velocity trajectory to be used as the reference for the vessel controllers. The optimization problem is solved by simply calculating the cost over the finite discrete set of trajectories and choosing the one with the lowest cost.

The next sections describe the different terms of the objective function (25). Notice that we strive to avoid using discontinuities and logic in order to improve the robustness with respect to obstacle estimate noise.

3.2.1 Trajectory alignment

The alignment between the desired trajectory and a candidate trajectory is used in the objective function (25) to motivate the algorithm to follow the desired trajectory. Given a desired trajectory $\boldsymbol{p}_d(t) : \mathrm{R}^+ \to \mathrm{R}^2$, required to be C^1, we obtain a desired course as:

$$\chi_d(t) = \text{atan2}(\dot{E}_d(t), \dot{N}_d(t)), \tag{27}$$

with $\boldsymbol{p}_d(t) = \begin{bmatrix} N_d(t) & E_d(t) \end{bmatrix}^T$. Given this, we define a weighted metric of Euclidean distance and orientation error as:

$$\text{align}(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_d(t)) = \int_{t_0}^{t_0 + T_{\text{full}}} \left(w_p \left\| \begin{bmatrix} \bar{N}(\gamma) \\ \bar{E}(\gamma) \end{bmatrix} - \boldsymbol{p}_d(\gamma) \right\|_2 + w_\chi |\Upsilon\left(\bar{\chi}(\gamma) - \chi_d(\gamma)\right)| \right) \mathrm{d}\gamma, \tag{28}$$

where $w_p, w_\chi > 0$ are weights controlling the influence of the Euclidean and angular error, respectively, $T_{\text{full}} = \sum_{i=1}^{B} T_i$ denotes the entire trajectory prediction horizon.For simplicity, we fix $w_p = 1$ and leave w_χ and w_{al} to control the weighting.

3.2.2 Obstacle avoidance

Obstacle avoidance is achieved by penalizing candidate trajectories with small distances to obstacles. We define three regions around the obstacles, named the collision, safety and margin regions, respectively. The idea behind this is to make it possible use different gradients on the penalty depending on how close the ownship is to the obstacle. This, together with avoiding logic and discontinuities, should improve the robustness with respect to noise on the obstacle estimates.

We define a time-varying vector between obstacle i and a predicted vessel trajectory as:

$$\boldsymbol{r}_i(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_i(t)) = \boldsymbol{p}_i(t) - \begin{bmatrix} \bar{N}(t) \\ \bar{E}(t) \end{bmatrix}, \tag{29}$$

where $\boldsymbol{r}_i = \begin{bmatrix} r_{N,i} & r_{E,i} \end{bmatrix}^T$ and $\boldsymbol{p}_i(t)$ is the position of obstacle i at time t. The obstacle position in future time is computed under the common assumption that obstacles will keep their current speed and course (Johansen et al., 2016; Kuwata et al., 2014; Eriksen et al., 2018), which is a reasonable assumption for relatively short time periods. More complex techniques can also be applied for predicting the future position of obstacles, for instance based on historic

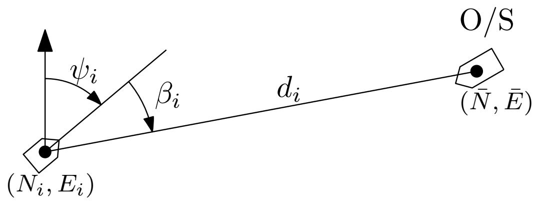


Figure 12: Distance d_i and relative bearing β_i to obstacle i. The ownship is marked O/S.

AIS data (Dalsnes et al., 2018) or by estimating the turn rate of the obstacles (Flåten and Brekke, 2017). Using (29), we define the distance and relative bearing to obstacle i given a predicted vessel trajectory $\bar{\boldsymbol{\eta}}(t)$ as:

$$\begin{aligned} d_i(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_i(t)) &= \|\boldsymbol{r}_i(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_i(t))\|_2 \\ \beta_i(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_i(t)) &= \Upsilon\left(\text{atan2}\big(r_{E,i}(\bar{\boldsymbol{\eta}}(t)), r_{N,i}(\bar{\boldsymbol{\eta}}(t))\big) - \chi_i(t)\right), \end{aligned} \tag{30}$$

where $\chi_i(t)$ is the course of obstacle i, calculated as $\chi_i(t) = \text{atan2}\left(\dot{E}_i(t), \dot{N}_i(t)\right)$ with $\boldsymbol{p}_i(t) = \begin{bmatrix} N_i(t) & E_i(t) \end{bmatrix}^T$. The distance d_i and relative bearing β_i are illustrated in Fig. 12.

The obstacle distance and relative bearing is used to calculate a penalty function, which we use to define the avoidance function as:

$$\text{avoid}(\bar{\boldsymbol{\eta}}(t)) = \sum_{i=1}^{M} \int_{t=t_0}^{t_0+T_{\text{full}}} w_i(\gamma)\text{penalty}_i(\bar{\boldsymbol{\eta}}(\gamma))\text{d}\gamma, \tag{31}$$

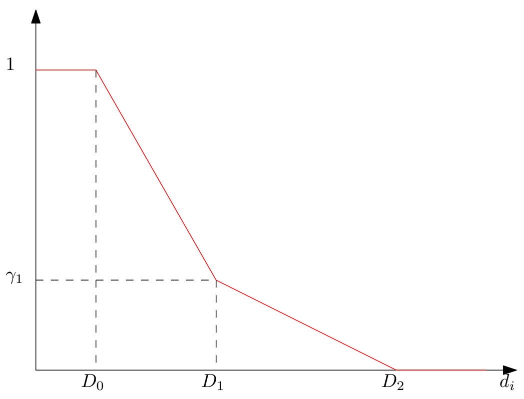
where M is the number of obstacles, $\text{penalty}_i(\bar{\boldsymbol{\eta}}(t))$ assigns a penalty to the predicted vessel trajectory $\bar{\boldsymbol{\eta}}(t)$ at time t with respect to obstacle i, while $w_i(t)$ are time and obstacle dependent weights. The weights can be useful for prioritizing vessels in multi-obstacle situations where properties like vessel type, size, speed, etc. can be used for differentiating the importance of avoiding the given vessels in severe situations. The weights can also facilitate time-dependent weighting, for instance as a heuristic method to incorporate uncertainty on obstacle estimates, combined with obstacle and time-dependent scaling of the obstacle region sizes. For simplicity, we keep the weights constant at $w_i(t) = 1\,\forall\, i$.

The penalty function can be designed in a variety of ways, with the simplest possibly being a circular penalty function. When using a circular penalty function, the relative bearing to the obstacle does not matter, and the function can be defined as:

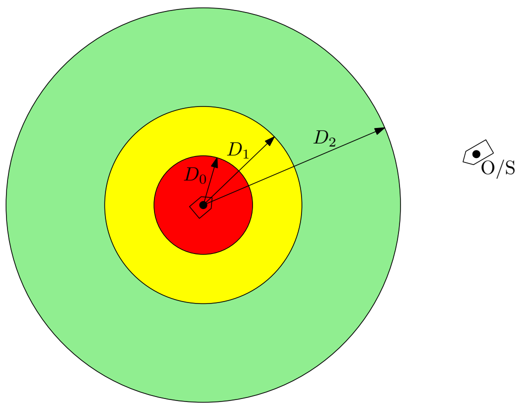
$$\text{penalty}_{i,\text{circular}}(\bar{\boldsymbol{\eta}}(t)) = \begin{cases} 1 & \text{if } d_i < D_0 \\ 1 + \frac{\gamma_1 - 1}{D_1 - D_0}(d_i - D_0) & \text{if } D_0 \leq d_i < D_1 \\ \gamma_1 - \frac{\gamma_1}{D_2 - D_1}(d_i - D_1) & \text{if } D_1 \leq d_i < D_2 \\ 0 & \text{else}, \end{cases} \tag{32}$$

where the parameters of $d_i(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_i(t))$ are omitted for notational simplicity. The variables $D_2 > D_1 > D_0 > 0$ are the margin, safety and collision region sizes, respectively, while $\gamma_1 \in (0, 1)$ is a tuning parameter controlling the cost gradient inside the margin and safety regions. The circular penalty function is illustrated in Fig. 13.

A circular penalty function is useful for static objects where there is no preference on which side of the object one should pass. For moving vessels, it should be considered to be more dangerous to be in front of the vessel than on the side or behind it, and COLREGs also introduce preferences on which side one should pass an obstacle. An intuitive approach to handle COLREGs would be to use logic to decide the applicable rule with respect to each obstacle, but this conflicts with with the idea of designing the algorithm with high robustness to noisy obstacle estimates. Also noting that the BC-MPC algorithm is intended to be used in a hybrid architecture with a mid-level algorithm taking a more proactive approach to the COLREGs rules, we here focus our attention towards a smooth and continuous approximation. In a short-term COLAV perspective, it is not beneficial to constrain the algorithm to strictly follow the head-on and crossing rules (rules 14 and 15), since rule 17 may require maneuvers ignoring these rules in cases where it



(a) Function value.



(b) Function regions. The red region is the collision region, yellow is the safety region and green is the margin region, given by the radiuses D_0, D_1 and D_2, respectively.

Figure 13: The circular penalty function value (a) and regions (b).

revokes the stand-on requirement. However, the algorithm should choose maneuvers compliant with rule 14, and rules 13 and 15 when this is possible. We therefore motivate the algorithm to choose maneuvers complying with rules 13–15 by defining an elliptical COLREGs penalty function by letting the region sizes D_0, D_1 and D_2 be dependent on the relative bearing. Such area-based ship domains are widely used in COLAV algorithms, also for handling COLREGs (Szlapczynski and Szlapczynska, 2017). Each region is defined by a combination of three elliptical and one circular segment as:

$$D_k(\beta_i) = \begin{cases} b_k & \text{if } \beta_i < -\frac{\pi}{2} \\ \frac{a_k b_k}{\sqrt{(b_k \cos\beta_i)^2 + (a_k \sin\beta_i)^2}} & \text{if } -\frac{\pi}{2} \leq \beta_i < 0 \\ \frac{a_k c_k}{\sqrt{(c_k \cos\beta_i)^2 + (a_k \sin\beta_i)^2}} & \text{if } 0 \leq \beta_i < \frac{\pi}{2} \\ \frac{b_k c_k}{\sqrt{(c_k \cos\beta_i)^2 + (b_k \sin\beta_i)^2}} & \text{if } \frac{\pi}{2} \leq \beta_i, \end{cases} \tag{33}$$

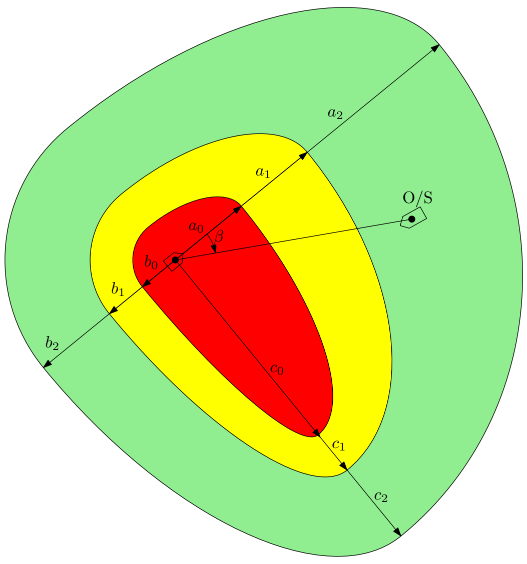
where a_k, b_k and $c_k = b_k + d_{\text{COLREGs}}$ with $k \in \{0, 1, 2\}$ define the major and minor ellipses axes. The parameter $d_{\text{COLREGs}} > 0$ controls the region expansion of the starboard side of the obstacle. The regions are illustrated in Figure 14a.

If we were to use (32) with D_k from (33) as the elliptical COLREGs penalty function, the entire collision region would have a constant penalty. This poses a potential problem since all points inside the region is considered to be equally costly. For the circular penalty function, this region is so small that the impact is quite low. For the elliptical COLREGs penalty function, however, it is natural to have a non-constant cost inside the collision region since this is rather large. We therefore define the elliptical COLREGs penalty function as:

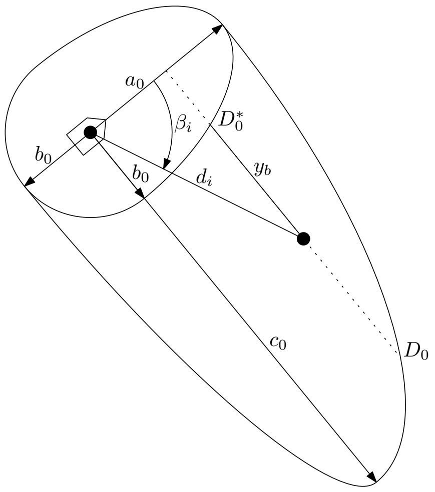
$$\text{penalty}_{i,\text{COLREGs}}(\bar{\boldsymbol{\eta}}(t)) = \text{inner\_penalty}_i(\bar{\boldsymbol{\eta}}(t)) + \begin{cases} 1 & \text{if } d_i < D_0 \\ 1 + \frac{\gamma_1 - 1}{D_1 - D_0}(d_i - D_0) & \text{if } D_0 \leq d_i < D_1 \\ \gamma_1 - \frac{\gamma_1}{D_2 - D_1}(d_i - D_1) & \text{if } D_1 \leq d_i < D_2 \\ 0 & \text{else,} \end{cases} \tag{34}$$

where D_k, $k \in \{0, 1, 2\}$ are given by (33) and $\text{inner\_penalty}_i(\bar{\boldsymbol{\eta}}(t))$ is an additional cost inside the collision region. This additional cost is given as:

$$\text{inner\_penalty}_i(\bar{\boldsymbol{\eta}}(t)) = \begin{cases} 1 & \text{if } d_i < D_0^* \\ 1 - \frac{y_b(d_i, \beta_i)}{d_{\text{COLREGs}}} & \text{if } D_0^* \leq d_i < D_0, \\ 0 & \text{else,} \end{cases} \tag{35}$$



(a) Function regions, each constructed by one circular and three elliptical segments. At a given distance from the obstacle, the elliptical form imposes a higher cost on the starboard side and in front of an obstacle, compared to the port side side and abaft an obstacle. This motivates the BC-MPC algorithm to pass on the port side and abaft the obstacle.



(b) Illustration of how y_b is calculated given a point (d_i, β_i). The outer boundary is the collision region boundary, as shown in red in (a), and D_0^* is the left-hand collision region boundary mirrored about the obstacle surge axis.

Figure 14: Elliptical COLREGs penalty function regions (a) and illustration of y_b for the inner penalty function (b).

where D_0^* given as:

$$D_0^*(\beta_i) = \begin{cases} \frac{a_0 b_0}{\sqrt{(b_0 \cos \beta_i)^2 + (a_0 \sin \beta_i)^2}} & \text{if } |\beta_i| < \frac{\pi}{2} \\ b_0 & \text{else,} \end{cases} \tag{36}$$

and $y_b(d_i, \beta_i)$ is the distance from the D_0^* region to the point (d_i, β_i) along the y-direction of the obstacle body frame, as illustrated in Figure 14b.

The actual parameters of the obstacle function should be selected such that the safety region represent the desired clearance, while the collision region represents the absolute minimum clearance required. The margin region should be selected as the distance when we want the ownship to initiate a maneuver, and should be quite much larger than the safety region. This, together with a quite small obstacle gradient parameter γ_1, will make the algorithm less sensitive towards fluctuating estimates of obstacle position, speed and course. To reduce the number of parameters to select, we consider that the clearance in front of the obstacle should be twice that behind the ship, hence $a_i = 2b_i, i \in \{0, 1, 2\}$. The COLREGs distance d_{COLREGs} controls how strict the maneuvering aspects of rules 14 and 15 are enforced.

3.2.3 Transitional cost

An important design criteria for the algorithm is that it should be robust with respect to noise on the obstacle estimates, making it well suited for use with tracking systems based on exteroceptive sensors. By introducing transitional cost in the objective function, a certain level of cost reduction will be required to make the algorithm change the current planned maneuver. This should increase the robustness to noise on the obstacle estimates, while also making the algorithm less affected by noise in the vessel state estimates and external disturbances, for instance wave induced motion.

Denoting the desired velocity trajectory from the previous iteration as $\boldsymbol{u}_d^-(t)$, which is currently being tracked by the vessel controllers, the transitional cost is computed as:

$$\operatorname{tran}(\boldsymbol{u}_d(t)) = \begin{cases} 1 & \text{if } \int_{t_0}^{t_0+T_1} \left|U_d(\gamma) - U_d^-(\gamma)\right| \mathrm{d}\gamma > e_{U,\min} \text{ or } \int_{t_0}^{t_0+T_1} \left|\chi_d(\gamma) - \chi_d^-(\gamma)\right| \mathrm{d}\gamma > e_{\chi,\min} \\ 0 & \text{else,} \end{cases} \tag{37}$$

with $\boldsymbol{u}_d(t) = \begin{bmatrix} U_d(t) & \chi_d(t) \end{bmatrix}^T$, $\boldsymbol{u}_d^-(t) = \begin{bmatrix} U_d^-(t) & \chi_d^-(t) \end{bmatrix}^T$ and where T_1 is the step time of the first trajectory maneuver. The variables $e_{U,\min}$ and $e_{\chi,\min}$ denote the minimum speed and course difference between the previous desired velocity trajectory and the candidates:

$$\begin{aligned} e_{U,\min} &= \min_{\boldsymbol{u}_d(t) \in \mathcal{U}_d} \int_{t_0}^{t_0+T_1} \left|U_d(\gamma) - U_d^-(\gamma)\right| \mathrm{d}\gamma \\ e_{\chi,\min} &= \min_{\boldsymbol{u}_d(t) \in \mathcal{U}_d} \int_{t_0}^{t_0+T_1} \left|\chi_d(\gamma) - \chi_d^-(\gamma)\right| \mathrm{d}\gamma. \end{aligned} \tag{38}$$

The transitional cost term is zero if the first maneuver of the candidate desired velocity trajectory $\boldsymbol{u}_d(t)$ is the one closest to the desired velocity trajectory from the previous iteration $\boldsymbol{u}_d^-(t)$, and one otherwise. Notice that the transitional cost term introduces discontinuities, which we previously stated that we would like to avoid in order to improve the robustness with respect to noise on obstacle estimates. The transitional cost term does, however, not rely on obstacle estimates, making the term insensitive to noise on the obstacle estimates and justifying the use of a discontinuous transitional cost function.

4 Experimental results

Full-scale experiments were conducted in the Trondheimsfjord, Norway, on the 12th of October 2017. This section describes the experimental setup and results.

4.1 Experimental setup

The Telemetron ASV, briefly introduced in Section 2, was used as the ownship. The vessel is fitted with a SIMRAD Broadband 4G$^{\text{TM}}$ Radar, and a Kongsberg Seatex Seapath 330+ GNSS-aided inertial navigation system was used during the experiments. See Table 1 for more details on the vessel specifications. The BC-MPC algorithm was implemented in discrete time using the Euler method to discretize the algorithm, see Table 2 for the algorithm parameters. The parameters was mostly selected heuristically through simulations of the algorithm and the vessel of interest, as described in Section 3. We inputted a user-specified straight line trajectory with constant speed as the desired trajectory, and used the elliptical COLREGs penalty function for obstacle avoidance. The BC-MPC algorithm was run at a rate of 0.2 Hz.

The implementation consists of a radar-based tracking system to provide obstacle estimates, the BC-MPC algorithm and the model-based speed and course controller described in section 2.2 for low-level vessel control. The system was implemented on a processing platform with an Intel® i7 3.4 GHz CPU running Ubuntu 16.04 Linux, using the Robot Operating System (ROS) (Quigley et al., 2009). Fig. 15 shows the implementation architecture.

The tracking system receives spoke detections from the radar through a UDP interface. The detections are transformed to a local reference frame and clustered together to form one measurement per obstacle, which is a common assumption for many tracking algorithms. The obstacle measurements are used by the radar tracker, which is based on a probabilistic data association filter (PDAF). See (Wilthil et al., 2017) for more details on the tracking system.

The BC-MPC algorithm interfaces the tracking system using a ROS service, which enables request-response functionality for providing obstacle estimates. The BC-MPC outputs a desired velocity trajectory to the model-based speed and course controller, which specifies a throttle and rudder command to the on-board control system through a TCP

Table 1: Telemetron ASV specifications.

Component	**Description**
Vessel hull	Polarcirkel Sport 845
Length	8.45 m
Width	2.71 m
Weight	1675 kg
Propulsion system	Yamaha 225 HP outboard engine
Motor control	Electro-mechanical actuation of throttle valve
Rudder control	Hydraulic actuation of outboard engine angle with proportional-derivative (PD) feedback control
Navigation system	Kongsberg Seatex Seapath 330+
Radar	Simrad Broadband 4G$^{\text{TM}}$ Radar
Processing platform	Intel® i7 3.4 GHz CPU, running Ubuntu 16.04 Linux

Table 2: BC-MPC algorithm parameters.

Parameter	**Value**	**Description**
$\boldsymbol{T}$	$\begin{bmatrix}5 & 20 & 30\end{bmatrix}$ s	Prediction horizon
$\boldsymbol{N}_U$	$\begin{bmatrix}5 & 1 & 1\end{bmatrix}$	Number of speed maneuvers
$\boldsymbol{N}_\chi$	$\begin{bmatrix}5 & 3 & 3\end{bmatrix}$	Number of course maneuvers
T_{ramp}	1 s	Ramp time
T_U	5 s	Speed maneuver length
T_χ	5 s	Course maneuver length
$T_{\tilde{U}}$	5 s	Speed error model time constant
$T_{\tilde{\chi}}$	5 s	Course error model time constant
Δ	500 m	LOS lookahead distance
γ_s	0.005 1/s	LOS along track distance gain
w_{al}	1	Align weight
w_{av}	6000	Avoid weight
w_{t}	4200	Transitional cost weight
w_χ	100	Angular error scaling weight
a_0	50 m	Collision region major axis
a_1	150 m	Safety region major axis
a_2	250 m	Margin region major axis
b_0	25 m	Collision region minor axis
b_1	75 m	Safety region minor axis
b_2	125 m	Margin region minor axis
d_{COLREGs}	100 m	COLREGs distance
γ_1	0.1	Obstacle cost gradient parameter

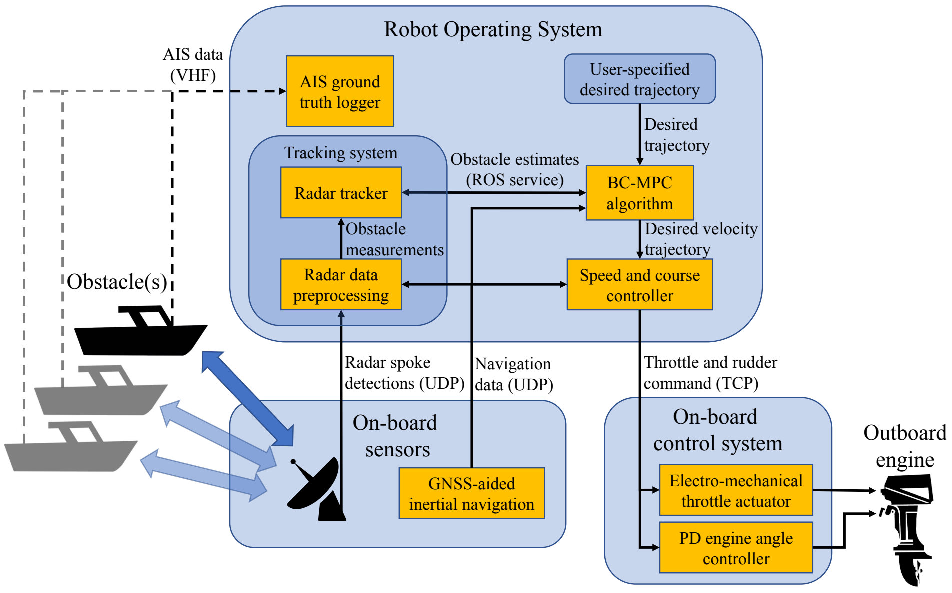

Robot Operating System
AIS data (VHF)
AIS ground truth logger
User-specified desired trajectory
Desired trajectory
Tracking system
Radar tracker
Obstacle estimates (ROS service)
BC-MPC algorithm
Obstacle measurements
Desired velocity trajectory
Radar data preprocessing
Speed and course controller
Obstacle(s)
Radar spoke detections (UDP)
Navigation data (UDP)
Throttle and rudder command (TCP)
On-board sensors
GNSS-aided inertial navigation
On-board control system
Electro-mechanical throttle actuator
PD engine angle controller
Outboard engine

Figure 15: Architecture of the COLAV implementation on the Telemetron ASV.

interface. The on-board control system has an electro-mechanical actuator for controlling the motor throttle, while the rudder command is handled by steering the outboard engine angle to the desired angle using a PD controller and a hydraulic actuation system.

The system receives AIS messages over VHF to obtain ground-truth trajectories for the vessels involved in the experiments. Notice that these are subject to the uncertainty of the navigation system providing the AIS data on the given vessels. They are, however, expected to be much more precise than the estimates from the radar-based tracking system. Figure 16a shows the inside of the Telemetron ASV, with the navigation system and processing platform.

The Kongsberg Seatex Ocean Space Drone 1 (OSD1) was used as the obstacle. This was originally an offshore lifeboat, which has been fitted with a full control and navigation system for testing autonomous control systems, shown in fig. 16b. The vessel is 12 m long, and has a mass of approximately 10 metric tons.

During the experiments, the OSD1 was steered on constant course with a speed of approximately $2.5\ \mathrm{m/s}$ (5 knots) using an autopilot. In addition to the OSD1, several commercial and leisure crafts were present in the area, affecting some of the scenarios.

We included four different scenarios in the experiments:

1. Head on. The ownship and OSD1 approaches each other on reciprocal courses. With respect to COLREGs, both vessels are required to perform starboard maneuvers.
2. Crossing from starboard. The OSD1 approaches from $90°$ on the ownship's starboard side. In this case, COLREGs requires the ownship to avoid collision, preferably by making a starboard maneuver and passing behind the OSD1.
3. Overtaking. The ownship approaches the OSD1 from behind with a higher speed. COLREGs requires the ownship to avoid collision by passing on either side. We prefer, however, to pass the OSD1 on its port side by doing a port maneuver.

(a) Erik Wilthil in the back of the Telemetron ASV with the navigation system and the processing platform in the rack to the right.

(b) The Kongsberg Seatex Ocean Space Drone 2, which is identical to the Ocean Space Drone 1. Courtesy of Kongsberg Seatex.

Figure 16: The inside of the Telemetron ASV (a) and the Kongsberg Seatex Ocean Space Drone 2 (b).

4. Crossing from port. Similar scenario as crossing from starboard, but here the OSD1 approaches the ownship from the port side. In this case, COLREGs deems the ownship as the stand-on vessel, and the OSD1 is supposed to avoid collision. The OSD1 will, however, keep its speed and course, resulting in rule 17 revoking the stand-on obligation and requiring the ownship to avoid collision, preferably avoiding maneuvering to port.

In the following sections, we present three head-on scenarios, two crossing from starboard scenarios, one overtaking scenario and one crossing from port scenario.

4.2 Head on: Experiments 1.1–1.3

The first experiments we performed were a number of head-on scenarios. In these scenarios, the desired trajectory inputted to the BC-MPC algorithm is a straight-line trajectory approaching the OSD1 on a reciprocal course, resulting in a collision with a relative bearing of $0°$ if the desired trajectory is followed. With respect to COLREGs, both vessels should perform starboard maneuvers. However, in our case, the OSD1 violates COLREGs by keeping its speed and course constant throughout the scenario.

To verify that the BC-MPC algorithm worked as it was supposed to, we first used AIS for providing obstacle estimates in Experiment 1.1. The OSD1 is equipped with an AIS transceiver providing low-noise estimates of the position, speed and course, originating from a Kongsberg Seatex SeaNav 300 navigation system. As shown in Fig. 17, we successfully avoid collision in this scenario. This is, however, achieved by performing a port maneuver, which violates the desired COLREGs behavior of maneuvering to starboard. This is most likely caused by the ownship approaching the obstacle on the port side of the desired trajectory, which together with the slightly angled obstacle trajectory makes a port maneuver attractive. The ownship is, however, either in a head-on or stand-on situation, but it is difficult to program an explicit understanding of this without introducing logic or discontinuous functions, which would reduce the robustness to noise. In addition, the algorithm is intended to handle short-term situations, in which the vague possibility of an obstacle making a port maneuver should not be neglected. The elliptical COLREGs obstacle function employs a soft COLREGs interpretation, which allows the algorithm to consider all actions in emergency situations, including maneuvering to port when the algorithm believes this is the safest. However, when making such non-conventional maneuvers, the algorithm requires a significantly increased obstacle clearance, which can be tuned. Notice also that in a hybrid architecture, the mid-level algorithm should have a harder interpretation of COLREGs which would maneuver to starboard at an earlier point, avoiding the situation in full as long as nothing unforeseen happen. Moreover, the maneuver is smooth with a sufficient course change to be readily observable for other vessels. Figure 18a shows the distance between the OSD1 and the ownship, and the predicted future distance given the trajectory the BC-MPC algorithm chose at each iteration, while Figure 18b shows the estimated and actual speed and course of the OSD1. The estimated values are in this case based on AIS, and hence equal to the ground truth. The OSD1 does, however,

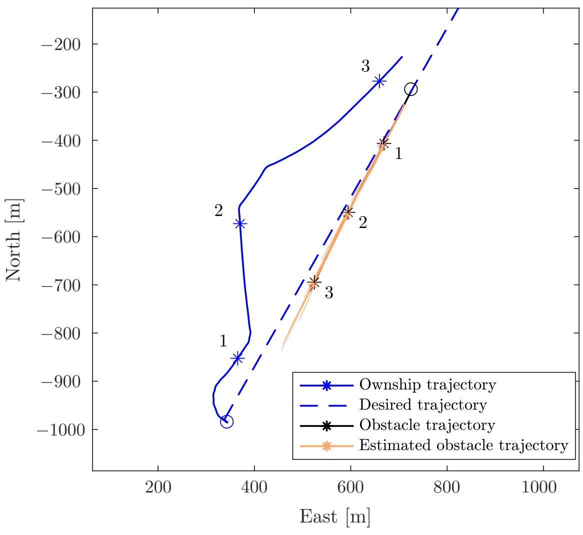
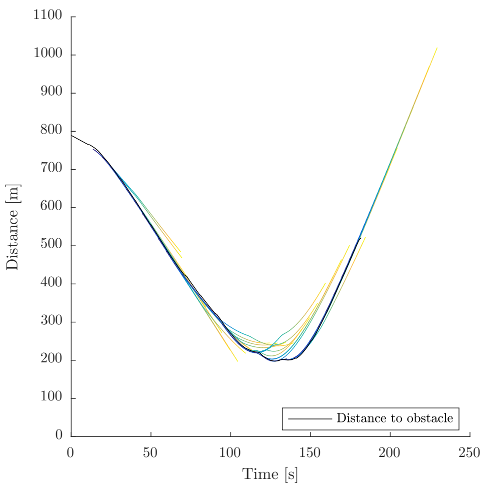
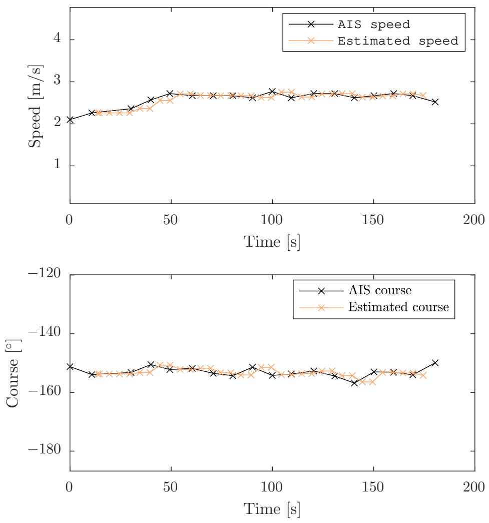


Figure 17: Experiment 1.1: Head-on scenario using AIS for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, the estimated obstacle trajectory is shown with the thick orange line, while predicted future trajectory for the obstacle at each timestep are shown as the thin orange lines. The numbers represent time markers for each 60 s. The obstacle trajectory in black is located behind the estimated obstacle trajectory in orange, since they both originate from the same AIS data in this experiment.



(a) The black line shows the actual distance between the vessels, while the colored lines show the predicted future distance at each BC-MPC iteration. Blue represents the start of the predictions while yellow represents the end.



(b) Speed and course estimates. The black crosses represent received AIS messages, while the orange crosses represent BC-MPC iterations.

Figure 18: Distance to the OSD1 (a) and the estimated speed and course (b) during Experiment 1.1.

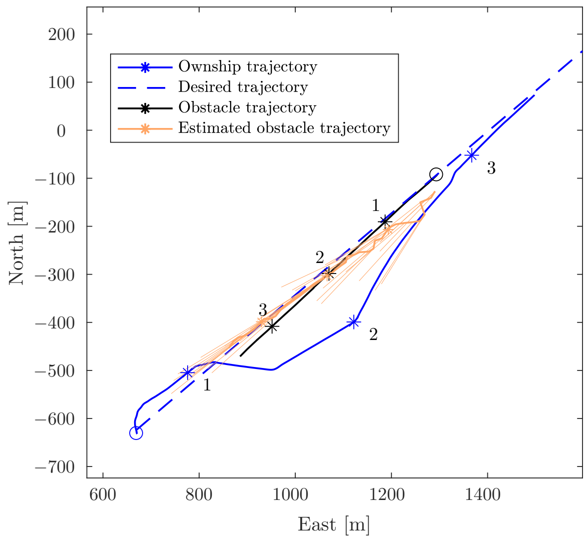
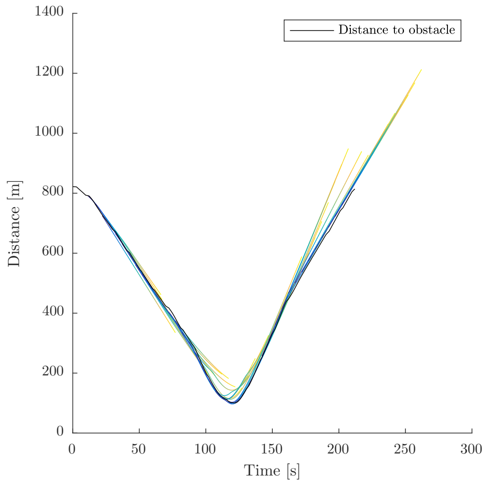


Figure 19: Experiment 1.2: Head-on scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, the estimated obstacle trajectory is shown with the thick orange line, while predicted future trajectory for the obstacle at each timestep are shown as the thin orange lines. The numbers represent time markers for each 60 s.

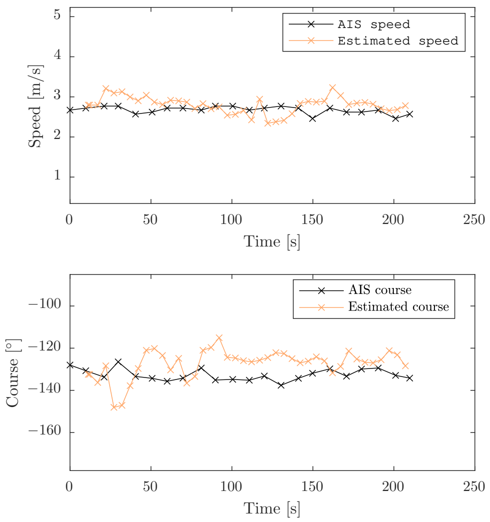
transmit AIS messages quite seldom, introducing some delay in the estimated speed and course.

Following this experiment, we performed several experiments using the radar-based tracking system for providing obstacle estimates. Fig. 19 shows the results from Experiment 1.2, a similar experiment as the one performed with AIS. In this experiment, the ownship performs a starboard maneuver in order to avoid collision, as preferred by COLREGs. As shown in the figure, there is a fair amount of noise on the obstacle estimates, in particularly the course estimate. This is confirmed by the course estimate shown in fig. 20b, which shows course fluctuations often in excess of 20°. Despite this, the ownship performs a smooth maneuver, which demonstrates the BC-MPC algorithm's robustness with respect to noise on the obstacle estimates. This is also shown in fig. 20a, where the predicted distance to the obstacle varies quite much without making the algorithm decide on a new maneuver.

The last head-on scenario, Experiment 1.3, is shown in Fig. 21, where we approach the OSD1 from north-east. The predicted future obstacle trajectories at each iteration are omitted from the following figures to improve the readability. This scenario was slightly more complex, as two other vessels unexpectedly entered the scenario. One of these was a high-speed leisure craft approaching from the west, while the other was a high-speed passenger ferry approaching from south-east, behind the ownship. The leisure craft did not have AIS, and we do therefore not have a ground-truth trajectory for this vessel. Fig. 22 shows an image captured by a drone during this experiment, with algorithm visualization embedded in the lower left corner. As in the previous scenario, we avoid the OSD1 by doing a starboard maneuver. Following this, we approach the desired trajectory before the passenger ferry approaches from abaft. With respect to COLREGs, this is an overtaking situation where we are deemed the stand-on vessel, and the passenger ferry "Trondheimfjord II" is supposed to give way to us. However, as mentioned earlier, the algorithm is designed to also handle the situations where the give-way vessel does not adhere to its obligations, requiring action by the stand-on vessel. Hence, the algorithm chooses to do a new starboard maneuver to let the passenger ferry pass. Eventually, the ferry turns towards the Trondheim Harbor allowing the ownship to approach the desired trajectory once again. There is some wobbling in the ownship trajectory which is most likely caused by obstacle estimate noise. This could possibly be avoided by changing the tuning parameters of the BC-MPC algorithm, namely increasing the transitional



(a) The black line shows the actual distance between the vessels, while the colored lines show the predicted future distance at each BC-MPC iteration. Blue represents the start of the predictions while yellow represents the end.



(b) Speed and course estimates. The black crosses represent received AIS messages, while the orange crosses represent BC-MPC iterations.

Figure 20: Distance to the OSD1 (a), and the estimated speed and course (b) during Experiment 1.2.

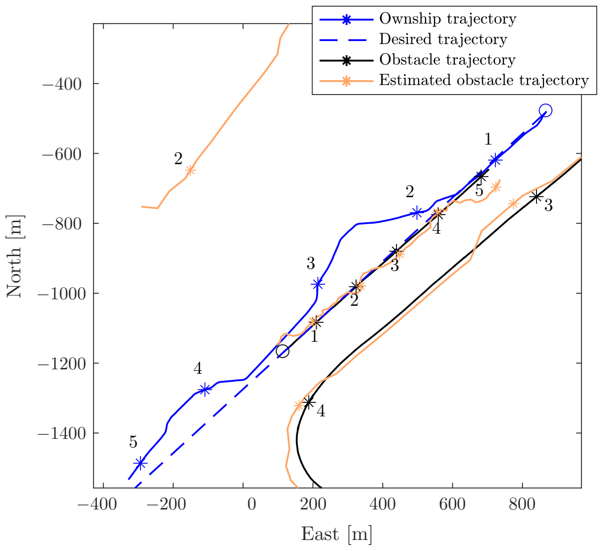


Figure 21: Experiment 1.3: Head-on scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 60 s. In this experiment, two vessels unexpectedly entered the scenario bringing the total vessels included up to three. The leisure craft in the upper-left corner was traveling towards North-East and did not have AIS, so there is no ground truth trajectory for this vessel.

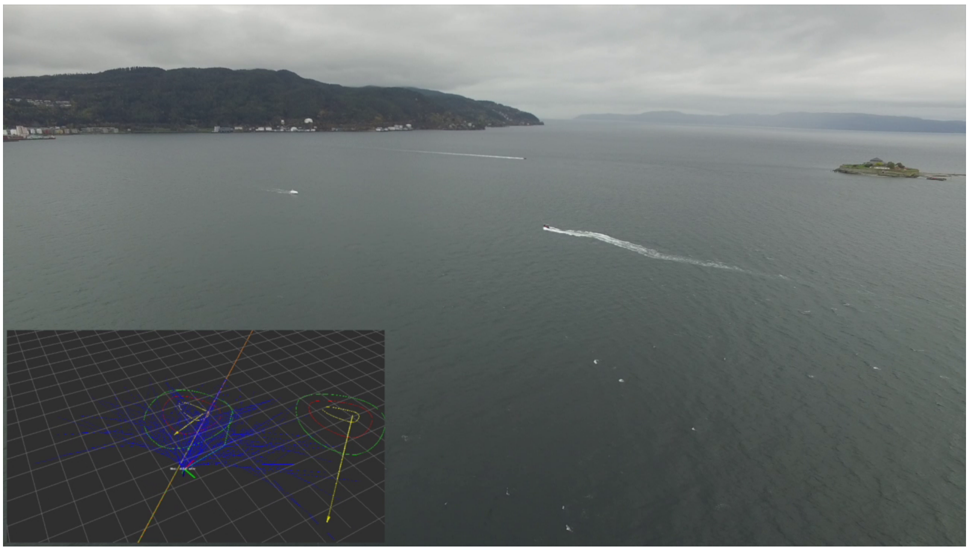


Figure 22: Drone picture during Experiment 1.3, approximately at the second time mark. The ownship is located in the middle of the picture, with the OSD1 to the left. The vessel in the background is a high-speed leisure craft. Yellow arrows in the visualization represent the estimated obstacle speed and course, while the orange line is the desired trajectory. The blue lines are the feedback-corrected BC-MPC pose trajectories, while the green line is the selected trajectory. Notice that the estimated course of the OSD1 deviates quite much from the actual vessel course, which was aligned by the orange line.

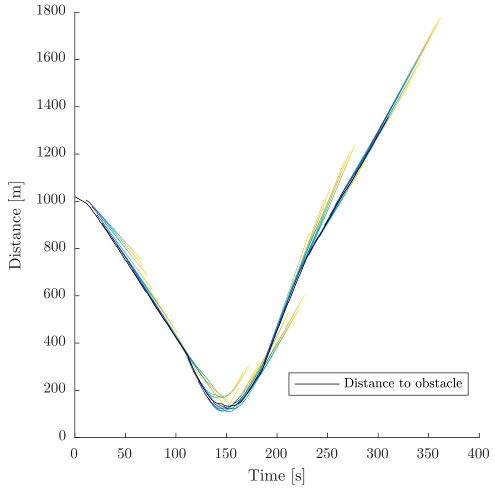
cost weight. In this experiment, the amount of estimate noise is even larger than in the previous experiment, with course fluctuations up to $40°$ as seen in fig. 23b. Still, as shown in fig. 23a and Fig. 21, the BC-MPC manages to make quite smooth maneuvers, which again shows robustness with respect to obstacle estimate noise. Fig. 24 shows similar plots for the Trondheimfjord II ferry. Notice that it takes some time before the tracking system detects that the passenger ferry makes a maneuver, which is due to a limited sample rate on the radar combined with some latency in the PDAF tracking system.

4.3 Crossing from starboard: Experiments 2.1–2.2

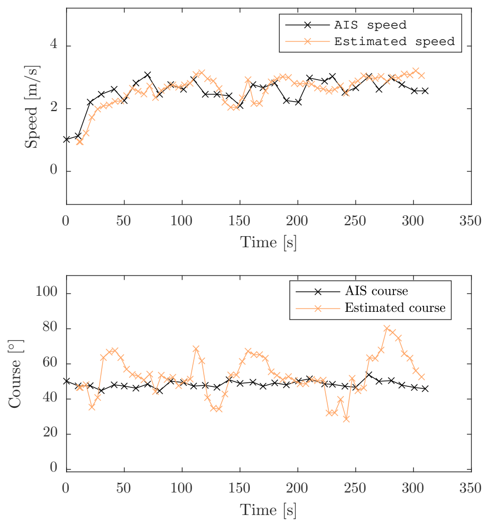
Crossing from starboard is a more complex scenario than the head-on scenario. We performed two experiments with the OSD1 approaching on collision course from starboard. The scenarios were constructed such that the desired trajectory coincides with the obstacle trajectory, resulting in a collision with a relative bearing of $-90°$ if the desired trajectory is followed. In such a scenario, the ownship is deemed the give-way vessel and should avoid collision by preferably maneuvering to starboard and passing abaft of the stand-on vessel.

In Experiment 2.1, shown in Fig. 25, we avoided collision with the OSD1 by maneuvering to port and passing in front of the obstacle. This can be considered as suboptimal with respect to the preferred action being passing abaft of the obstacle. Passing in front in a crossing situation is, however, not strictly forbidden by rule 15. Furthermore, the minimum distance to the obstacle is 214.0 m, meaning that the obstacle is only slightly inside the margin region. With this in mind, we consider this maneuver to be safe with similar arguments as for Experiment 1.1.

In Experiment 2.2, shown in Fig. 26, we avoided collision by passing abaft of the OSD1, as preferred by COLREGs. In this experiment, the minimum distance to the obstacle was 106.2 m, significantly closer than when we passed in front of the obstacle. This is still only slightly inside the margin region, remembering that the elliptical COLREGs penalty function with the tuning in Table 2 is smaller abaft an obstacle than in front of an obstacle.

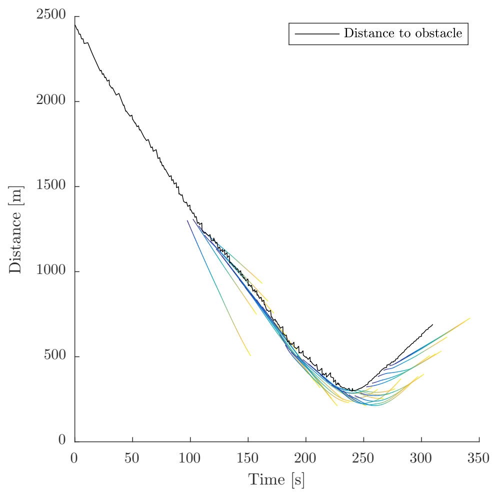


(a) The black line shows the actual distance between the vessels, while the colored lines show the predicted future distance at each BC-MPC iteration. Blue represents the start of the predictions while yellow represents the end.

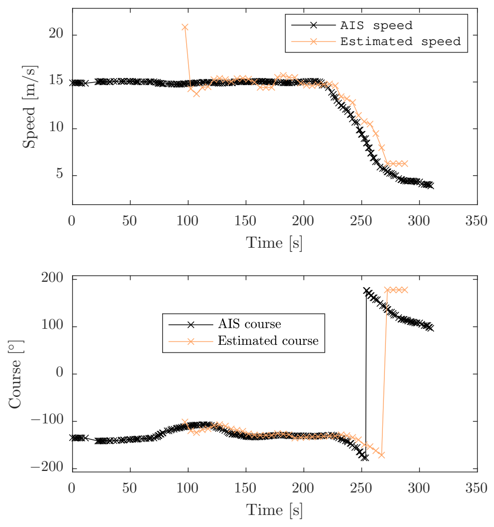


(b) Speed and course estimates. The black crosses represent received AIS messages, while the orange crosses represent BC-MPC iterations.

Figure 23: Distance to the OSD1 (a), and and the estimated speed and course (b) during Experiment 1.3.



(a) The black line shows the actual distance between the vessels, while the colored lines show the predicted future distance at each BC-MPC iteration. Blue represents the start of the predictions while yellow represents the end.



(b) Speed and course estimates. The black crosses represent received AIS messages, while the orange crosses represent BC-MPC iterations. The radar did not detect the Trondheimsfjord II after the course discontinuity at approximately $t = 270$ s, causing a large course error.

Figure 24: Distance to the Trondheimfjord II (a), and the estimated speed and course (b) during Experiment 1.3.

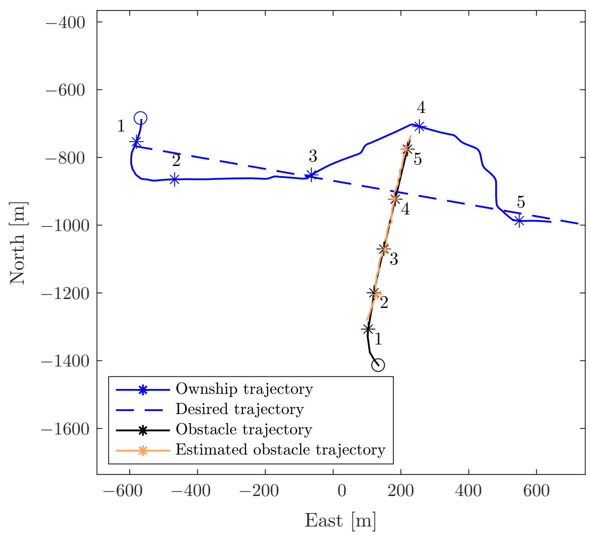


Figure 25: Experiment 2.1: Crossing from starboard scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 60 s.

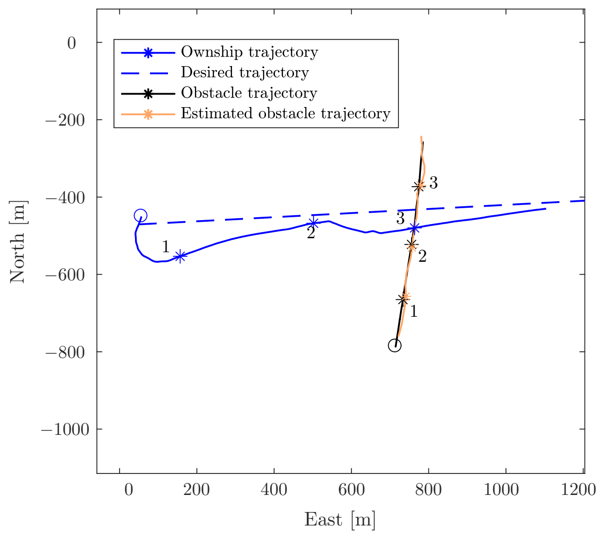


Figure 26: Experiment 2.2: Crossing from starboard scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 60 s.

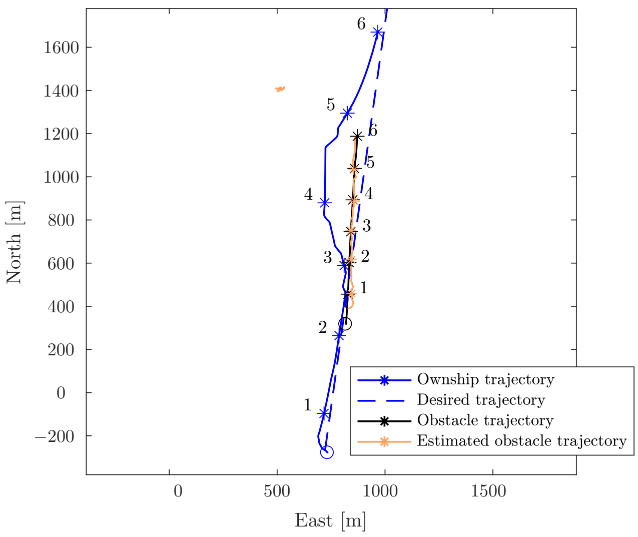


Figure 27: Experiment 3: Overtaking scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 60 s. The trajectory located at approximately $(1450, 500)$ originates from a navigational aid, which was detected approximately 130 s into the experiment.

4.4 Overtaking: Experiment 3

Another distinct situation is when the ownship approaches an obstacle from behind, overtaking it. With respect to COLREGs, the overtaking vessel has to keep out of the way of the overtaken vessel. There are no strict rules on whether the overtaking vessel should pass the overtaken vessel on the port or starboard side. However, we prefer to pass on the port side, as this does not block the overtaken vessel's possibilities in maneuvering to starboard if it finds itself in a head-on or crossing situation while being overtaken.

Fig. 27 shows Experiment 3, where the ownship overtakes the OSD1. The ownship maneuvers to port, passing the OSD1 on her port side. The ownship trajectory is quite smooth, but turns towards the desired trajectory a bit early. This was caused by the radar tracking system detecting a navigational aid in front of the ownship on the port side, which made maneuvering closer to the desired trajectory preferable. The ownship was approximately 200 m in front of the obstacle when doing this maneuver. Notice that we currently do not distinguish between dynamic and static objects in the tracking system, hence this navigational aid was considered as a moving vessel. The closest distance to the obstacle during the overtaking maneuver was 127.3 m, approximately equal to the size of the margin region on the port side of an obstacle.

4.5 Crossing from port: Experiment 4

The last scenario we tested was a crossing from port, which may be the most complex of the experiments presented in this article. This situation was generated similarly as the crossing from starboard situation, but with a relative bearing of $90°$ instead of $-90°$. Here, COLREGs deems the ownship as the stand-on vessel, while the OSD1 is deemed the give-way vessel. However, the OSD1 keeps its speed and course, requiring the ownship to avoid collision. In such a situation, COLREGs recommends the ownship to avoid maneuvering to port, favoring a starboard maneuver.

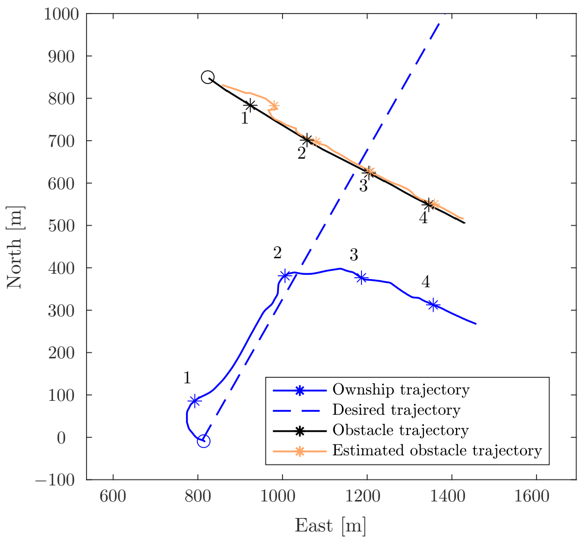


Figure 28: Experiment 4: Crossing from port scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 60 s.

Fig. 28 shows the results from Experiment 4, where the BC-MPC algorithm maneuvers the ownship to starboard, following the recommendations in COLREGs regarding this situation. The algorithm chose to maneuver the ownship at the minimum speed in order to minimize the distance to the desired trajectory. This minimum speed ensures maneuverability of the ownship, and was by coincidence similar as the speed of the OSD1 during the experiment, resulting in the ownship trajectory following parallel to the obstacle trajectory. Obviously, the ownship could increase the speed and pass in front of the obstacle, but this is not apparent to the BC-MPC algorithm due to the limited prediction horizon. In a hybrid COLAV architecture, this situation should be solved by the mid-level COLAV algorithm, designed with a longer prediction horizon than the BC-MPC algorithm.

4.6 Experiment summary

The BC-MPC algorithm has been tested in four different scenarios, each with different desirable behavior. A total of 7 experiments are presented, and the key points and numbers of the experiments are summarized in Table 3.

In the head-on experiments, both AIS and radar tracking was used for providing obstacle estimates. In Experiment 1.1, where we used AIS for providing obstacle estimates, the ownship avoided collision by passing the obstacle on its starboard side, violating the desired behavior of COLREGs. The ownship did, however, maneuver with increased clearance to the obstacle compared to experiments 1.2 and 1.3 where we passed the obstacle on its port side in accordance with the desired behavior of COLREGs. In experiments 1.2 and 1.3, we used the radar tracking system for obtaining obstacle estimates, which provided estimates with a large amount of noise compared to Experiment 1.1. The BC-MPC algorithm did, however, not seem to be significantly affected by this noise.

In the crossing from starboard experiments, we only used radar tracking for providing obstacle estimates. In Experiment 2.1, we passed in front of the obstacle. This is not strictly forbidden by COLREGs, but is neither desirable. The ownship did, however, have a large clearance to the obstacle, as required when performing such non-conventional maneuvers. In Experiment 2.2, we passed behind the obstacle, complying with the desirable behavior of COLREGs.

Table 3: Key points and numbers from the experiments. *In Experiment 2.1 we passed in front of the obstacle, while COLREGs prefers that the ownship pass behind the obstacle. Passing in front is, however, not strictly forbidden.

Experiment type and number	**Obstacle sensor**	**Rule 13–15 compliance**	**Minimum distance to obstacle**
Head on			
1.1	AIS	No	197.8 m
1.2	Radar	Yes	100.8 m
1.3	Radar	Yes	132.5 m
Crossing from starboard			
2.1	Radar	Yes*	214.0 m
2.2	Radar	Yes	106.2 m
Overtaking			
3.1	Radar	Yes	127.3 m
Crossing from port			
4.1	Radar	N/A	231.7 m

In Experiment 3.1 the ownship overtook the obstacle on its port side. COLREGs does not dictate which side the obstacle should be passed on, but by maneuvering to port the obstacle is free to maneuver to starboard if it finds itself in a separate collision situation.

Experiment 4.1 is a crossing situation where the ownship is deemed the stand-on vessel, and the OSD1 is required to avoid collision. The OSD1 did, however, not fulfill her obligation to avoid collision, requiring that the ownship avoided collision in accordance with rule 17 of COLREGs. The ownship avoided collision by performing a starboard maneuver, as suggested by COLREGs.

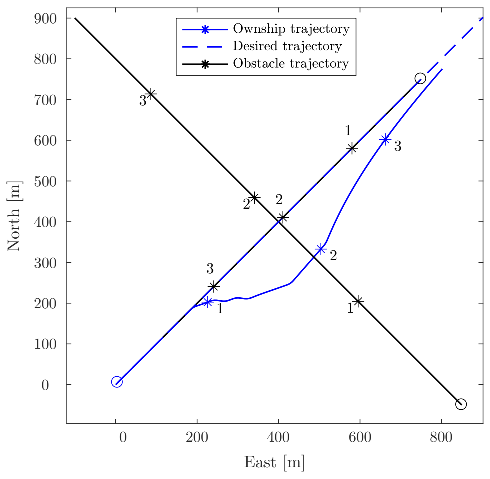
In two of the experiments, the BC-MPC algorithm chose to ignore the maneuvering aspects of COLREGs rules 14 and 15. This is because the algorithm in scope of a hybrid architecture is designed with a soft COLREGs interpretation, making the algorithm capable of handling emergency situations where the maneuvering aspect of rules 14 and 15 may need to be ignored. This can e.g. be situations where rule 17 revokes a stand-on requirement, in which the ownship is required to take such action that best aid avoiding collision, not necessarily strictly following rules 14 and 15. However, in cases where the algorithm choses non-conventional maneuvers ignoring the maneuvering aspects of rules 14 and/or 15, the obstacle is passed with extra clearance. The soft COLREGs interpretation also avoids the use of logic, which provides the algorithm with increased robustness towards obstacle estimate noise. This is an important property when using tracking systems based on exteroceptive sensors such as e.g. radar.

5 Simulation results

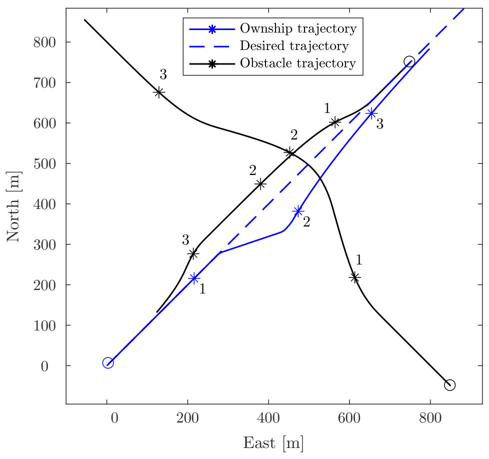
To complement the experimental results presented in the previous section, in this section we present simulation results in more complex situations. The simulations include multi-obstacle scenarios where multiple COLREGs rules apply simultaneously, also with obstacles that maneuver in accordance with the rules.

5.1 Simulation setup

The simulations are performed with the same tuning parameters as the experiments, shown in Table 2. To focus on the algorithm performance itself, we present the algorithm with noise-free measurements of the obstacle position, course and speed during the simulations. In order to challenge the algorithm, we presented it with four multi-obstacle scenarios:



(a) Obstacles not maneuvering.



(b) Obstacles maneuvering in accordance with COLREGs.

Figure 29: Simulation 1: Head on with simultaneous crossing from starboard with non-maneuvering (a) and maneuvering (b) obstacles. The ownship and obstacle initial positions are marked with circles, and the numbers represent time markers for each 60 s.

1. Head on and crossing from starboard.
2. Head on and crossing from port.
3. Head on and crossing from starboard with an extra obstacle.
4. Simultaneous crossing from starboard and port.

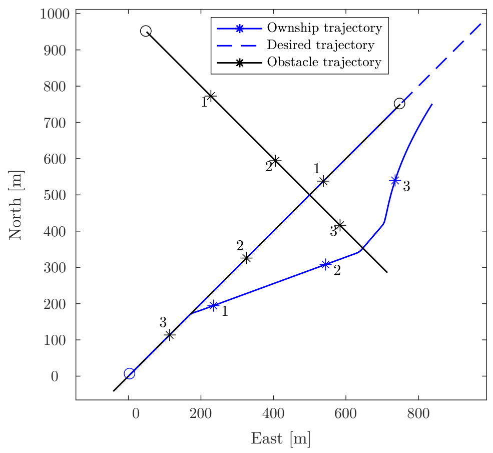
The scenarios are simulated both with obstacles not maneuvering, similar as in the experiments, and obstacles maneuvering in accordance with COLREGs.

5.2 Head on and crossing from starboard: Simulation 1

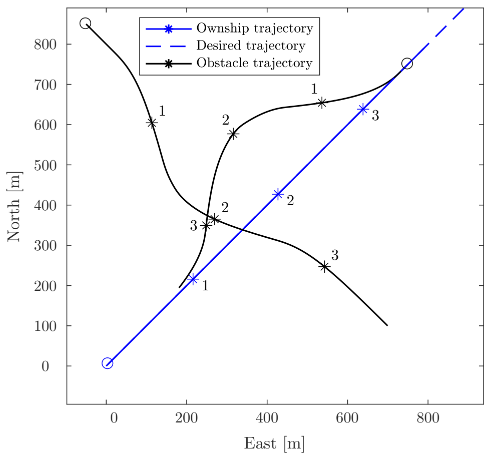
In this scenario, shown in Fig. 29, the ownship faces a simultaneous head-on and crossing from starboard situation, which both require the ownship to maneuver to starboard. With respect to COLREGs, the crossing obstacle has a stand-on obligation with respect to the ownship, and a give-way obligation with respect to the head-on obstacle. In this situation, the crossing obstacle should maneuver towards starboard, and pass behind the head-on obstacle, which should maneuver to starboard in accordance with the head-on situation with the ownship. In fig. 29a, the obstacles do not maneuver, and the BC-MPC algorithm choose a maneuver to starboard in order to avoid the head-on obstacle and pass behind the crossing obstacle. In fig. 29b, the obstacles maneuver in accordance with COLREGs, and the ownship makes a starboard maneuver and passes behind the crossing obstacle. The maneuver is, however, somewhat smaller than when the obstacles do not maneuver, which is caused by the head-on obstacle cooperating in achieving the required clearance.

5.3 Head on and crossing from port: Simulation 2

In this scenario, shown in Fig. 30, the ownship faces a simultaneous head-on and crossing from port situation. This situation is more complex than Simulation 1, since the crossing obstacle requires the ownship to stand on in accordance



(a) Obstacles not maneuvering.



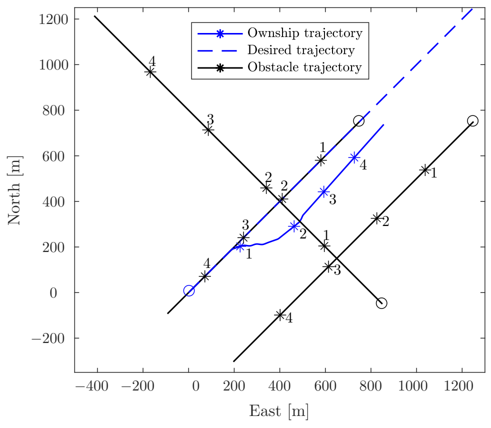
(b) Obstacles maneuvering in accordance with COLREGs.

Figure 30: Simulation 2: Head on with simultaneous crossing from port with non-maneuvering (a) and maneuvering (b) obstacles. The ownship and obstacle initial positions are marked with circles, and the numbers represent time markers for each 60 s.

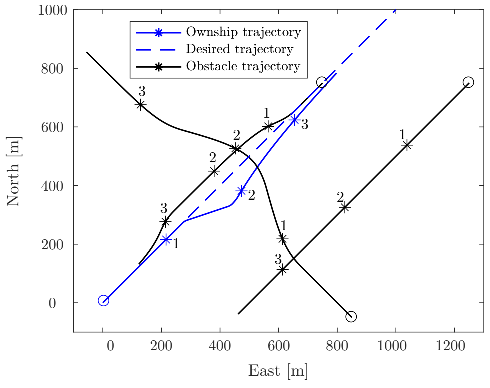
with rule 17, while the head-on obstacle requires the ownship to maneuver to starboard in accordance with rule 14. It is, however, dangerous to ignore a head-on obligation in order to stand on, and the algorithm should therefore prioritize the head-on situation. The head-on obstacle should give way to the crossing obstacle and maneuver to starboard in accordance with the head-on situation with the ownship, while the crossing obstacle should give way for the ownship. In fig. 30a, the obstacles do not maneuver, and the ownship applies a clear and large maneuver to starboard in order to avoid the head-on obstacle, and avoid collision with the crossing obstacle. When the obstacles maneuver, shown in fig. 30b, the BC-MPC algorithm evaluates the predicted clearance given how the obstacles maneuver, and chooses to stand on. It is clear that the head-on obstacle performs a large maneuver in order to pass behind the crossing obstacle, which combined with the crossing obstacle's maneuver makes it safe for the ownship to stand on. Notice, however, that time delays in estimating the obstacles position, speed and course would delay the ownship in detecting that the obstacles maneuver. This could, depending on the amount of time delay, make the BC-MPC algorithm initiate a maneuver to starboard, as when the obstacles did not maneuver.

5.4 Head on and crossing from starboard with an extra obstacle: Simulation 3

In this scenario, shown in Fig. 31, the ownship faces a head-on obstacle, and one crossing obstacle from starboard. In addition, there is another vessel approaching the ownship with an opposing course on a parallel path. The head-on obstacle has a stand-on obligation with respect to the crossing vessel, and a head-on obligation with respect to the ownship. The crossing obstacle has a give-way obligation with respect to the head-on obstacle, and a stand-on obligation with respect to the ownship. The ownship is in a head-on situation with the head-on obstacle, and has also to give way to the crossing obstacle. The third obstacle is considered to have sufficient clearance to the ownship and the two other obstacles to not be considered to be in a collision situation. In fig. 31a, the obstacles do not maneuver, and the ownship makes a starboard maneuver to avoid the head-on obstacle and pass behind the crossing obstacle. Following this, the ownship makes a port maneuver in order to avoid interfering with the third obstacle. This is an example of a situation where including future maneuvers in the search space is beneficial, since the algorithm has to plan for the port maneuver already when making the starboard maneuver in order see the full picture. Notice that the ownship has a slow convergence towards the desired trajectory in fig. 31a, which is due to the transitional cost term introducing a just too large cost for the algorithm to change to a trajectory with a faster convergence. When the obstacles maneuver, the BC-MPC algorithm chooses a similar, but smaller maneuver, as shown in fig. 31b.



(a) Obstacles not maneuvering.



(b) Obstacles maneuvering in accordance with COLREGs.

Figure 31: Simulation 3: Head on with simultaneous crossing from starboard with non-maneuvering (a) and maneuvering (b) obstacles. In addition, a third obstacle approaches with an opposing course on a parallel path. The ownship and obstacle initial positions are marked with circles, and the numbers represent time markers for each 60 s.

Table 4: Key points and numbers from the simulations. HO: Head on, CS: Crossing from starboard, CP: Crossing from port. *Denotes the extra obstacle approaching with an opposing course on a parallel path.

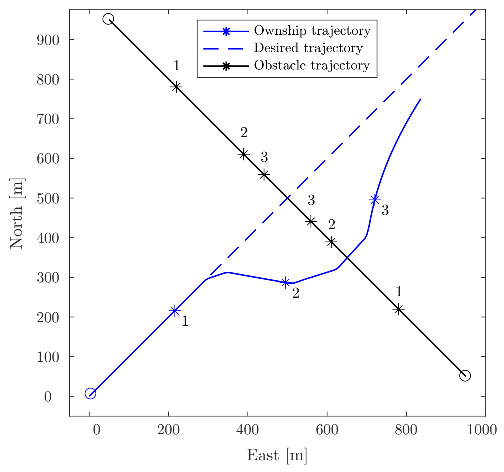
Simulation type and number	**Rule 13–15 compliance**	**Minimum distance to obstacle Non-maneuvering/maneuvering**		
		Head-on	**Starboard-crossing**	**Port-crossing**
1: HO + CS	Yes	121.3 m/115.0 m	116.6 m/146.4 m	N/A
2: HO + CP	Yes	130.6 m/181.7 m	N/A	169.6 m/165.4 m
3: HO + CS + extra	Yes	124.1 m/115.0 m	124.1 m/146.4 m	241.6 m/308.5 m*
4: CS + CP	Yes	N/A	124.2 m/112.2 m	167.6 m/192.3 m

5.5 Simultaneous crossing from starboard and port: Simulation 4

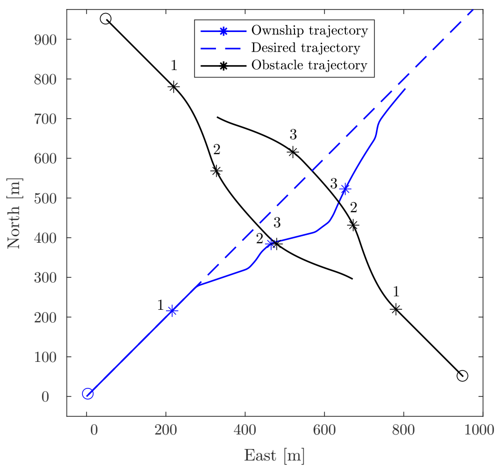
In this scenario, shown in Fig. 32, the ownship faces a simultaneous crossing from starboard and port. The obstacles are in head-on situations with each other. With respect to the ownship, the port obstacle has a give-way obligation, while the starboard obstacle has a stand-on obligation. The ownship is obliged to stand on with respect to the port obstacle, and give way to the starboard obstacle. In fig. 32a, the obstacles do not maneuver, and the ownship makes a large maneuver to starboard to pass behind the obstacle crossing from starboard and avoid to interfere with the obstacle crossing from port. This simulation is unrealistic since the obstacles collide with each other, but it does nevertheless provide insight into the performance of the BC-MPC algorithm. When the obstacles maneuver, as shown in fig. 32b, the ownship still maneuvers to starboard and passes behind the obstacle crossing from starboard. The maneuver is, however, performed with two subsequent turns, where the second starboard turn is made when the starboard obstacle turns to port to pass parallel to the other crossing obstacle.

5.6 Simulation summary

Key points and numbers from the simulations are presented in Table 4. To present some insight into how the BC-MPC algorithm performs in situations with multiple obstacles, and when multiple COLREGs rules apply at the same time,



(a) Obstacles not maneuvering.



(b) Obstacles maneuvering in accordance with COLREGs.

Figure 32: Simulation 4: Simultaneous crossing from starboard and port with non-maneuvering (a) and maneuvering (b) obstacles. The ownship and obstacle initial positions are marked with circles, and the numbers represent time markers for each 60 s.

the simulations have focused on more complex scenarios than the experiments. The results both include simulations where the obstacles continue on straight-line paths, like in the experiments, and simulations where the obstacles maneuver in accordance with COLREGs. In order to limit the scope, only four simulation scenarios are presented.

The BC-MPC algorithm managed to solve all the scenarios satisfactory, while complying with rules 13–15 of COLREGs. In the situations where the ownship was given both stand-on and give-way obligations, the give-way obligation was prioritized, except in simulation 2 when the obstacles maneuvered. The specific reason for this was that head-on obstacle made a large avoidance maneuver in order to fulfill its give-way obligation with respect to a crossing obstacle, which allowed the ownship to achieve a sufficient clearance while obeying the stand-on obligation.

When the obstacles maneuver in accordance with COLREGs, the BC-MPC algorithm generally chooses smaller maneuvers. As shown in Table 4, the minimum distance to head-on obstacles is approximately the same both when the obstacles maneuver and don't maneuver, except for Simulation 2 where the head-on obstacle makes a large maneuver. There is not a clear trend on how the minimum distance to crossing vessels is influenced when obstacles maneuver, but the number of simulations is anyhow too small to draw any statistical conclusions. Nevertheless, this indicates that the BC-MPC algorithm has an understanding of the joint responsibility in achieving the required clearance since it achieves approximately the same clearance regardless of whether the obstacles maneuver or not.

6 Conclusion and further work

We have presented a new algorithm named the branching-course MPC (BC-MPC) algorithm for ASV collision avoidance (COLAV). The algorithm has been validated in closed-loop full-scale experiments in the Trondheimsfjord in October 2017, using a radar-based system for obstacle detection and tracking. The algorithm performs well and displays good robustness with respect to noise on obstacle estimates, which is a significant source of disturbance when using tracking systems based on exteroceptive sensors to provide estimates of obstacle position, speed and course. In addition to the controlled obstacle, leisure and commercial vessels entered by coincidence some of the scenarios and were successfully avoided by the BC-MPC algorithm without human intervention. To complement the experimental results, we have performed simulations in complex scenarios involving multiple obstacles, where multiple COLREGs rules apply simultaneously. The simulations are performed both with non-maneuvering obstacles, and obstacles maneuvering in accordance with COLREGs. In the simulations, the BC-MPC algorithm successfully managed to avoid collision, while maneuvering in accordance with COLREGs.

The BC-MPC algorithm is intended for use as a short-term COLAV algorithm, and should therefore always be able to find a feasible solution to avoiding collision. This includes situations where rule 17 of COLREGs revokes a stand-on obligation, possibly requiring the algorithm to ignore the specific maneuvering parts of rules 14 and 15, dictating how to maneuver in head-on and crossing situations. However, the algorithm is motivated to follow the normal behavior described by rules 14 and 15 COLREGs when possible, and extra clearance is required if choosing non-conventional maneuvers ignoring the maneuvering aspects of rules 14 and 15. Hence, we consider the algorithm as being compliant with rules 8, 13 and 17 of COLREGs, and motivated to follow rules 14 and 15. This makes the algorithm well suited to handle the short-term aspects in a COLREGs-compliant hybrid COLAV architecture.

The authors have continued the work on the BC-MPC algorithm, specifically on including static obstacles and providing smoother trajectories with clearer maneuvers, which will be published in (Eriksen and Breivik, 2019). Future work includes performing an extensive simulation study, analyzing the algorithm's performance to a greater detail than what was possible to do in the scope of this article. This should e.g. include the effects of increasing noise levels on obstacle estimates, and how robust the algorithm is with respect to changes in the tuning parameters. Furthermore, we would also like to combine the algorithm with a long-term COLAV algorithm in a hybrid architecture.

Acknowledgments

This work was supported by the Research Council of Norway through project number 244116 and the Centres of Excellence funding scheme with project number 223254. The authors would like to express great gratitude to Kongsberg Maritime and Maritime Robotics for providing high-grade navigation technology, the Ocean Space Drone 1 and the Telemetron ASV at our disposal for the experiments.

References

Abdelaal, M. and Hahn, A. (2016). NMPC-based trajectory tracking and collision avoidance of unmanned surface vessels with rule-based COLREGs confinement. In *Proc. of the 2016 IEEE Conference on Systems, Process and Control (ICSPC)*, pages 23–28, Melaka, Malaysia.

Benjamin, M. R., Leonard, J. J., Curcio, J. A., and Newman, P. M. (2006). A method for protocol-based collision avoidance between autonomous marine surface craft. *Journal of Field Robotics*, 23(5):333–346.

Benjamin, M. R., Schmidt, H., Newman, P. M., and Leonard, J. J. (2010). Nested autonomy for unmanned marine vehicles with MOOS-IvP. *Journal of Field Robotics*, 27(6):834–875.

Breivik, M. and Fossen, T. (2004). Path following for marine surface vessels. In *Proc. of OCEANS*, pages 2282–2289, Kobe, Japan.

Caldwell, C. V., Dunlap, D. D., and Collins, E. G. (2010). Motion planning for an autonomous underwater vehicle via sampling based model predictive control. In *Proc. of IEEE OCEANS*, pages 1–6, Seattle, WA, USA.

Chauvin, C. (2011). Human factors and maritime safety. *Journal of Navigation*, 64(4):625–632.

Cockcroft, A. N. and Lameijer, J. N. F. (2004). *A Guide to the Collision Avoidance Rules.* Elsevier.

Dalsnes, B. R., Hexeberg, S., Flåten, A. L., Eriksen, B.-O. H., and Brekke, E. F. (2018). The neighbor course distribution method with Gaussian mixture models for AIS-based vessel trajectory prediction. In *Proc. of the 21st IEEE International Conference on Information Fusion (FUSION)*, pages 580–587, Cambridge, UK.

Elkins, L., Sellers, D., and Monach, W. R. (2010). The autonomous maritime navigation (AMN) project: Field tests, autonomous and cooperative behaviors, data fusion, sensors, and vehicles. *Journal of Field Robotics*, 27(6):790–818.

Eriksen, B.-O. H. and Breivik, M. (2017a). *Modeling, Identification and Control of High-Speed ASVs: Theory and Experiments*, pages 407–431. Springer International Publishing, Cham.

Eriksen, B.-O. H. and Breivik, M. (2017b). MPC-based mid-level collision avoidance for ASVs using nonlinear programming. In *Proc. of the 1st IEEE Conference on Control Technology and Applications (CCTA)*, pages 766–772, Mauna Lani, Hawai'i, USA.

Eriksen, B.-O. H. and Breivik, M. (2018). A model-based speed and course controller for high-speed ASVs. In *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*, pages 317–322, Opatija, Croatia.

Eriksen, B.-O. H. and Breivik, M. (2019). Short-term ASV collision avoidance with static and moving obstacles. Submitted to the 12th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles (CAMS).

Eriksen, B.-O. H., Breivik, M., Pettersen, K. Y., and Wiig, M. S. (2016). A modified dynamic window algorithm for horizontal collision avoidance for AUVs. In *Proc. of the 2016 IEEE Conference on Control Applications (CCA)*, pages 499–506, Buenos Aires, Argentina.

Eriksen, B.-O. H., Wilthil, E. F., Flåten, A. L., Brekke, E. F., and Breivik, M. (2018). Radar-based maritime collision avoidance using dynamic window. In *Proc. of the 2018 IEEE Aerospace Conference*, pages 1–9, Big Sky, Montana, USA.

Faltinsen, O. M. (2005). *Hydrodynamics of High-Speed Marine Vehicles*. Cambridge University Press.

Fiorini, P. and Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research*, 17(7):760–772.

Flåten, A. L. and Brekke, E. F. (2017). Rao-blackwellized particle filter for turn rate estimation. In *Proc. of IEEE Aerospace*, pages 1–7, Big Sky, Montana, USA.

Fossen, T. I. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons Ltd.

Fox, D., Burgard, W., and Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33.

Gray, A., Ali, M., Gao, Y., Hedrick, J. K., and Borrelli, F. (2013). A unified approach to threat assessment and control for automotive active safety. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1490–1499.

Hagen, I. B., Kufoalor, D. K. M., Brekke, E., and Johansen, T. A. (2018). MPC-based collision avoidance strategy for existing marine vessel guidance systems. In *Proc. of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7618–7623, Brisbane, Australia.

Harati-Mokhtari, A., Wall, A., Brooks, P., and Wang, J. (2007). Automatic identification system (AIS): Data reliability and human error implications. *Journal of Navigation*, 60(3):373–389.

Hart, P., Nilsson, N., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

IMO (2018). AIS transponders. `http://www.imo.org/en/OurWork/Safety/Navigation/Pages/AIS.aspx`. Accessed: 2018-08-31.

Johansen, T. A., Perez, T., and Cristofaro, A. (2016). Ship collision avoidance and COLREGS compliance using simulation-based control behavior selection with predictive hazard assessment. *IEEE Transactions on Intelligent Transportation Systems*, 17(12):3407–3422.

Keller, M., Haß, C., Seewald, A., and Bertram, T. (2015). A model predictive approach to emergency maneuvers in critical traffic situations. In *Proc. of IEEE ITSC*, pages 369–374, Las Palmas, Spain.

Kongsberg Maritime (2018). Autonomous ship project, key facts about YARA Birkeland. `https://www.km.kongsberg.com/ks/web/nokbg0240.nsf/AllWeb/4B8113B707A50A4FC125811D00407045`. Accessed: 2018-08-31.

Kuwata, Y. and How, J. P. (2011). Cooperative distributed robust trajectory optimization using receding horizon MILP. *IEEE Transactions on Control Systems Technology*, 19(2):423–431.

Kuwata, Y., Wolf, M. T., Zarzhitsky, D., and Huntsberger, T. L. (2014). Safe maritime autonomous navigation with COLREGS, using velocity obstacles. *IEEE Journal of Oceanic Engineering*, 39(1):110–119.

LaValle, S. M. (1998). Rapidly-exploring random trees: A new tool for path planning. Technical report, Computer Science Dept., Iowa State University.

Levander, O. (2017). Autonomous ships on the high seas. *IEEE Spectrum*, 54(2):26–31.

Ögren, P. and Leonard, N. (2005). A convergent dynamic window approach to obstacle avoidance. *IEEE Transactions on Robotics*, 21(2):188–195.

Paliotta, C. (2017). *Control of Under-actuated Marine Vehicles*. PhD thesis, Norwegian University of Science and Technology, Trondheim, Norway.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., and Ng, A. (2009). ROS: an open-source Robot Operating System. In *Proc. of ICRA Workshop on Open Source Software*, Kobe, Japan.

Schuster, M., Blaich, M., and Reuter, J. (2014). Collision avoidance for vessels using a low-cost radar sensor. In *Proc. of the 19th IFAC World Congress*, pages 9673–9678, Cape Town, South Africa.

Szlapczynski, R. and Szlapczynska, J. (2017). Review of ship safety domains: Models and applications. *Ocean Engineering*, 145:277–289.

Tan, C. S., Sutton, R., and Chudley, J. (2004). Collision avoidance systems for autonomous underwater vehicles part A: A review of obstacle detection. *Journal of Marine Science and Environment*, C2:39–50.

Švec, P., Shah, B. C., Bertaska, I. R., Alvarez, J., Sinisterra, A. J., von Ellenrieder, K., Dhanak, M., and Gupta, S. K. (2013). Dynamics-aware target following for an autonomous surface vehicle operating under COLREGs in civilian traffic. In *Proc. of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3871–3878, Tokyo, Japan.

Wilthil, E. F., Flåten, A. L., and Brekke, E. F. (2017). *A Target Tracking System for ASV Collision Avoidance Based on the PDAF*, pages 269–288. Springer International Publishing, Cham.

Paper G Energy-optimized hybrid collision avoidance for ASVs

Postprint of G. Bitar, **B.-O. H. Eriksen**, A. M. Lekkas, and M. Breivik, "Energy-optimized hybrid collision avoidance for ASVs", in *Proc. of the 17th IEEE European Control Conference (ECC)*, (Naples, Italy), 2019, pp. 2522–2529.

Energy-Optimized Hybrid Collision Avoidance for ASVs

Glenn Bitar, Bjørn-Olav H. Eriksen, Anastasios M. Lekkas and Morten Breivik

***Abstract*— This paper considers the development of a hybrid planning and collision avoidance architecture for autonomous surface vehicles (ASVs). The proposed architecture combines a high-level optimization-based planning algorithm with a mid-level collision avoidance (COLAV) algorithm based on model-predictive control (MPC). The high-level planner produces an energy-optimized trajectory by solving an optimal control problem via a pseudospectral method, taking into account known static obstacles and ocean currents. The mid-level algorithm performs MPC by solving a nonlinear program (NLP) to produce a collision-free local trajectory, also taking into account dynamic obstacles. In particular, the NLP optimizes for a combination of following the energy-optimized trajectory with performing readily observable maneuvers, as defined by Rule 8 of the International Regulations for Preventing Collisions at Sea (COLREGs). Numerical simulations are used to verify that the hybrid architecture produces safe, efficient and readily observable trajectories.**

I. Introduction

Autonomous ships are currently being explored for transportation and marine operations. The Yara Birkeland project in Norway [1] is an example of this, where the aim is to replace 40 thousand truck journeys of fertilizer per year with an autonomous cargo ship. Specifically, autonomous surface vehicles (ASVs) can provide reduced costs, risks and environmental impact, increased operational windows and introduce new opportunities in ocean operations. In excess of $75\,\%$ of maritime accidents are caused by human errors, which emphasizes the potential for increased safety by autonomous technology [2]. Approximately $9\,\%$ of Norwegian domestic CO_2 emissions originate from ships, which signifies the potential impact of energy-optimization in ship technology [3].

A fundamental requirement for ASVs is a robust and safe collision avoidance (COLAV) system. The ASV must be able to avoid static obstacles, such as land and shallow waters, as well avoid dynamic obstacles, including other ships. In addition, the ASV must perform its maneuvers in compliance with the International Regulations for Preventing Collisions at Sea (COLREGs) [4], which are often referred to as the "rules of the road" for marine vessels.

There exist numerous COLAV algorithms for marine applications. They may be divided into terms of the temporal planning horizon, e.g. long-term and short-term algorithms [5]. Long-term COLAV algorithms include offline planners as well as algorithms intended to run online, with a large

The authors are with the Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway. Email: {glenn.bitar, anastasios.lekkas}@ntnu.no, {bjorn-olav.h.eriksen, morten.breivik}@ieee.org

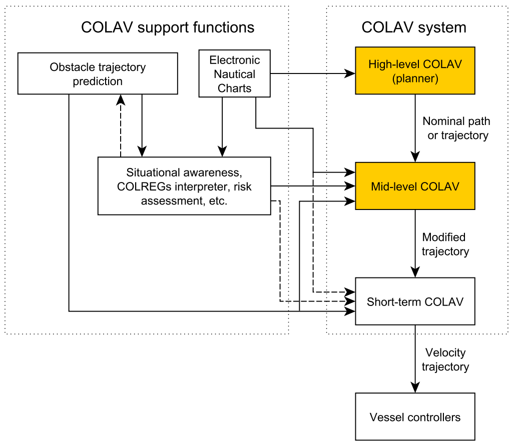

COLAV support functions
COLAV system
Obstacle trajectory prediction
Electronic Nautical Charts
High-level COLAV (planner)
Nominal path or trajectory
Situational awareness, COLREGs interpreter, risk assessment, etc.
Mid-level COLAV
Modified trajectory
Short-term COLAV
Velocity trajectory
Vessel controllers

Fig. 1. Suggested hybrid COLAV architecture with three layers. A high-level planner is combined with a mid-level COLAV algorithm to form parts of a hybrid architecture for an ASV. A complete hybrid architecture would also include short-term COLAV for immediate obstacle avoidance. The hybrid COLAV architecture should be supported by data from electronic nautical charts, represented in a suitable manner for the algorithms, as well as situational awareness functions that track and predict obstacles, perform risk assessment, etc.

spatial and temporal horizon. Bitar *et al.* present a long-term path and trajectory planning algorithm based on pseudospectral optimal control [6], inspired by findings in e.g. [7]. The algorithm produces energy-optimized trajectories and takes into account environmental factors, such as ocean current and static obstacles. Eriksen and Breivik have developed a long-term COLAV algorithm which takes into account Rule 8 of COLREGs, which requires that maneuvers are readily observable for other vessels [8]. The algorithm is based on model predictive control (MPC) and avoids static and dynamic obstacles by planning local trajectories via a nonlinear program (NLP). The NLP objective function includes terms to penalize slow changes in course and speed, hence making the resulting maneuvers observable. Other MPC-based COLAV algorithms include [9]–[11]. Other long-term algorithms include a *Voronoi diagram*-based method [12], *cell decomposition* [13], *rapidly exploring random trees (RRTs)* [14], with a COLREGs-compliant implementation in [15].

Short-term COLAV algorithms take into account a limited spatial and temporal horizon, and plan maneuvers to avoid immediate collision. Eriksen *et al.* have developed

Fig. 2. The *Telemetron* ASV. Courtesy of Maritime Robotics.

the *Branching-Course MPC Algorithm*, which produces collision-free, dynamically feasible local trajectories, aligned with an input desired trajectory, and is partially COLREGs compliant [5]. Other examples of short-term COLAV algorithms include *dynamic window* [16], and *velocity obstacles* [17].

Loe [18] and Casalino *et al.* [19] propose a hybrid COLAV architecture where the COLAV task is split into two or three layers to exploit the complementary strengths of long-term and short-term algorithms. We base our hybrid architecture on [8], with the structure illustrated in Fig. 1, where the COLAV system is divided into three layers. The top-level layer is the path or trajectory planner, which is intended to run offline prior to a mission, with global information about static obstacles. To be able to optimize with respect to energy, it is necessary to use a trajectory rather than a path, since energy consumption is tightly coupled with speed and acceleration. The mid-level layer is responsible for adhering to the parts of COLREGs that are related to maneuvering, and takes into account information about local dynamic and static obstacles. The short-term level is intended to execute dynamically feasible maneuvers and avoid immediate collision situations. This makes the hybrid COLAV architecture comply with Rule 17 of COLREGs, stating that the stand-on vessel (which has right of way) should take action if the give-way vessel does not, by having the ability to ignore the COLREGs considerations from the mid-level.

The authors have worked extensively on several parts of the hybrid COLAV architecture in Fig. 1. Examples include long-term COLAV algorithms intended to run offline as high-level planners [6], [12], or as mid-level algorithms [8], and short-term algorithms [5], [16]. The short-term algorithms have been tested in several full-scale experiments with radar-based obstacle detection and tracking. The COLAV algorithms are dependent on the performance of the vessel controllers. Therefore, the authors have also put significant effort into modeling, identification and control of ASVs [20], [21]. In this paper, we develop parts of the hybrid architecture shown in Fig. 1, by connecting and further developing two existing algorithms [6], [8]. Several improvements have been made to both algorithms:

- The high-level planner [6] has been extended to work with a non-first-principles model.
- The mid-level algorithm [8] is extended with relative velocities as a way to include ocean currents,
- has improved numerical properties, and
- has been extended with an interface for *relative* trajectory tracking, enabling the ASV to track a desired trajectory with a time-varying time offset.

Both algorithms utilize a mathematical model of the ASV *Telemetron*, depicted in Fig. 2.

The rest of the paper is organized as follows: The updated mathematical model of the ASV is introduced in Section II. The high-level planner and the mid-level algorithm are presented in Section III and Section IV, respectively. A description of the scenario used to test the approach is presented in Section V, along with a discussion of our results. Section VI concludes the paper.

II. ASV MODELING

In [20], Eriksen and Breivik developed a nonlinear dynamic model structure for high-speed ASVs operating in the displacement, semi-displacement and planing regions. They identified parameters for the *Telemetron* ASV, which is the vessel considered in this paper. The model has the form

$$\dot{\boldsymbol{\eta}} = \mathbf{R}(\chi)\boldsymbol{x} \tag{1a}$$

$$\dot{\chi} = r + \dot{\beta} \tag{1b}$$

$$\mathbf{M}(\boldsymbol{x})\dot{\boldsymbol{x}} + \boldsymbol{\sigma}(\boldsymbol{x}) = \boldsymbol{\tau}\,. \tag{1c}$$

Here, the pose vector $\boldsymbol{\eta} = [x, y, \psi]^\top \in \mathbb{R}^2 \times S$ contains the ASV's position and heading angle in the Earth-fixed North-East-Down (NED) frame $\{n\}$. The vector $\boldsymbol{x} = [U, r]^\top \in \mathbb{X} \subset \mathbb{R}^2$ contains the ASV's speed over ground (SOG) U and the rate of turn (ROT) r. The matrix $\mathbf{R}(\chi)$ transforms the SOG and ROT to kinematic velocities:

$$\mathbf{R}(\chi) = \begin{bmatrix} \cos\chi & 0 \\ \sin\chi & 0 \\ 0 & 1 \end{bmatrix}. \tag{2}$$

The course rate $\dot{\chi}$ is determined by the ROT r and the sideslip rate $\dot{\beta}$. The matrix $\mathbf{M}(\boldsymbol{x}) \in \mathbb{R}^{2\times 2}$ represents velocity-dependent inertia, and $\boldsymbol{\sigma}(\boldsymbol{x}) \in \mathbb{R}^2$ is the damping effect. The ASV is controlled by the control vector $\boldsymbol{\tau} = [\tau_m, \tau_\delta]^\top \in \mathbb{U} \subset \mathbb{R}^2$, which contains normalized values for throttle and rudder, respectively.

The states and controls are restricted to the sets $\mathbb{X}$ and $\mathbb{U}$ which are where the model (1) is valid. Detailed information about the valid sets are found in [20].

A. Extension to relative velocities

The model for *Telemetron* (1) does not take into account ocean currents, nor does it separate surge and sway motion. In this paper, we change (1c) to include relative velocities in order to account for ocean currents.

The following assumptions are made to include ocean current disturbances.

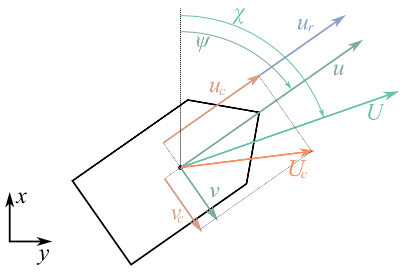


Fig. 3. An illustration of Assumption 2 with ocean currents. The ship's absolute surge velocity u is larger than the ocean current velocity in the surge direction u_c, giving a positive relative surge velocity $u_r = u - u_c$. However, in the sway direction, there is no model of the ship's movement. Thus we assume that the ship's relative sway velocity v_r is zero, implying that the sway velocity v is equal to the ocean current velocity in the sway direction v_c. The figure also illustrates the heading and course angles, ψ and χ.

Assumption 1. *The model for SOG U from* (1c) *is valid for relative surge velocity u_r.*

Assumption 2. *The relative sway velocity v_r is identically equal to zero.*

The reasoning for Assumption 1 is that the majority of the damping effects during model identification experiments were due to relative velocity. Assumption 2 may seem counterintuitive, but since we don't know anything about the sway dynamics of the vessel, we assume that the ship's sway motion follows the ocean currents, implying that the sway velocity v is equal to the sway-component of the ocean current v_c. Fig. 3 illustrates this idea.

Under these assumptions we change the model (1c) from a SOG and ROT model to a relative-velocity model without loss of generality for the rest of this paper. In fact, any ordinary ship model may be used for the hybrid architecture. The new model has the form

$$\dot{\boldsymbol{\eta}} = \mathbf{R}(\psi)\boldsymbol{x}_r + \begin{bmatrix}\boldsymbol{V}_c^\top & 0\end{bmatrix}^\top \tag{3a}$$

$$\mathbf{M}(\boldsymbol{x}_r)\dot{\boldsymbol{x}}_r + \boldsymbol{\sigma}(\boldsymbol{x}_r) = \boldsymbol{\tau}\,, \tag{3b}$$

where $\boldsymbol{x}_r = [u_r, r]^\top \in \mathbb{X}_r \subset \mathbb{R}^2$ and $\boldsymbol{V}_c = [V_x, V_y]^\top \in \mathbb{R}^2$ describes the ocean current velocity in NED. The set $\mathbb{X}_r$ is equivalent to $\mathbb{X}$, based on the assumptions made for the ocean current extension. The rest of the symbols are the same as in (1). The relationship between the absolute surge u and sway v velocities and the relative surge u_r and sway v_r velocities is described by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} u_r + u_c \\ v_r + v_c \end{bmatrix} = \begin{bmatrix} u_r + u_c \\ v_c \end{bmatrix}, \tag{4}$$

where u_c and v_c are the surge and sway components of the ocean current velocity.

The ocean current velocity is assumed to be constant and known in the NED frame. The body-fixed frame $\{b\}$ has origin and orientation fixed in the ASV, which results in the following ocean-current representation:

$$\begin{bmatrix} u_c \\ v_c \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi \\ -\sin\psi & \cos\psi \end{bmatrix} \boldsymbol{V}_c\,. \tag{5}$$

III. High-level planner

The high-level planner solves an optimal control problem (OCP) by using a pseudospectral method to find an energy-optimized trajectory, as described in [6]. The OCP is given by

$$\min_{\boldsymbol{z}_{r,d},\boldsymbol{\tau}_d,t_f} \int_0^{t_f} F(\boldsymbol{z}_{r,d}(t), \boldsymbol{\tau}_d(t), t)\,\mathrm{d}t \tag{6a}$$

subject to

$$\dot{\boldsymbol{z}}_{r,d}(t) = \boldsymbol{f}(\boldsymbol{z}_{r,d}(t), \boldsymbol{\tau}_d(t))\ \forall t \in [0, t_f] \tag{6b}$$

$$\boldsymbol{h}_p(\boldsymbol{z}_{r,d}(t), \boldsymbol{\tau}_d(t), t) \leq \mathbf{0}\ \forall t \in [0, t_f] \tag{6c}$$

$$\boldsymbol{e}_p(\boldsymbol{z}_{r,d}(0), \boldsymbol{z}_{r,d}(t_f), t_f) = \mathbf{0}\,. \tag{6d}$$

The symbols used in (6) will be further explained in Section III-A. We subscript the states and inputs with $(\cdot)_d$ to emphasize that these are desired values.

A. Cost functional, dynamics, inequality constraints and boundary conditions

To obtain an energy-optimized trajectory, we utilize an energy-based cost functional in (6a), with

$$F(\boldsymbol{z}_{r,d}(t), \boldsymbol{\tau}_d(t), t) = n(\tau_{m,d}(t))^2 \cdot \Big(\big|\cos\delta(\tau_\delta(t))\, u_{r,d}(t)\big| + \big|L_e \sin\delta(\tau_\delta(t))\, r\big|\Big)\,, \tag{7}$$

where $n(\tau_m)$ is a function that maps the control input τ_m to propeller rounds per minute (RPM), $\delta(\tau_\delta)$ maps the control input τ_δ to engine angle, and L_e is the length between the engine and the axis of rotation. Force is often modeled as being proportional to the square of propeller rate, and the product of force and velocity gives a measure proportional to power [22].

The dynamics (6b) are the same as (3), collected in the model

$$\dot{\boldsymbol{z}}_r = \boldsymbol{f}(\boldsymbol{z}_r, \boldsymbol{\tau})\,, \tag{8}$$

where $\boldsymbol{z}_r = [\boldsymbol{\eta}^\top, \boldsymbol{x}_r^\top]^\top$, and

$$\boldsymbol{f}(\boldsymbol{z}_r, \boldsymbol{\tau}) = \begin{bmatrix} \mathbf{R}(\psi)\boldsymbol{x}_r + [\boldsymbol{V}_c^\top, 0]^\top \\ \mathbf{M}(\boldsymbol{x}_r)^{-1}(-\boldsymbol{\sigma}(\boldsymbol{x}_r) + \boldsymbol{\tau}) \end{bmatrix}. \tag{9}$$

Static obstacles are represented in (6c) as elliptic inequalities. The basis for the elliptic inequality is

$$\left(\frac{x - x_c}{x_a}\right)^2 + \left(\frac{y - y_c}{y_a}\right)^2 \geq 1\,, \tag{10}$$

where x_c and y_c describe the ellipse center in NED, and x_a and y_a describe the sizes of the two elliptic axes. To allow angled ellipses, the differences $x - x_c$ and $y - y_c$ are rotated by the angle α between the direction of x_a and north. Applying the logarithmic function to both sides of the inequality avoids quadratic growth in x and y, which is convenient for the solvers, and adding a small value $\epsilon > 0$ improves the numerical properties when $x \to x_c$ and $y \to y_c$, without changing the inequality. The resulting inequality in (6c) is:

$$h_o(x, y, x_c, y_c, x_a, y_a, \alpha) = \\ -\log\left[\left(\frac{(x-x_c)\cos\alpha + (y-y_c)\sin\alpha}{x_a}\right)^2 \right. \\ \left. + \left(\frac{-(x-x_c)\sin\alpha + (y-y_c)\cos\alpha}{y_a}\right)^2 + \epsilon\right] \\ + \log(1+\epsilon) \leq 0\,. \quad (11)$$

In addition to the static obstacles, the state constraints $\boldsymbol{x}_{r,d} \in \mathbb{X}_r$ and control constraints $\boldsymbol{\tau}_d \in \mathbb{U}$ are also enforced in (6c).

Boundary conditions (6d) are employed to decide initial position and velocity, final position, and to set a maximum end time $t_f \leq t_{f,\max}$. The maximum end-time limitation is necessary for convergence of the OCP, since without it, the cost functional is minimized by not using any control action, resulting in the ship following the ocean currents.

B. Planner output

The OCP solution yields a trajectory $\boldsymbol{z}_{r,d}(\cdot)$ which describes the motion of the ASV over time t. The notation $\boldsymbol{z}_{r,d}(\cdot)$ emphasizes that this is not a point $\boldsymbol{z}_{r,d}(t)$ at a certain time t, but describes a trajectory of states. The planner is run offline ahead of time, and the x, y coordinates of the trajectory are used as input to the mid-level MPC algorithm, i.e. $\boldsymbol{p}_d(t) = [x_d(t), y_d(t)]^\top$, as illustrated in Fig. 4. Fig. 1 shows this interface in the context of the hybrid architecture.

IV. Mid-level COLAV

The mid-level COLAV algorithm is an MPC-based algorithm intended for long-term COLAV with respect to both static and dynamic obstacles, originally presented in [8]. The algorithm complies with Rule 8 of COLREGs, producing maneuvers that are readily observable for other vessels while following a reference trajectory specified by the trajectory planner.

In this section, we further develop the algorithm to be able to perform *relative* trajectory tracking and improve on the numeric properties of the algorithm. By relative trajectory tracking, we mean that we want the algorithm to track a desired trajectory with a time offset $t_b \in \mathbb{R}$. This implies that if the ASV for some reason lag behind the desired trajectory $\boldsymbol{p}_d(t) = [x(t), y(t)]^\top$, we may adjust t_b to offset the desired trajectory rather than speeding up to catch up with $\boldsymbol{p}_d(t)$. We define the relative desired trajectory as

$$\bar{\boldsymbol{p}}_d(t) = \boldsymbol{p}_d(t + t_b), \quad (12)$$

where t_b is the time offset, which is computed each time the mid-level algorithm is run. For notational simplicity, we omit t_b from the function parameters since (12) at each MPC run should be interpreted as a time-shifted trajectory only dependent on the time t. At a time step t_0, denoting the current time when the MPC is called, the time offset t_b is computed by the optimization problem

$$t_b(t_0) = \arg\min_{t_b} \left\| \boldsymbol{p}_d(t_0 + t_b) - \boldsymbol{p}(t_0) \right\|_2 , \quad (13)$$

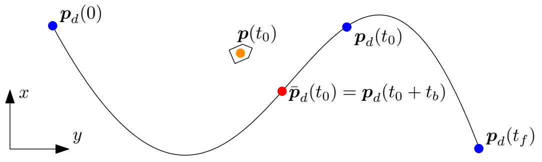


Fig. 4. Illustration of relative trajectory tracking. The black line shows the desired trajectory, with the blue circles marking the start position $\boldsymbol{p}_d(0)$, current position $\boldsymbol{p}_d(t_0)$ and goal desired position $\boldsymbol{p}_d(t_f)$. The orange circle marks the vessel position at time t, while the red point shows the relative desired position $\bar{\boldsymbol{p}}_d(t) = \boldsymbol{p}_d(t_0 + t_b)$.

which can be solved by a simple line search algorithm. Here, we find the offset t_b which minimizes the Euclidean distance between the current relative desired trajectory position $\bar{\boldsymbol{p}}_d(t_0)$ and the current vessel position $\boldsymbol{p}(t_0)$. The concept is illustrated in Fig. 4. In general, other paradigms can be used to find t_b based on the desired behavior. Using the Euclidean distance makes the algorithm track the desired trajectory from the closest point.

A similar concept can be employed if the input to the mid-level algorithm is a geometric path $\boldsymbol{p}_d(\theta)$ with an associated along-path speed $U_d(\theta)$, where $\theta \in \mathbb{R}$ is a path parameter. In this case, we compute an initial path parameter

$$\theta_0(t_0) = \arg\min_{\theta} \left\| \boldsymbol{p}_d(\theta) - \boldsymbol{p}(t_0) \right\|_2 , \quad (14)$$

and generate a map from time to path parameter $\theta : \mathbb{R} \rightarrow \mathbb{R}$ by integrating

$$\dot{\theta} = \frac{U_d(\theta)}{\sqrt{\left(\frac{\partial x_d(\theta)}{\partial \theta}\right)^2 + \left(\frac{\partial y_d(\theta)}{\partial \theta}\right)^2}} \quad \theta(t_0) = \theta_0(t_0)\,. \quad (15)$$

This gives $\theta(t)$ valid for $t \geq t_0$. This map is then used to compute the relative desired trajectory as $\bar{\boldsymbol{p}}_d(t) = \boldsymbol{p}_d(\theta(t))$. This would for instance be useful if we would like the mid-level algorithm to input a waypoint-defined path.

The mid-level algorithm is formalized as the OCP:

$$\min_{\boldsymbol{\eta}, \boldsymbol{x}_r} \ \phi(\boldsymbol{\eta}(\cdot), \boldsymbol{x}_r(\cdot)) \quad (16a)$$

subject to

$$\dot{\boldsymbol{\eta}}(t) = \mathbf{R}(\psi(t))\boldsymbol{x}_r(t) + \begin{bmatrix} \boldsymbol{V}_c \\ 0 \end{bmatrix} \ \forall t \in [t_0, t_0 + T_h] \quad (16b)$$

$$\boldsymbol{h}_m(\boldsymbol{\eta}(t), \boldsymbol{x}_r(t), t) \leq \mathbf{0} \ \forall t \in [t_0, t_0 + T_h] \quad (16c)$$

$$\boldsymbol{e}_m(\boldsymbol{\eta}(t_0)) = \mathbf{0}\,, \quad (16d)$$

where $T_h > 0$ is the prediction horizon. Notice that we use a purely kinematic model (16b) in the mid-level algorithm, which is done to reduce the computational load, ensuring that the algorithm can be run in real time as an MPC algorithm. In the hybrid architecture in Fig. 1, the output of the algorithm is passed to the short-term algorithm, which takes the vessel dynamics into account, ensuring that only feasible trajectories are fed to the vessel controllers and justifying neglecting kinetic feasibility in the mid-level algorithm. The constraint $x_r \in \mathbb{X}_r$ as well as static and dynamic obstacles are enforced in (16c), while the boundary constraints (16d) ensures that the trajectory starts at the current pose $\boldsymbol{\eta}(t_0)$.

To solve the OCP (16), we discretize it over $t \in [t_0, t_0 + T_h]$ using multiple shooting with N_p time steps. The vessel model (16b) is discretized using 4th-order Runge-Kutta, while the cost functional is discretized using forward Euler. For more details on the discretization, see [8]. This results in the NLP:

$$\begin{aligned} &\min_{\boldsymbol{w}} \ \phi_p(\boldsymbol{w}, \bar{\boldsymbol{p}}_{d,1:N_p}) + \phi_c(\boldsymbol{w}) \\ &\text{subject to} \\ &\boldsymbol{g}(\boldsymbol{w}, \boldsymbol{\eta}(t_0)) = \boldsymbol{0} \\ &\boldsymbol{h}(\boldsymbol{w}) \leq \boldsymbol{0}\,, \end{aligned} \tag{17}$$

where $\boldsymbol{w} = [\boldsymbol{\eta}_0^\top, \boldsymbol{x}_{r,0}^\top, \dots, \boldsymbol{\eta}_{N_p-1}^\top, \boldsymbol{x}_{r,N_p-1}^\top, \boldsymbol{\eta}_{N_p}^\top]^\top \in \mathbb{R}^{5N_p+3}$ is a vector of $5N_p + 3$ decision variables, $\bar{\boldsymbol{p}}_{d,1:N_p} = [\bar{\boldsymbol{p}}_{d,1}, \bar{\boldsymbol{p}}_{d,2}, \dots, \bar{\boldsymbol{p}}_{d,N_p}]$ is a sequence of desired positions and $\boldsymbol{g}(\boldsymbol{w}, \boldsymbol{\eta}(t_0)) \in \mathbb{R}^{3N_p+3}$ is a vector of $3N_p + 3$ shooting and boundary constraints.

A. *Objective function*

The objective function consists of two terms, where $\phi_p(\boldsymbol{w}, \bar{\boldsymbol{p}}_d(t))$ ensures convergence to the relative trajectory and $\phi_c(\boldsymbol{w})$ ensures that the resulting trajectory is readily observable. The functions are given as

$$\phi_p(\boldsymbol{w}, \bar{\boldsymbol{p}}_{d,1:N_p}) = \sum_{k=1}^{N_p} K_p q_p \left(\boldsymbol{p}_k, \bar{\boldsymbol{p}}_{d,k}\right) \tag{18a}$$

$$\phi_c(\boldsymbol{w}) = \sum_{k=0}^{N_p-1} K_{\dot{U}} q_{\dot{U}}(\dot{U}_k) + K_{\dot{\chi}} q_{\dot{\chi}}(\dot{\chi}_k), \tag{18b}$$

where the function $q_p(\cdot,\cdot)$ penalizes trajectories deviating from the desired trajectory, while and $q_{\dot{U}}(\cdot)$ and $q_{\dot{\chi}}(\cdot)$ penalize speed and course changes that are not readily observable to other vessels. The values $K_p, K_{\dot{U}}, K_{\dot{\chi}} > 0$ are tuning parameters, while $\bar{\boldsymbol{p}}_{d,k} = \bar{\boldsymbol{p}}_d(t_k)$ denotes the desired position at time t_k. Notice that neither the speed U or the course χ are elements in $\boldsymbol{w}$, but they can be computed using $U = \sqrt{u^2 + v^2}$ and $\chi = \psi + \arcsin \frac{v}{U}$, respectively. Furthermore, their derivatives are computed using finite differencing.

To measure the deviance from the desired trajectory, we want to use the Huber loss function, which is quadratic around the origin and resembles the absolute value function above a threshold $\sigma > 0$:

$$H(\rho) = \begin{cases} \frac{1}{2}\rho^2 & |\rho| \leq \sigma \\ \sigma(|\rho| - \frac{1}{2}\sigma) & |\rho| > \sigma\,. \end{cases} \tag{19}$$

Using the Huber function rather than a quadratic cost avoids the issue of the position error dominating over the other terms in the objective function when the position error is large, ensuring a similar behavior close and far from the desired trajectory [8]. The Huber function (19) can be used to define $q_p(\boldsymbol{p}, \boldsymbol{p}_d)$ as:

$$q_p(\boldsymbol{p}, \bar{\boldsymbol{p}}_d) = H(x - \bar{x}_d) + H(y - \bar{y}_d) \tag{20}$$

to evaluate a 2-dimensional loss [23].

The Huber function is, however, only C^1, resulting in (17) having a discontinuous Hessian matrix, making the NLP difficult to solve. In [8], this was circumvented using the pseudo-Huber function, which is a smooth approximation of (19). This can, however, cause numerical issues since the pseudo-Huber function introduces a square root in the objective function. To remedy this, the Huber function (19) can be implemented as a quadratic program (QP) [23]:

$$\begin{aligned} H(\rho) = &\min_{\omega,\mu} \ \sigma\omega + \frac{1}{2}\mu^2 \\ &\text{subject to} \\ &-\mu - \omega \leq \rho \leq \mu + \omega \\ &\omega \geq 0\,. \end{aligned} \tag{21}$$

This avoids the discontinuity issues by utilizing slack variables. To use (21) rather than (19) to implement the Huber function, we combine (17) and (21) to define a new NLP which is equivalent to (17) [23]:

$$\begin{aligned} &\min_{\boldsymbol{w},\boldsymbol{\omega},\boldsymbol{\mu}} \ \bar{\phi}_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}) + \phi_c(\boldsymbol{w}) \\ &\text{subject to} \\ &\boldsymbol{g}(\boldsymbol{w}, \boldsymbol{\eta}(t_0)) = \boldsymbol{0} \\ &\boldsymbol{h}(\boldsymbol{w}) \leq \boldsymbol{0} \\ &\bar{\boldsymbol{h}}_k(\boldsymbol{\eta}_k, \boldsymbol{\omega}_k, \boldsymbol{\mu}_k, \bar{\boldsymbol{p}}_{d,k}) \leq \boldsymbol{0} \ \forall k \in \{1, \dots, N_p\}\,, \end{aligned} \tag{22}$$

where $\boldsymbol{\omega} = [\boldsymbol{\omega}_1^\top, \boldsymbol{\omega}_2^\top, \dots, \boldsymbol{\omega}_{N_p}^\top]^\top \in \mathbb{R}^{2N_p}$ and $\boldsymbol{\mu} = [\boldsymbol{\mu}_1^\top, \boldsymbol{\mu}_2^\top, \dots, \boldsymbol{\mu}_{N_p}^\top]^\top \in \mathbb{R}^{2N_p}$ are slack variables. The function $\bar{\phi}_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu})$ is

$$\bar{\phi}_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}) = \sum_{k=1}^{N_p} K_p \left(\sigma \boldsymbol{1}^\top \boldsymbol{\omega}_k + \frac{1}{2}\boldsymbol{\mu}_k^\top \boldsymbol{\mu}_k\right), \tag{23}$$

and $\bar{\boldsymbol{h}}_k(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}) \in \mathbb{R}^{6N_p}$ encodes the constraints in (21):

$$\bar{\boldsymbol{h}}_k(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}, \bar{\boldsymbol{p}}_{d,k}) = \begin{bmatrix} \boldsymbol{v_k} + \boldsymbol{\mu_k} + \boldsymbol{p}_k - \bar{\boldsymbol{p}}_{d,k} \\ \boldsymbol{v_k} + \boldsymbol{\mu_k} - (\boldsymbol{p}_k - \bar{\boldsymbol{p}}_{d,k}) \\ -\boldsymbol{\omega}_k \end{bmatrix}. \tag{24}$$

The penalty terms in speed and course change motivates the algorithm to perform readily observable maneuvers by penalizing maneuvering with small speed and course changes more than using large changes. This is done by using a nonlinear cost based on a combination of a quadratic and decaying exponential function:

$$q(\rho; a, b) = a\rho^2 + (1 - e^{-\frac{\rho^2}{b}}), \tag{25}$$

where $a > 0$ and $b > 0$ are parameters. Using this function, the penalty terms in speed and course change are defined as

$$q_{\dot{U}}(\dot{U}) = \frac{100}{q(\dot{U}_{\max}; a_{\dot{U}}, b_{\dot{U}})} q(\dot{U}; a_{\dot{U}}, b_{\dot{U}}) \tag{26a}$$

$$q_{\dot{\chi}}(\dot{\chi}) = \frac{100}{q(\dot{\chi}_{\max}; a_{\dot{\chi}}, b_{\dot{\chi}})} q(\dot{\chi}; a_{\dot{\chi}}, b_{\dot{\chi}})\,. \tag{26b}$$

This results in changing course or speed with a high acceleration or turn rate is preferred over changes with low rates of change, see [8] for more details on the speed and course change penalty terms.

B. Obstacle handling and steady-state feasibility

The constraint $\boldsymbol{h}(\boldsymbol{w}) \leq \boldsymbol{0}$ ensures COLAV and that only steady-state feasible trajectories are specified.

As in the trajectory planning algorithm, static obstacles are avoided by using elliptic inequalities to define constraints. For the i-th static obstacle, we define the inequality

$$\boldsymbol{h}_{s_i}(\boldsymbol{w}) = \begin{bmatrix} h_o(x_1, y_1, x_{c,i}, y_{c,i}, x_{a,i}, y_{a,i}, \alpha_i) \\ \vdots \\ h_o(x_{N_p}, y_{N_p}, x_{c,i}, y_{c,i}, x_{a,i}, y_{a,i}, \alpha_i) \end{bmatrix} \leq \boldsymbol{0}, \tag{27}$$

where $h_o(\cdot)$ is defined in (11) and $x_{c,i}, y_{c,i}, x_{a,i}, y_{a,i}$ and α_i denote the x and y center coordinates, major and minor axis lengths and major axis orientation of the i-th static obstacle, respectively.

Dynamic obstacles are handled by letting the ellipse parameters be time varying. For the i-th dynamic obstacle, we define the inequality

$$\boldsymbol{h}_{m_i}(\boldsymbol{w}) = \begin{bmatrix} h_o(x_1, y_1, x_{c,i}(t_1), y_{c,i}(t_1), \\ x_{a,i}, y_{a,i}, \alpha_i(t_1)) \\ \vdots \\ h_o(x_{N_p}, y_{N_p}, x_{c,i}(t_{N_p}), y_{c,i}(t_{N_p}), \\ x_{a,i}, y_{a,i}, \alpha_i(t_{N_p})) \end{bmatrix} \leq \boldsymbol{0}, \tag{28}$$

where $x_c(t), y_c(t), x_{a,i}, y_{a,i}$ and $\alpha_i(t)$ denote the time-varying x and y center coordinates, major and minor axis lengths and major axis orientation of the i-th dynamic obstacle, respectively. We assume, without loss of generality, that the minor and major axis lengths are constant.

Given S static and M dynamic obstacles, we define the inequality

$$\begin{aligned} \boldsymbol{h}_o(\boldsymbol{w}) = \Big[\boldsymbol{h}_{s_1}(\boldsymbol{w})^\top \quad \dots \quad \boldsymbol{h}_{s_S}(\boldsymbol{w})^\top \\ \boldsymbol{h}_{m_1}(\boldsymbol{w})^\top \quad \dots \quad \boldsymbol{h}_{m_M}(\boldsymbol{w})^\top\Big]^\top \leq \boldsymbol{0}, \end{aligned} \tag{29}$$

which ensures avoidance of both static and dynamic obstacles.

Similarly as in [8], we ensure steady-state feasibility at each time step through a constraint $\boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_r) \leq \boldsymbol{0} \in \mathbb{R}^4$, which encodes the constraint $\boldsymbol{x}_r \in \mathbb{X}_r$. To ensure feasibility for the entire prediction horizon, we define the inequality

$$\boldsymbol{h}_{\boldsymbol{x}_r}(\boldsymbol{w}) = \Big[\boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,0})^\top \quad \dots \quad \boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,N_p-1})^\top\Big]^\top \leq \boldsymbol{0}. \tag{30}$$

Finally, the inequality constrains are combined as

$$\boldsymbol{h}(\boldsymbol{w}) = \begin{bmatrix} \boldsymbol{h}_o(\boldsymbol{w}) \\ \boldsymbol{h}_{\boldsymbol{x}_r}(\boldsymbol{w}) \end{bmatrix} \in \mathbb{R}^{(M+S+4)N_p}, \tag{31}$$

which is used in (22).

V. Simulation scenario and results

The scenario used for testing the hybrid architecture is shown in Fig. 5. It consists of two static obstacles (a and b) and one dynamic obstacle (c) with parameters listed in Table I. The dynamic obstacle (c) has the start position listed in the table, and continues with a speed of $5\,\mathrm{m/s}$ and course $-11.25°$. It comes in the way of the ship in the middle of the nominal trajectory, which requires the mid-level algorithm to plan a trajectory around it.

TABLE I
OBSTACLE PARAMETERS.

Obstacle	x_c	y_c	x_a	y_a	α
a	5500 m	2000 m	4000 m	1000 m	$-5°$
b	2000 m	6500 m	4000 m	1000 m	$5°$
c*	500 m	4800 m	1000 m	400 m	$-11.25°$

* Obstacle c continues with speed $5\,\mathrm{m/s}$ and course $-11.25°$ after initialization.

TABLE II
SIMULATION AND TUNING PARAMETERS.

Param.	Value	Comment
$t_{f,\max}$	1500 s	High-level planner time constraint
$[V_x, V_y]$	$[2.5, 0]\,\mathrm{m/s}$	Ocean current velocity
N_s	165	Number of simulation steps
h	10 s	Mid-level step size
N_p	36	Mid-level prediction steps
K_p	$2 \cdot 10^{-2}$	Position error scaling
$K_{\dot{U}}$	2	SOG-derivative penalty term scaling
$K_{\dot{\chi}}$	7.5	Course-derivative penalty term scaling
$[a_{\dot{U}}, b_{\dot{U}}]$	$[8, 2.5 \cdot 10^{-4}]$	SOG-derivative penalty term parameters
$[a_{\dot{\chi}}, b_{\dot{\chi}}]$	$[112, 1.875 \cdot 10^{-4}]$	Course-derivative penalty term parameters

The algorithms are run with the parameters listed in Table II. The high-level algorithm is implemented using the pseudospectral optimal control package DIDO for MATLAB on a desktop computer [7]. The mid-level algorithm is implemented using CASADI [24] and IPOPT [25] for MATLAB on a desktop computer. Since the mid-level algorithm is MPC-based, a new solution is computed at every time step, where only the first step of the solution is implemented.

Fig. 5 shows how the mid-level algorithm modifies the nominal trajectory to produce collision-free and observable maneuvers. Fig. 6 shows speeds and course angle plots from the mid-level algorithm. From Fig. 5, we see that when the nominal trajectory is free from obstacles, the mid-level algorithm tracks it well, while performing readily observable maneuvers. Deviations from the nominal trajectory occur from time stamp t_2, where the mid-level algorithm chooses to keep a different course to pass behind the dynamic obstacle. Fig. 6 shows that the SOG U is held at a lower value to be able to pass behind the obstacle between 400 s and 650 s, which also makes the maneuver more observable, according to the objective function (18) which discourages small changes in SOG. The figure also shows that course changes are performed in large steps, in excess of 30°, which is accepted as *readily observable* maneuvers even in restricted visibility [4]. When the obstacle is passed, the mid-level algorithm again tracks the reference speed and moves

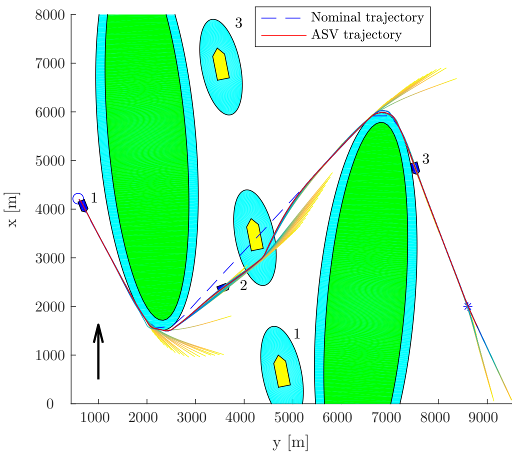


Fig. 5. Test scenario with static and moving obstacles. Nominal trajectory (striped blue) and resulting trajectory (red) together with predicted trajectory at every time step of the MPC. The predicted trajectories start with blue color at t_0, and gets more yellow towards the end of the prediction horizon $t_0 + T_h$. The green ellipses are the "real" static obstacles, contained in blue ellipses which are safety margins. The moving obstacle is a yellow patch encapsulated by a blue ellipse representing its safety margin. The time stamps (1, 2, 3) represent times 20 s, 540 s and 1190 s, respectively. The arrow in the lower-left corner is the ocean current direction.

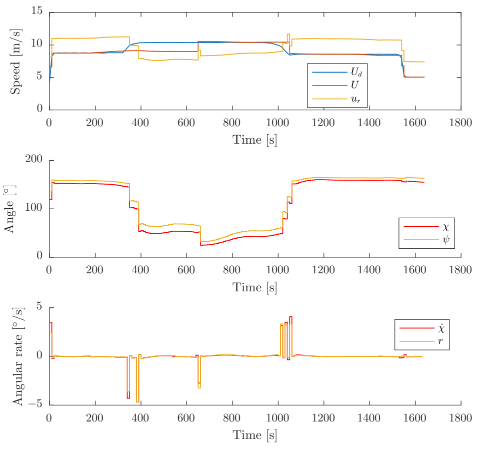


Fig. 6. Desired SOG, actual SOG and relative surge velocity (top), course and yaw angle (middle), and course and turn rate (bottom) over the duration of the scenario.

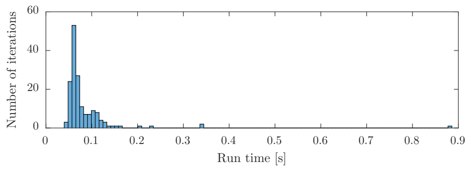


Fig. 7. Run times for the mid-level algorithm.

back to the nominal trajectory. In addition, notice that the relative surge velocity u_r is actively controlled to ensure readily observable changes in the SOG, especially as the ASV changes course at around 350 s and 1050 s.

The run time for the high-level planner is between 150 s and 200 s, depending on the scenario. Since the planner is run offline before the start of the scenario, high run times are not detrimental. For the mid-level algorithm, the majority of the algorithm steps are completed in less than 0.1 s, as seen in Fig. 7. There are some outliers, but the maximum observed run time is approximately 0.9 s, which with a step size of 10 s is considered to be real-time feasible. Ideally, the mid-level algorithm provides safe commands in a timely manner, but the hybrid architecture allows the short-term COLAV algorithm to serve as a backup in case of a malfunction above it in the hierarchy.

VI. Conclusion

We have developed and verified parts of a hybrid COLAV architecture for ASVs. The simulation results show that the architecture enables an ASV to avoid both static and dynamic obstacles and produces readily observable maneuvers in compliance with Rule 8 of COLREGs. In the absence of dynamic obstacles, the ASV tracks an energy-optimized trajectory.

Possibilities for further work include:

- Complete the development and implementation of the three-layered hybrid architecture by including a short-term COLAV algorithm, e.g. [5], and perform full-scale experiments.
- Investigate the possibility of performing periodic high-level re-planning to increase optimality of the trajectory by taking into account updated information about environmental factors and ship state.
- Investigate if an alternative OCP solver can decrease run time for the high-level algorithm.
- Include more COLREGs rules in the mid-level algorithm, specifically the required behavior in *overtaking*, *crossing* and *head-on* situations.

Acknowledgments

This work was supported by the Research Council of Norway through project number 269116, project number 244116, as well as the Centers of Excellence funding scheme with project number 223254.

References

[1] Kongsberg Maritime. (2018). Autonomous ship project, key facts about YARA Birkeland, [Online]. Available: `https://www.km.kongsberg.com/ks/web/nokbg0240.nsf/AllWeb/4B8113B707A50A4FC125811D00407045` (visited on 2018-11-10).

[2] O. Levander, "Autonomous ships on the high seas," *IEEE Spectrum*, vol. 54, no. 2, pp. 26–31, 2017-02.

[3] DNV GL, "Sammenstilling av grunnlagsdata om dagens skipstrafikk og drivstofforbruk, Miljøtiltak for maritim sektor," Norwegian, Klima- og miljødepartementet, Tech. Rep. 2014-1667, 2014.

[4] A. N. Cockcroft and J. N. F. Lameijer, *A Guide to the Collision Avoidance Rules*. Elsevier Butterworth-Heinemann, 2004, ISBN: 0-7506-6179-8.

[5] B.-O. H. Eriksen, M. Breivik, E. F. Wilthil, A. L. Flåten, and E. Brekke, "The branching-course MPC algorithm for maritime collision avoidance," Submitted to Journal of Field Robotics.

[6] G. Bitar, M. Breivik, and A. M. Lekkas, "Energy-optimized path planning for autonomous ferries," in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*, Opatija, Croatia, 2018, pp. 389–394.

[7] I. M. Ross and M. Karpenko, "A review of pseudospectral optimal control: From theory to flight," *Annual Reviews in Control*, vol. 36, no. 2, pp. 182–197, 2012.

[8] B.-O. H. Eriksen and M. Breivik, "MPC-based mid-level collision avoidance for ASVs using nonlinear programming," in *Proc. of the IEEE Conference on Control Technology and Applications (CCTA)*, Mauna Lani, HI, USA, 2017, pp. 766–772.

[9] I. B. Hagen, D. K. M. Kufoalor, E. F. Brekke, and T. A. Johansen, "MPC-based collision avoidance strategy for existing marine vessel guidance systems," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, Australia, 2018, pp. 7618–7623.

[10] P. Švec, B. C. Shah, I. R. Bertaska, J. Alvarez, A. J. Sinisterra, K. von Ellenrieder, M. Dhanak, and S. K. Gupta, "Dynamics-aware target following for an autonomous surface vehicle operating under COLREGs in civilian traffic," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 2013, pp. 3871–3878.

[11] M. Abdelaal and A. Hahn, "NMPC-based trajectory tracking and collision avoidance of unmanned surface vessels with rule-based colregs confinement," in *Proc. of the IEEE Conference on Systems, Process and Control (ICSPC)*, 2016.

[12] M. Candeloro, A. M. Lekkas, and A. J. Sørensen, "A Voronoi-diagram-based dynamic path-planning system for underactuated marine vessels," *Control Engineering Practice*, vol. 61, pp. 41–54, 2017.

[13] A. Tsourdos, B. White, and M. Shanmugavel, *Cooperative Path Planning of Unmanned Aerial Vehicles*. John Wiley & Sons, Inc., 2010, 190 pp., ISBN: 978-0-470-74129-0.

[14] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path plannning," Tech. Rep., 1998.

[15] H.-T. L. Chiang and L. Tapia, "COLREG-RRT: An RRT-based COLREGS-compliant motion planner for surface vehicle navigation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2024–2031, 2018.

[16] B.-O. H. Eriksen, E. F. Wilthil, A. L. Flåten, E. F. Brekke, and M. Breivik, "Radar-based maritime collision avoidance using dynamic window," in *Proc. of the IEEE Aerospace Conference*, Big Sky, MT, USA, 2018.

[17] Y. Kuwata, M. T. Wolf, D. Zarzhitsky, and T. L. Huntsberger, "Safe maritime autonomous navigation with COLREGs, using velocity obstacles," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 110–119, 2014.

[18] Ø. A. G. Loe, "Collision avoidance for unmanned surface vehicles," Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2008.

[19] G. Casalino, A. Turetta, and E. Simetti, "A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields," in *Proc. of the IEEE Oceans Conference*, IEEE, Bremen, Germany, 2009, pp. 1–8.

[20] B.-O. H. Eriksen and M. Breivik, "Modeling, identification and control of high-speed ASVs: Theory and experiments," in *Sensing and Control for Autonomous Vehicles*, T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer, Eds., Springer International Publishing, 2017, pp. 407–431.

[21] B.-O. H. Eriksen and M. Breivik, "A model-based speed and course controller for high-speed ASVs," in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*, Opatija, Croatia, 2018, pp. 317–322.

[22] T. I. Fossen, *Guidance and Control of Ocean Vehicles*. John Wiley & Sons, 1994, ISBN: 0-471-94-113-1.

[23] S. Gros and M. Zanon, "Penalty functions for handling large deviation of quadrature states in NMPC," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3848–3860, 2017.

[24] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, 2018.

[25] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 25–57, 2005.

Paper H Short-term ASV collision avoidance with static and moving obstacles

Preprint of **B.-O. H. Eriksen** and M. Breivik, “Short-term ASV collision avoidance with static and moving obstacles”, 2019, Submitted to Modeling, Identification and Control, available at https://arxiv.org/abs/1907.04877.

Short-term ASV Collision Avoidance with Static and Moving Obstacles

Bjørn-Olav H. Eriksen and Morten Breivik

Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway.
E-mail: {bjorn-olav.h.eriksen, morten.breivik}@ieee.org

August 5, 2019

Abstract

This article considers collision avoidance (COLAV) for both static and moving obstacles using the branching-course model predictive control (BC-MPC) algorithm, which is designed for use by autonomous surface vehicles (ASVs). The BC-MPC algorithm originally only considered COLAV of moving obstacles, so in order to make the algorithm also be able to avoid static obstacles, we introduce an extra term in the objective function based on an occupancy grid. In addition, other improvements are made to the algorithm resulting in trajectories with less wobbling. The modified algorithm is verified through full-scale experiments in the Trondheimsfjord in Norway with both virtual static obstacles and a physical moving obstacle. A radar-based tracking system is used to detect and track the moving obstacle, which enables the algorithm to avoid obstacles without depending on vessel-to-vessel communication. The experiments show that the algorithm is able to simultaneously avoid both static and moving obstacles, while providing clear and readily observable maneuvers. The BC-MPC algorithm is compliant with rules 8, 13 and 17 of the the International Regulations for Preventing Collisions at Sea (COLREGs), and favors maneuvers following rules 14 and 15.

Keywords: Autonomous surface vehicles, collision avoidance, model predictive control

1 Introduction

All parts of society are currently being automated at a rapid pace. One example is the development of autonomous cars, as exemplified by the development efforts made by e.g. Tesla, Google and Uber. Such a trend is also ongoing in the maritime domain, where autonomous technology presents opportunities for increased cost efficiency, in addition to reducing the environmental impact of goods and passenger transport. One example of this is the Yara Birkeland project in Norway, where an electrically-powered autonomous cargo ship will replace 40000 diesel-powered truckloads of fertilizer each year by 2022 (Paris, 2017). Furthermore, it is reported that in excess of 75% of maritime accidents are caused by human errors (Chauvin, 2011; Levander, 2017), which also reveals a potential for increased safety by introducing autonomous technology at sea. Employing ASVs in areas where other vessels are present does, however, require a robust COLAV system in order to avoid collisions and operate safely.

There exists several algorithms for ASV COLAV, e.g. the velocity obstacle (VO) algorithm (Kuwata et al., 2014), the A* algorithm (Schuster et al., 2014) and algorithms based on model predictive control (MPC) and optimization (Benjamin et al., 2006; Švec et al., 2013; Abdelaal and Hahn, 2016; Hagen et al., 2018). These algorithms are, however, designed with the idea of "one size fits all", where the same algorithm is used to solve both situations requiring proactive and reactive behaviors. A challenge

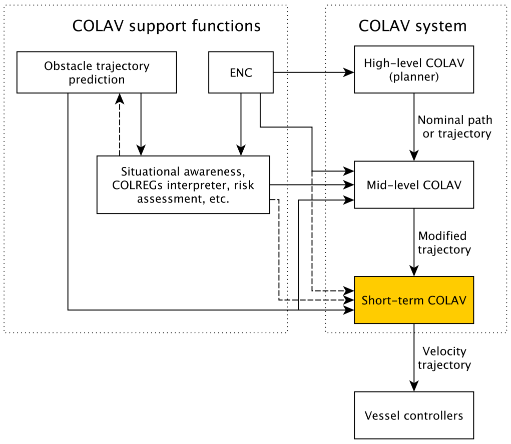

COLAV support functions
COLAV system
Obstacle trajectory prediction
ENC
High-level COLAV (planner)
Nominal path or trajectory
Situational awareness, COLREGs interpreter, risk assessment, etc.
Mid-level COLAV
Modified trajectory
Short-term COLAV
Velocity trajectory
Vessel controllers

Figure 1: A hybrid architecture with three layers. The support functions provide relevant information for the COLAV algorithms, including prediction of obstacle trajectories, static obstacles from electronic nautical charts (ENC) and situational awareness in the form of COLREGs situations. Courtesy of (Eriksen et al., 2019b).

with this approach is that the algorithm must be able to solve problems of a wide range sufficiently well, which makes the algorithm difficult to design and tune. A different approach is to utilize a hybrid architecture (Loe, 2008; Casalino et al., 2009), where the complementary strengths of different algorithms can be combined in a layered architecture. An example of a hybrid architecture is shown in Figure 1, where the COLAV system is divided into three layers, namely a high-level, mid-level and a short-term COLAV algorithm. The high-level planner performs long-term planning by finding a path or trajectory from an initial position to a goal position while being able to avoid static obstacles, satisfy time constraints and minimize energy consumption. The mid-level algorithm attempts to follow the planned path or trajectory from the high-level planner, while making local modifications in order to avoid moving obstacles. This algorithm should be designed to comply with the maneuvering rules of the COLREGs, which dictates how vessels should behave in situations where there exists a risk of collision with other vessels (Cockcroft and Lameijer, 2004). The short-term COLAV algorithm inputs the modified trajectory from the mid-level algorithm, and should have low computational requirements ensuring that the COLAV system can react to sudden changes in the environment. This algorithm should also serve as a final safety barrier in situations where e.g. the mid-level algorithm fails to find a solution (Eriksen and Breivik, 2017b). In addition, the short-term algorithm should have a shorter planning horizon than the mid-level algorithm, making it inherently capable of handling situations where the COLREGs may require ignoring the maneuvering aspects of rules 14 and 15 when moving obstacles do not comply with the COLREGs. The algorithm should, however, maneuver in accordance with rules 14 and 15 when the situation allows it.

The authors have performed a significant amount of work on the hybrid architecture in Figure 1, concerning e.g. model-based vessel controllers (Eriksen and Breivik, 2017a, 2018), short-term COLAV (Eriksen et al., 2018, 2019b), mid-level COLAV (Eriksen and Breivik, 2017b) and a high-level planner interfaced to the mid-level algorithm (Bitar et al., 2019). In an upcoming article (Eriksen et al., 2019a), we populate the hybrid architecture with algorithms including the BC-MPC algorithm discussed in this article, and demonstrate COLAV compliant with COLREGs rules 8 and 13–17 in simulations. Work has also been performed on obstacle trajectory prediction (Hexeberg et al., 2017; Dalsnes et al., 2018). For the short-term COLAV layer, we initially focused on the dynamic window (DW) algorithm, using a radar-based tracking system for detecting and tracking obstacles (Wilthil et al., 2017). The reason for using exteroceptive sensors such as radars for detecting obstacles is that they do not depend on vessel-to-vessel communication or collaboration with other vessels, hence enabling avoidance of vessels which do not have or use automatic identification system (AIS) transponders. Another questionable aspect of AIS is that other vessels may provide incorrect information (Harati-Mokhtari et al., 2007), which can be difficult to detect and handle. However, there is a fair amount of noise on obstacle estimates originating from systems using exteroceptive sensors, which the DW algorithm was shown not to handle sufficiently well in full-scale experiments (Eriksen et al., 2018). We therefore developed the BC-MPC algorithm for short-term COLAV (Eriksen et al., 2019b), which is based on MPC and designed to be robust to obstacle estimate noise. This algorithm is shown to have good performance in full-scale experiments, but originally only accounts for moving obstacles.

In this article, we further develop the BC-MPC algorithm to also handle avoidance of static obstacles in addition to moving obstacles, as well as producing trajectories with less wobbling. The modified algorithm is verified in full-scale experiments in Trondheimsfjorden, Norway, showing good performance. The experiments are performed with virtual static obstacles, while a moving obstacle is detected and tracked using a radar, not depending on vessel-to-vessel communication.

The rest of this article is organized as follows: Section 2 presents the BC-MPC algorithm and the modifications we do to it, Section 3 presents the experimental setup and results, while Section 4 concludes the article and points to possibilities for further work.

2 The BC-MPC algorithm

The BC-MPC algorithm (Eriksen et al., 2019b) is a COLAV algorithm designed using sample-based MPC, intended for short-term COLAV for ASVs. Sample-based MPC algorithms are based on computing an objective function over a finite discrete search space and selecting the optimized solution, rather than utilizing search algorithms as in gradient-based algorithms. A benefit of sample-based algorithms is that they do not have problems with solving highly nonlinear and non-convex problems, which in general is difficult for gradient-based algorithms. This makes sample-based algorithms well suited for use in the short-term layer in Figure 1. Furthermore, the BC-MPC algorithm is designed to be robust with respect to noisy obstacle estimates, which is a significant source of disturbance when using exteroceptive sensors such as radars for detecting and tracking obstacles.

With respect to the COLREGs, the BC-MPC algorithm complies with rules 8, 13 and 17, and favors maneuvers following rules 14 and 15. In cases where the algorithm chooses to ignore the maneuvering aspects of rules 14 and 15, which can be required when rule 17 revokes a stand-on obligation, the maneuvers have an extended clearance to obstacles.

At each iteration, the algorithm computes a search space consisting of a finite number of possible trajectories, which each contains a sequence of maneuvers. Given this search space, an objective function is computed on the trajectories, and the optimized trajectory is selected and used as the reference to the vessel controllers which control the speed over ground (SOG) and course. The algorithm is based on MPC, hence only the first part of the optimized trajectory is used before a new solution is computed and implemented.

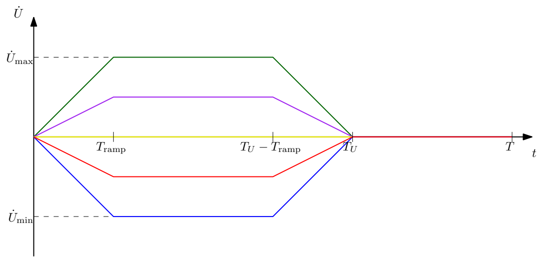
This section presents an overview of the BC-MPC algorithm. Interested readers are referred to Eriksen et al. (2019b) for more details on the algorithm. In addition, this section presents modifications enabling the algorithm to perform static obstacle avoidance and produce trajectories with less wobbling than the original algorithm.

2.1 Trajectory generation

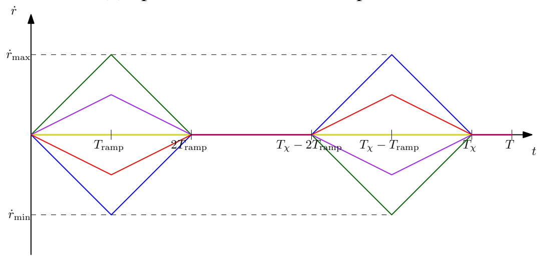
At each iteration, a new finite search space of possible trajectories is generated. Every trajectory contains a number of sub-trajectories, each containing one maneuver. This naturally forms a tree structure, with the nodes representing vessel configurations and edges representing maneuvers. The initial condition is used as the root node, and the depth of the tree is equal to the number of maneuvers in each trajectory.

The trajectory generation is performed by a repeatable maneuver-generation procedure, which when given a vessel configuration computes a set of sub-trajectories each containing one maneuver. Piecewise linear acceleration profiles in speed and course serve as a template for the maneuvers. An example of 5 motion primitives based on the acceleration profiles in speed and course is shown in Figure 2. The acceleration profiles are dependent on the step time length (the maneuver time length) $T > 0$, the ramp time $T_{\text{ramp}} \in (0, \min(\frac{T_U}{2}, \frac{T_\chi}{4})]$ and the speed and course maneuver lengths, $T_U, T_\chi \in (0, T]$, respectively. Given a current vessel velocity, the maximum and minimum speed and course accelerations $\dot{U}_{\max}$, $\dot{U}_{\min}$, $\dot{r}_{\max}$ and $\dot{r}_{\min}$ are computed using a vessel model.

To improve the convergence properties of the algorithm, we employ a guidance function which can modify some of the trajectories in the search space. This is done by moving the closest acceleration sample in speed and course to a desired acceleration generated by the guidance function, if this is inside the feasible acceleration region.



(a) Speed acceleration motion primitives



(b) Course acceleration motion primitives.

Figure 2: Acceleration motion primitives, where T is the step time, T_{ramp} denotes the ramp time, while T_U and T_χ are the SOG and course maneuver time lengths, respectively. The symbols $\dot{U}_{\max}$, $\dot{U}_{\min}$, $\dot{r}_{\max}$ and $\dot{r}_{\min}$ denote the acceleration limits of the vessel at the initial vessel state. Courtesy of (Eriksen et al., 2019b).

Desired speed and course trajectories $U_d(t)$ and $\chi_d(t)$ are generated by analytically integrating the acceleration motion primitives. Numerical examples of 5 speed and 5 course trajectories are shown in figures 3 and 4. It should be noted that these trajectories are intended as reference trajectories for the vessel controllers, hence they are initiated in an open-loop fashion with the current desired speed and course in order to ensure continuous references for the vessel controllers. The desired speed and course trajectories are joined together in a union set of desired velocity trajectories:

$$\mathcal{U}_d = \{U_{d,1}(t), U_{d,2}(t), \ldots, U_{d,N_U}(t)\} \times \{\chi_{d,1}(t), \chi_{d,2}(t), \ldots, \chi_{d,N_\chi}(t)\}, \quad (1)$$

resulting in a total of $N_U \cdot N_\chi$ desired velocity trajectories where $N_U \in \mathbb{Z}^+$ and $N_\chi \in \mathbb{Z}^+$ are the number of speed and course motion primitives. To include feedback in the trajectory generation, we use an error model of the vessel to generate feedback-corrected speed and course trajectories $\bar{U}_d(t)$ and $\bar{\chi}_d(t)$, which similarly as in (1) is combined in a set $\bar{\mathcal{U}}_d$. The feedback-corrected speed and course trajectories are used to generate feedback-corrected predicted pose trajectories:

$$\bar{\mathcal{H}} = \left\{\bar{\boldsymbol{\eta}}(t; \bar{U}(t), \bar{\chi}(t)) \middle| (\bar{U}(t), \bar{\chi}(t)) \in \bar{\mathcal{U}}\right\}, \quad (2)$$

where $\bar{\boldsymbol{\eta}}(t; \bar{U}(t), \bar{\chi}(t))$ denotes a kinematic simulation procedure to obtain the vessel pose.

A full trajectory search space is created by first generating a set of sub-trajectories by using the maneuver-generation procedure initialized with the initial vehicle pose. At this stage, the prediction tree has a depth of one with the initial vessel pose as the root node and a set of leaf nodes each reached by one maneuver. Following this, we append the trajectories with another maneuver by repeating the maneuver-generation procedure, initialized on each of the leaf nodes, which increases the depth of the trajectory prediction tree with one level. This is repeated until the trajectory prediction tree has the desired depth, i.e. each trajectory has the desired number of maneuvers. This concept is illustrated in Figure 5. The acceleration profile parameters and number of speed and course motion primitives can be level-dependent, which allows for shaping the maneuvers differently and avoiding exponential growth with the number of levels. To reduce

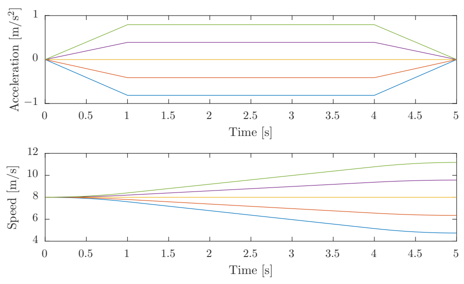


Figure 3: Example of 5 speed trajectories with ramp time $T_{\text{ramp}} = 1$ s, and maneuver and step time lengths $T_U = T = 5$ s. Acceleration is shown in the top plot, while speed is shown in the bottom plot. Courtesy of (Eriksen et al., 2019b).

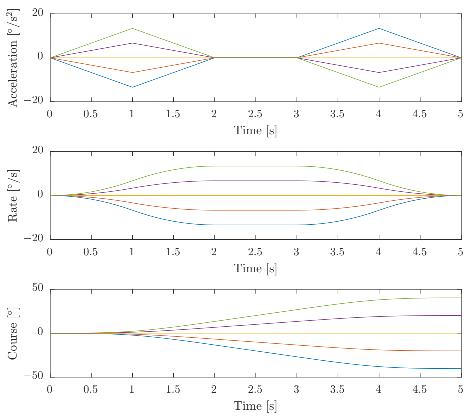


Figure 4: Example of 5 course trajectories with ramp time $T_{\text{ramp}} = 1$ s, and maneuver and step time lengths $T_\chi = T = 5$ s. Acceleration is shown in the top plot, rate in the middle plot and course in the bottom plot. Courtesy of (Eriksen et al., 2019b).

the complexity in tuning the algorithm, we use the same ramp time T_{ramp} and speed and course maneuver lengths T_U and T_χ throughout each level. For a desired trajectory tree depth B (B maneuvers in each trajectory), this leaves us with deciding the step time lengths of each level $\boldsymbol{T} = [T_1, T_2, \ldots, T_B]$, and the number of speed and course maneuvers at each level $\boldsymbol{N}_U = [N_{U,1}, N_{U,2}, \ldots, N_{U,B}]$ and $\boldsymbol{N}_\chi = [N_{\chi,1}, N_{\chi,2}, \ldots, N_{\chi,B}]$.

A set of feedback-corrected predicted pose trajectories for a trajectory generation with $B = 3$ levels is shown in Figure 6. The ramp time is $T_{\text{ramp}} = 1$ s, and the speed and course maneuver lengths are $T_U = T_\chi = 5$ s. The step time lengths are $\boldsymbol{T} = [20, 30, 30]$ s, and the number of speed and course maneuvers are $\boldsymbol{N}_U = [1, 1, 1]$ and $\boldsymbol{N}_\chi = [5, 3, 3]$.

2.2 Selecting the optimized trajectory

Given a search space of vessel trajectories and a desired trajectory $\boldsymbol{p}_d(t) \in \mathbb{R}^2$, we solve an optimization problem to find the optimized desired velocity trajectory $\boldsymbol{u}_d^*(t) = \begin{bmatrix} U_d^*(t) & \chi_d^*(t) \end{bmatrix}^\top$ as:

$$\boldsymbol{u}_d^*(t) = \underset{(\bar{\boldsymbol{\eta}}_k(t), \boldsymbol{u}_{d,k}(t)) \in (\bar{\mathcal{H}}, \mathcal{U}_d)}{\operatorname{argmin}} G(\bar{\boldsymbol{\eta}}_k(t), \boldsymbol{u}_{d,k}(t); \boldsymbol{p}_d(t)). \quad (3)$$

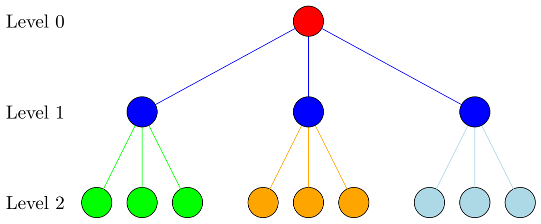


Figure 5: Illustration of a trajectory prediction tree with two levels. The red node is the root node containing the initial vessel configuration. Other colors group nodes and edges associated with each maneuver-generation procedure, which generate three maneuvers each time (given by combinations of N_U and N_χ satisfying $N_U \cdot N_\chi = 3$). The tree contains a total of nine trajectories, each consisting of two sub-trajectories.

The objective function is given as:

$$\begin{aligned} G(\bar{\boldsymbol{\eta}}(t), \boldsymbol{u}_d(t); \boldsymbol{p}_d(t)) &= w_{\text{al}} \text{align}(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_d(t)) \\ &\quad + w_{\text{av,m}} \text{avoid}_{\text{m}}(\bar{\boldsymbol{\eta}}(t)) + w_{\text{av,s}} \text{avoid}_{\text{s}}(\bar{\boldsymbol{\eta}}(t)) \\ &\qquad + w_{\text{t},U} \text{tran}_U(\boldsymbol{u}_d(t)) + w_{\text{t},\chi} \text{tran}_\chi(\boldsymbol{u}_d(t)), \quad (4) \end{aligned}$$

where $w_{\text{al}}, w_{\text{av,m}}, w_{\text{av,s}}, w_{\text{t},U}, w_{\text{t},\chi} > 0$ are tuning parameters.

The align$(\cdot)$ function assigns a value to following the desired trajectory $\boldsymbol{p}_d(t)$. The avoid$_{\text{m}}(\cdot)$ function assigns a cost to traveling close to moving obstacles, which depends on the distance to an obstacle for each point on the predicted trajectories. The maneuvering rules in the COLREGs, rules 13–15, require the vessel to maneuver to starboard in head-on situations, and recommend to pass behind an obstacle if the obstacle approaches from the starboard side. To motivate the algorithm to follow these rules, while being free to ignore the specific maneuvering aspects if required in situations where the other vessel violates the COLREGs, we use the obstacle regions in Figure 7 when calculating this cost. The regions can be interpreted as follows: the margin region is allowable to enter, the safety region is not desirable to enter, while the collision region should not be entered. Notice that the algorithm will require a larger clearance in situations where the maneuvering rules in the COLREGs are ignored, e.g. if maneuvering to port in a head-on situation. See Eriksen

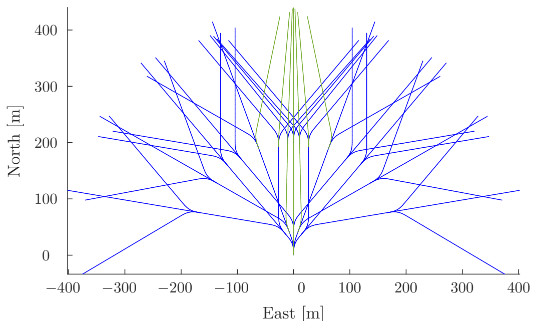


Figure 6: A set of predicted pose trajectories with three levels. Notice how the guidance function shifts some of the maneuvers, marked in dark green, to converge towards the desired trajectory, which is a straight-north trajectory from the initial pose (not shown in the figure). For illustration purposes, the trajectories only contain course maneuvers.

et al. (2019b) for more details on the align(·) and avoid$_\text{m}$(·) terms.

In this article, we introduce the avoid$_\text{s}$(·), tran$_U$(·) and tran$_\chi$(·) terms. The avoid$_\text{s}$(·) term assigns a cost to avoiding static obstacles, while tran$_U$(·) and tran$_\chi$(·) are transitional cost terms increasing the robustness to noise. These terms will be discussed in detail in the following two sections.

2.3 Static obstacle avoidance

Static obstacles are modeled using an occupancy grid, which allows for easy representation of obstacles with arbitrary shapes like e.g. land and islands. In addition, static obstacles are padded with a decaying gradient to introduce some smoothness to the static obstacle avoidance function. Given an occupancy grid $O(\boldsymbol{p}) \in [0, 100]$ where $O(\boldsymbol{p}) = 100$ and $O(\boldsymbol{p}) = 0$ represents an occupied and empty cell, respectively, we define the static obstacle term as:

$$\text{avoid}_\text{s}(\bar{\boldsymbol{\eta}}(t)) = \int_{t_0}^{t_0+T} O(\bar{\boldsymbol{p}}(\gamma))\mathrm{d}\gamma, \qquad (5)$$

where t_0 denotes the initial time and $\bar{\boldsymbol{\eta}}(t) = \begin{bmatrix}\bar{\boldsymbol{p}}(t)^\top & \bar{\psi}(t)\end{bmatrix}^\top$.

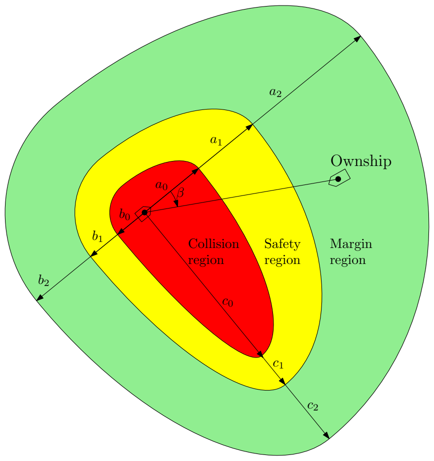


Figure 7: Avoidance cost regions centered at the moving obstacle, each constructed by one circular and three elliptical segments. The green, yellow and red regions are named the margin, safety and collision regions, respectively. The avoidance cost increases linearly with different gradients inside the green and yellow regions, while the cost is constant inside the red region. The variables a_i, b_i and c_i, $i \in \{1, 2, 3\}$ denote the region sizes, where $c_i = b_i + d_\text{COLREGs}$ with d_COLREGs controlling the COLREGs expansion. Courtesy of (Eriksen et al., 2019b).

2.4 Speed and course transitional costs

In order to improve the robustness to noise on obstacle estimates, transitional cost is included in the objective function, which penalizes changing the planned trajectory from iteration to iteration. In Eriksen et al. (2019b), a single transitional cost term is used, which introduces a cost if one selects a different speed and/or course than the one closest to the one selected in the previous iteration. Note that the trajectory prediction is based on sampling the possible acceleration of the vessel in the current iteration, which implies that the exact trajectory selected in the previous iteration may not exist in the current search space.

Here, it is proposed to split the transitional cost term into separate speed and course terms. This motivates the algorithm to not alter the course if the speed is changed and vice versa, which would not be the case when using a single transitional cost term. The transitional cost terms are defined as:

$$\mathrm{tran}_U(\boldsymbol{u}_d(t)) = \begin{cases} 1, & \int_{t_0}^{t_0+T_1} \left|U_d(\gamma) - U_d^-(\gamma)\right| \mathrm{d}\gamma > e_{U,\min} \\ 0, & \text{else}, \end{cases} \tag{6}$$

$$\mathrm{tran}_\chi(\boldsymbol{u}_d(t)) = \begin{cases} 1, & \int_{t_0}^{t_0+T_1} \left|\chi_d(\gamma) - \chi_d^-(\gamma)\right| \mathrm{d}\gamma > e_{\chi,\min} \\ 0, & \text{else}, \end{cases} \tag{7}$$

with $\boldsymbol{u}_d(t) = \begin{bmatrix} U_d(t) & \chi_d(t) \end{bmatrix}^\top$. The variables $U_d^-(t)$ and $\chi_d^-(t)$ denote the current desired velocity trajectory tracked by the vessel controllers, and T_1 is the step time of the first trajectory maneuver. The variables $e_{U,\min}$ and $e_{\chi,\min}$ denote the minimum difference between the current desired velocity trajectory and the candidates:

$$\begin{aligned} e_{U,\min} &= \min_{\boldsymbol{u}_d(t)\in\mathcal{U}_d} \int_{t_0}^{t_0+T_1} \left|U_d(\gamma) - U_d^-(\gamma)\right| \mathrm{d}\gamma \\ e_{\chi,\min} &= \min_{\boldsymbol{u}_d(t)\in\mathcal{U}_d} \int_{t_0}^{t_0+T_1} \left|\chi_d(\gamma) - \chi_d^-(\gamma)\right| \mathrm{d}\gamma. \end{aligned} \tag{8}$$

3 Experimental results

The modified BC-MPC algorithm was tested in full-scale experiments in the Trondheimsfjord in Norway on the 27th of September 2018. This section describes the experimental setup and presets results from the experiments.

3.1 Experimental setup

The experimental setup was similar to the setup reported in Eriksen et al. (2019b), using the Telemetron ASV from Maritime Robotics as the ownship and the Ocean Space Drone 1 (OSD1) from Kongsberg Seatex as the moving obstacle. In addition, virtual static obstacles, expanded with a padding radius, were used to emulate static obstacles. The padding radius was selected as 150 m in most of the experiments. Notice that this padding radius only relates to static obstacles and that safety margins for moving obstacles are enforced by the obstacle regions in Figure 7. The Telemetron ASV, shown in Figure 8, is a 26-foot high-speed ASV capable of speeds up to 18 m/s and equipped for both manned and unmanned operations. The OSD1, shown in Figure 9, is a modified offshore lifeboat with a length of 12 m, and was steered at a constant speed of 5 knots during the experiments. The OSD1 played the role of a moving obstacle in the experiments, and was detected and tracked using a radar-based tracking system, which is discussed in detail in Wilthil et al. (2017) and Wilthil (2019). Both the BC-MPC algorithm and the radar tracking system was implemented using the Robot Operating System (ROS), and was run on a processing platform with an Intel® i7 3.4 GHz CPU running Ubuntu 16.04 Linux onboard the Telemetron ASV. See Table 1 for specifications on the Telemetron ASV and the sensor system.

The BC-MPC algorithm was run at a rate of 0.2 Hz with the parameters in Table 2. At sea, vessels typically maneuver with large margins, making it safe to run the BC-MPC

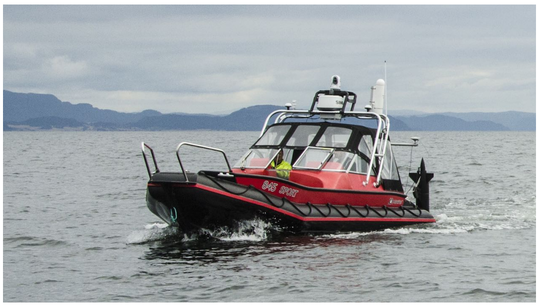


Figure 8: The Telemetron ASV, owned and operated by Maritime Robotics. Courtesy of Maritime Robotics.

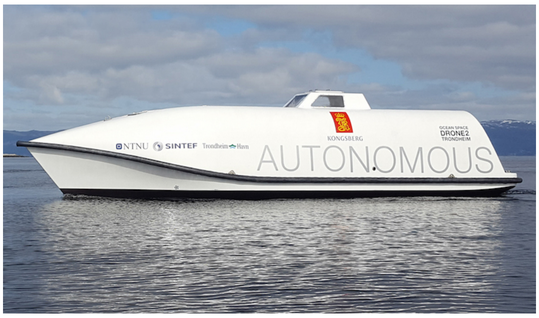


Figure 9: The Kongsberg Seatex Ocean Space Drone 2, which is identical to the Ocean Space Drone 1 (OSD1). Courtesy of Kongsberg Seatex.

Table 1: Telemetron ASV specifications.

Component	Description
Vessel hull	Polarcirkel Sport 845
Length	8.45 m
Width	2.71 m
Weight	1675 kg
Propulsion system	Yamaha 225 HP outboard engine
Motor control	Electro-mechanical actuation of throttle valve
Rudder control	Hydraulic actuation of outboard engine angle with proportional-derivative (PD) feedback control
Navigation system	Kongsberg Seatex Seapath 330+
Radar	Simrad Broadband 4G™ Radar
Processing platform	Intel® i7 3.4 GHz CPU, running Ubuntu 16.04 Linux

Table 2: BC-MPC algorithm parameters.

Parameter	Value	Description
B	3	Trajectory prediction tree depth
$\boldsymbol{T}$	[20, 30, 30] s	Step time lengths
$\boldsymbol{N}_U$	[5, 1, 1]	Number of SOG maneuvers
$\boldsymbol{N}_\chi$	[5, 3, 3]	Number of course maneuvers
T_{ramp}	1 s	Ramp time
T_U	5 s	SOG maneuver length
T_χ	5 s	Course maneuver length
w_{al}	1.5	Align weight
$w_{\text{av,m}}$	6000	Moving obstacle avoid weight
$w_{\text{av,s}}$	30	Static obstacle avoid weight
$w_{\text{t},U}$	2100	SOG transitional cost weight
$w_{\text{t},\chi}$	1050	Course transitional cost weight
a_0	50 m	Collision region major axis
a_1	150 m	Safety region major axis
a_2	250 m	Margin region major axis
b_0	25 m	Collision region minor axis
b_1	75 m	Safety region minor axis
b_2	125 m	Margin region minor axis
d_{COLREGs}	100 m	COLREGs expansion

algorithm at this rate. Furthermore, the sample time of the radar is 2.5 s, which together with the dynamics of the tracking system algorithms results in the closed-loop time delay being dominated by the obstacle detection and tracking system. With the given tuning parameters, the BC-MPC algorithm has a runtime of approximately 0.4 s (including interfacing the radar tracking system), allowing for a higher rate if sensors providing faster updates are available. The tuning parameters are quite similar to the ones used in the original algorithm, with the exception of the first step time length, which is selected as 20 s instead of 5 s in Eriksen et al. (2019b). With this tuning, the algorithm plans for making one maneuver of 5 s at the current time and keeping a constant course until 20 s have passed, rather than planning to do a new maneuver after only 5 s. This represents a more "maritime" way to maneuver compared to performing rapid consecutive maneuvers, and the transitional cost terms will motivate the algorithm to keep a constant course rather than selecting a new planned maneuver. Notice, however, that the algorithm is still free to choose a new maneuver every 5 s, but the transitional cost terms will favor keeping constant speed and course. To avoid that the vessel controller limited the performance of the COLAV system, we used a model-based speed and course controller shown to have high performance for high-speed ASVs (Eriksen and Breivik, 2018).

During the experiments, we tested four different scenarios:

1. A static-only scenario with two static obstacles.
2. A head-on situation with the OSD1 and four static obstacles.
3. A crossing situation with the OSD1 and one static obstacle.
4. An overtaking situation with the OSD1 and one static obstacle.

The desired speed of the Telemetron ASV was 5 m/s in all the scenarios, except the overtaking scenario where the desired speed was 8 m/s.

3.2 Scenario 1

Scenario 1 is shown in Figure 10. Here, two static obsta-

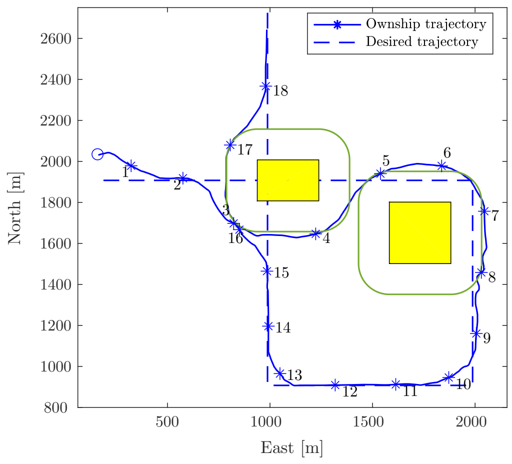


Figure 10: Scenario 1: Static-only scenario. The desired trajectory intersects with two obstacles, which the ownship successfully avoids. The blue circle denotes the initial position, while the text and asterisks mark each 60 s of the experiment. The yellow patches show the static obstacles, while the dark green contour lines show the padding regions.

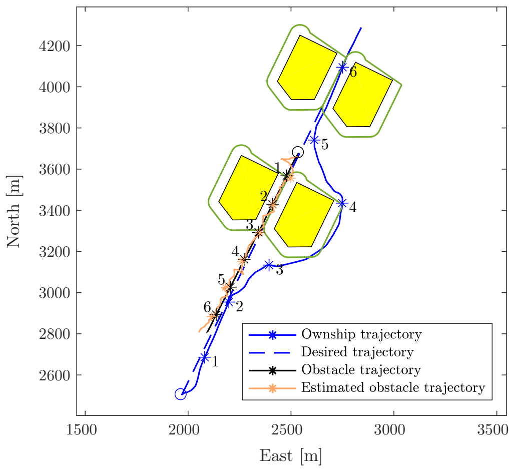


Figure 11: Scenario 2: Head-on situation. The desired trajectory passes through a narrow channel, which is blocked by the OSD1. The circles denote the initial positions, while the text and asterisks mark each 60 s of the experiment. The yellow patches show the static obstacles, while the dark green contour lines show the padding regions.

cles block the desired trajectory, requiring the BC-MPC algorithm to circumvent the obstacles. This scenario may seem a bit unrealistic, since the high-level planner and mid-level COLAV algorithm should plan paths which avoid static obstacles. However, the BC-MPC algorithm must be able to avoid static obstacles in order to stay safe in situations where we deviate from the desired trajectory, e.g. when avoiding moving obstacles or in situations where the mid-level algorithm is unable to produce a solution. The ownship converges to the desired trajectory before avoiding the first static obstacle by maneuvering to starboard. It would probably have been better to maneuver to port, since this would avoid having to pass through the narrow channel between the first and the second obstacle. The BC-MPC algorithm does, however, have a limited planning horizon of 80 s with the current tuning parameters, which makes it unaware of the narrow channel when making the decision of maneuvering to starboard. Subsequently, the ownship converges towards the desired trajectory and passes the second obstacle by having a small distance to the desired trajectory, which resides slightly inside the padding region of the static obstacle. After passing the second obstacle, the ownship converges to the desired trajectory, before avoiding the first obstacle once again.

3.3 Scenario 2

Scenario 2 is a head-on situation where the desired trajectory goes through a narrow channel composed by two static obstacles, and the channel entry is blocked by the OSD1. In this scenario, the padding distance was selected as 50 m in order to create the narrow channel between the obstacles. As shown in Figure 11, the ownship avoids the OSD1 by maneuvering to starboard and hence complying with the COLREGs. Following this turn, the first static obstacle is passed on the east side. The ownship returns to the desired trajectory and travels through the channel composed by the two last static obstacles.

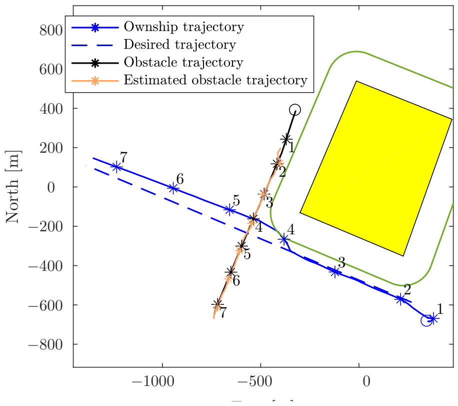


Figure 12: Scenario 3: Crossing situation. The desired trajectory intersects with the OSD1, which approaches from starboard. The static obstacle encloses Munkholmen, which is a small island located in the Trondheimsfjord. The circles denote the initial positions, while the text and asterisks mark each 60 s of the experiment. The yellow patch shows the static obstacle, while the dark green contour line shows the padding region.

3.4 Scenario 3

Scenario 3, shown in Figure 12, is a crossing situation where the OSD1 approaches from the ownship's starboard side, requiring the ownship to give way to avoid collision according to the COLREGs. In addition, there is a static obstacle on the starboard side of the ownship, blocking the ownship from maneuvering to starboard early. In compliance with the COLREGs, the ownship performs a starboard maneuver in order to pass behind the OSD1, while passing close to the boundary of the static obstacle. When the OSD1 has been passed, the ownship slowly converges towards the desired trajectory. The reason for the slow convergence is that the cost that the transitional cost terms introduces is just too large for the algorithm to change to a trajectory with a faster convergence. This is sometimes observed, but does not compromise safety and is a subject of tuning the transitional cost weights $w_{t,U}$ and $w_{t,\chi}$.

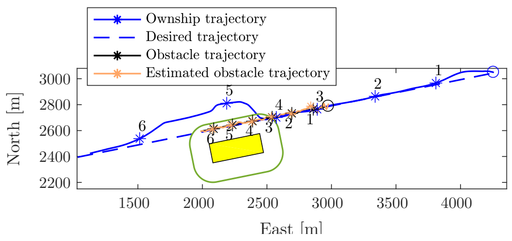


Figure 13: Scenario 4: Overtaking situation. The ownship overtakes the OSD1 by passing on the starboard side, while avoiding the static obstacle. The circles denote the initial positions, while the text and asterisks mark each 60 s of the experiment. The yellow patch shows the static obstacle, while the dark green contour line shows the padding region.

3.5 Scenario 4

Scenario 4 is an overtaking situation where the ownship approaches the OSD1 from behind. To allow the vessel being overtaken to maneuver to starboard if it finds itself in a separate collision situation, the BC-MPC algorithm is designed to favor a port turn in overtaking situations. However, as shown in Figure 13, a static obstacle is blocking the port side of the obstacle, which makes the ownship pass the obstacle on its starboard side. As mentioned, the BC-MPC algorithm is designed to pass with a larger clearance if passing on the port side rather than the starboard side, which can be seen by comparing this scenario with Experiment 3 in (Eriksen et al., 2019b).

3.6 Experiment summary

The BC-MPC algorithm is able to avoid collisions in all the scenarios, while converging to the desired trajectory when it is not obstructed by obstacles. The resulting ownship trajectories are clear and generally show the intension of the BC-MPC algorithm. The ownship trajectories are, however, a bit wobbly when the algorithm traverses alongside static obstacles. The reason for this is that the trajectory search space consists of a finite number of trajectories, of which none may traverse exactly parallel to the static obstacle. This results in that the algorithm sometimes choose to "zig-zag" along static obstacles, as seen in Sce-

Table 3: Minimum distance to obstacles. *The padding distance in Scenario 2 is 50 m.

Scenario number	Minimum distance to static obstacles	Minimum distance to moving obstacle
Scenario 1	130.4 m	–
Scenario 2	31.3 m*	167.1 m
Scenario 3	148.6 m	76.1 m
Scenario 4	115.9 m	145.3 m

nario 1. In the usual case where the mid-level algorithm would recompute a collision-free trajectory circumventing the obstacles, the BC-MPC algorithm would however be able to traverse smoothly along the obstacles by following the desired trajectory. Also, due to algae growth on the hull, the vessel dynamics had changed quite a bit since the model-based vessel controller was tuned, which also contributed to wobbling in the form of course overshoots.

As seen in Table 3, the ownship travels inside the padding region of the static obstacles. This is to be expected, since the objective function is only sensitive to the static obstacles when the trajectory resides inside of the padding region. Hence, the padding region and static avoidance weight $w_{\text{av,s}}$ should be selected such that a sufficient safety margin is achieved. A formulation with multiple regions with different gradients, as for moving obstacles, could make it easier to tune the algorithm to obtain a desired safety margin to static obstacles. The required distance to the moving obstacle is a bit more complex to discuss, since the obstacle regions sizes depend on the relative bearing. The ownship does, however, stay outside of the safety region in the head-on and crossing scenarios (scenarios 2 and 3), while we slightly enter the safety region in the overtaking scenario (Scenario 4).

4 Conclusion and further work

In this article, we have presented two modifications to the BC-MPC algorithm for ASV COLAV. The first modification allows the algorithm to avoid static obstacles in the form of an occupancy grid. The second modification concerns improved transitional cost terms by introducing transitional cost in speed and course separately, motivating the algorithm to not change the course if the speed is changed and vice versa. In addition, the algorithm tuning has been changed in order to obtain more “maritime” maneuvers and better utilize the transitional cost terms. The modified BC-MPC algorithm is tested in full-scale experiments in the Trondheimsfjord in Norway. A moving obstacle is detected and tracked using a radar-based system, while virtual static obstacles are added in the COLAV system. Four different scenarios were tested in experiments, all of which provided good results.

In Eriksen et al. (2019a), the authors have used the BC-MPC algorithm described in this article in a hybrid architecture, demonstrating COLAV compliant with COLREGs rules 8 and 13–17 in simulations. In the future, we would like to perform an extensive simulation study of the BC-MPC algorithm, in order to analyze the algorithm’s performance in greater detail.

Acknowledgments

This work was supported by the Research Council of Norway through project number 244116 and the Centres of Excellence funding scheme with project number 223254. The authors would like to express great gratitude to Kongsberg Seatex and Maritime Robotics for providing high-grade navigation technology, the Telemetron ASV and the OSD1 at our disposal for the experiments.

References

Abdelaal, M. and Hahn, A. NMPC-based trajectory tracking and collision avoidance of unmanned surface vessels with rule-based COLREGs confinement. In *Proc. of the 2016 IEEE Conference on Systems, Process and Control (ICSPC)*. Melaka, Malaysia, pages 23–28, 2016. doi:10.1109/SPC.2016.7920697.

Benjamin, M. R., Leonard, J. J., Curcio, J. A., and Newman, P. M. A method for protocol-based collision avoidance between autonomous marine surface craft. *Journal of Field Robotics*, 2006. 23(5):333–346. doi:10.1002/rob.20121.

Bitar, G., Eriksen, B.-O. H., Lekkas, A. M., and Breivik, M. Energy-optimized hybrid collision avoidance for

ASVs. In *Proc. of the 17th IEEE European Control Conference (ECC)*. Naples, Italy, pages 2522–2529, 2019.

Casalino, G., Turetta, A., and Simetti, E. A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields. In *Proc. of the 2009 IEEE OCEANS-EUROPE Conference*. Bremen, Germany, 2009. doi:10.1109/oceanse.2009.5278104.

Chauvin, C. Human factors and maritime safety. *Journal of Navigation*, 2011. 64(4):625–632. doi:10.1017/S0373463311000142.

Cockcroft, A. N. and Lameijer, J. N. F. *A Guide to the Collision Avoidance Rules*. Elsevier, 2004.

Dalsnes, B. R., Hexeberg, S., Flåten, A. L., Eriksen, B.-O. H., and Brekke, E. F. The neighbor course distribution method with Gaussian mixture models for AIS-based vessel trajectory prediction. In *Proc. of the 21st IEEE International Conference on Information Fusion (FUSION)*. Cambridge, UK, pages 580–587, 2018. doi:10.23919/ICIF.2018.8455607.

Eriksen, B.-O. H., Bitar, G., Breivik, M., and Lekkas, A. M. Hybrid collision avoidance for ASVs compliant with COLREGs rules 8 and 13–17. 2019a. Submitted to Frontiers in Robotics and AI, preprint available at `https://arxiv.org/abs/1907.00198`.

Eriksen, B.-O. H. and Breivik, M. *Modeling, Identification and Control of High-Speed ASVs: Theory and Experiments*, pages 407–431. Springer International Publishing, 2017a. doi:10.1007/978-3-319-55372-6_19.

Eriksen, B.-O. H. and Breivik, M. MPC-based mid-level collision avoidance for ASVs using nonlinear programming. In *Proc. of the 1st IEEE Conference on Control Technology and Applications (CCTA)*. Mauna Lani, Hawai'i, USA, pages 766–772, 2017b. doi:10.1109/CCTA.2017.8062554.

Eriksen, B.-O. H. and Breivik, M. A model-based speed and course controller for high-speed ASVs. In *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*. Opatija, Croatia, pages 317–322, 2018. doi:10.1016/j.ifacol.2018.09.504.

Eriksen, B.-O. H., Breivik, M., Wilthil, E. F., Flåten, A. L., and Brekke, E. F. The branching-course MPC algorithm for maritime collision avoidance. 2019b. Accepted for publication in Journal of Field Robotics, preprint available at `https://arxiv.org/abs/1907.00039`.

Eriksen, B.-O. H., Wilthil, E. F., Flåten, A. L., Brekke, E. F., and Breivik, M. Radar-based maritime collision avoidance using dynamic window. In *Proc. of the 2018 IEEE Aerospace Conference*. Big Sky, Montana, USA, pages 1–9, 2018. doi:10.1109/AERO.2018.8396666.

Hagen, I. B., Kufoalor, D. K. M., Brekke, E. F., and Johansen, T. A. MPC-based collision avoidance strategy for existing marine vessel guidance systems. In *Proc. of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Australia, pages 7618–7623, 2018. doi:10.1109/ICRA.2018.8463182.

Harati-Mokhtari, A., Wall, A., Brooks, P., and Wang, J. Automatic identification system (AIS): Data reliability and human error implications. *Journal of Navigation*, 2007. 60(3):373–389. doi:10.1017/S0373463307004298.

Hexeberg, S., Flåten, A. L., Eriksen, B.-O. H., and Brekke, E. F. AIS-based vessel trajectory prediction. In *Proc. of the 20th IEEE International Conference on Information Fusion (FUSION)*. Xi'an, China, pages 1–8, 2017. doi:10.23919/ICIF.2017.8009762.

Kuwata, Y., Wolf, M. T., Zarzhitsky, D., and Huntsberger, T. L. Safe maritime autonomous navigation with COLREGS, using velocity obstacles. *IEEE Journal of Oceanic Engineering*, 2014. 39(1):110–119. doi:10.1109/joe.2013.2254214.

Levander, O. Autonomous ships on the high seas. *IEEE Spectrum*, 2017. 54(2):26–31. doi:10.1109/MSPEC.2017.7833502.

Loe, Ø. A. G. *Collision Avoidance for Unmanned Surface Vehicles*. Master's thesis, Norwegian University of Science and Technology (NTNU), 2008.

Paris, C. Norway takes lead in race to build autonomous cargo ships. https://www.wsj.com/articles/norway-takes-lead-in-race-to-build-autonomous-cargo-ships-1500721202, 2017. Accessed: 2019-05-22.

Schuster, M., Blaich, M., and Reuter, J. Collision avoidance for vessels using a low-cost radar sensor. In *Proc. of the 19th IFAC World Congress*. pages 9673–9678, 2014. doi:10.3182/20140824-6-ZA-1003.01872.

Švec, P., Shah, B. C., Bertaska, I. R., Alvarez, J., Sinisterra, A. J., von Ellenrieder, K., Dhanak, M., and Gupta, S. K. Dynamics-aware target following for an autonomous surface vehicle operating under COLREGs in civilian traffic. In *Proc. of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Tokyo, Japan, pages 3871–3878, 2013. doi:10.1109/IROS.2013.6696910.

Wilthil, E. F. *Maritime Target Tracking with Varying Sensor Performance*. Ph.D. thesis, Norwegian University of Science and Technology (NTNU), 2019. Under evaluation.

Wilthil, E. F., Flåten, A. L., and Brekke, E. F. *A Target Tracking System for ASV Collision Avoidance Based on the PDAF*, pages 269–288. Springer International Publishing, Cham, 2017. doi:10.1007/978-3-319-55372-6_13.

Paper I Hybrid collision avoidance for ASVs compliant with COLREGs rules 8 and 13–17

Preprint of **B.-O. H. Eriksen**, G. Bitar, M. Breivik, and A. M. Lekkas, "Hybrid collision avoidance for ASVs compliant with COLREGs rules 8 and 13–17", 2019, Submitted to Frontiers in Robotics and AI, available at https://arxiv.org/abs/1907.00198.

Hybrid Collision Avoidance for ASVs Compliant with COLREGs Rules 8 and 13–17

Bjørn-Olav H. Eriksen [*], Glenn Bitar, Morten Breivik and Anastasios M. Lekkas

Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), O. S. Bragstads Plass 2D, 7491 Trondheim, Norway

Correspondence*:
Bjørn-Olav H. Eriksen
bjorn-olav.h.eriksen@ieee.org

ABSTRACT

This paper presents a three-layered hybrid collision avoidance (COLAV) system for autonomous surface vehicles, compliant with rules 8 and 13–17 of the International Regulations for Preventing Collisions at Sea (COLREGs). The COLAV system consists of a high-level planner producing an energy-optimized trajectory, a model predictive control based mid-level COLAV algorithm considering moving obstacles and the COLREGs, and the branching-course model predictive control algorithm for short-term COLAV handling emergency situations in accordance with the COLREGs. Previously developed algorithms by the authors are used for the high-level planner and short-term COLAV, while we in this paper further develop the mid-level algorithm to make it comply with COLREGs rules 13–17. This includes developing a state machine for classifying obstacle vessels using a combination of the geometrical situation, the distance and time to the closest point of approach (CPA) and a new CPA-like measure. The performance of the hybrid COLAV system is tested through numerical simulations for three scenarios representing a range of different challenges, including multi-obstacle situations with multiple simultaneously active COLREGs rules, and also obstacles ignoring the COLREGs. The COLAV system avoids collision in all the scenarios, and follows the energy-optimized trajectory when the obstacles do not interfere with it.

Keywords: Hybrid collision avoidance, Autonomous surface vehicle (ASV), COLREGs, COLREGs compliant, Model predictive control (MPC), Energy-optimized control

1 INTRODUCTION

Motivated by the potential for reduced costs and increased safety, the maritime industry is rapidly moving towards autonomous operations. Following groundbreaking advances in the automotive industry, many sectors within the maritime industry are considering the benefits of autonomy, which includes more environmentally friendly operations. For instance, the agricultural chemical company *Yara* together with the maritime technology supplier *Kongsberg Maritime* are developing the electrical autonomous container vessel *Yara Birkeland*, which aims to replace 40 thousand yearly truck journeys in urban eastern Norway[1]. Another example is the world's first autonomous car ferry, *Falco*, developed by *Rolls-Royce* (recently bought by Kongsberg Maritime) and *Finferries*. In

[1] `https://www.wsj.com/articles/norway-takes-lead-in-race-to-build-autonomous-cargo-ships-1500721202`, Accessed 2019-05-22.

2018, *Falco* navigated autonomously between two ports in Finland[2]. Reports state that in excess of 75 % of maritime accidents are due to human errors (Chauvin, 2011; Levander, 2017), indicating that there is also a potential for increased safety in addition to the economical and environmental benefits.

An obvious prerequisite for autonomous ship operations is the development of robust and well-functioning collision avoidance (COLAV) systems. In addition to generating collision-free maneuvers, a COLAV system must adhere to the "rules of the road" of the oceans, i.e. the International Regulations for Preventing Collisions at Sea (COLREGs) (Cockcroft and Lameijer, 2004). These rules are written for human ship operators and include qualitative requirements on how to perform safe and readily observable maneuvers. Part B of the COLREGs concern steering and sailing, and includes the following rules that are the most relevant to a motion control system:

Rule 8	Requires maneuvers to be readily observable and to be done in ample time.
Rules 13–15	Describe the maneuvers to perform in cases of overtaking, head-on and crossing situations. Participants in crossing situations are defined by the terms *give-way* and *stand-on* vessels.
Rule 16	Requires that a give-way vessel must take early and substantial action to keep clear of the stand-on vessel.
Rule 17	Consists of two main parts. The first part requires a stand-on vessel to maintain its course and speed, while the second part allows/requires[3] a stand-on vessel to take action to avoid collision if the give-way vessel is not taking action.

Since the rules are written for humans, with few quantitative figures, a challenge for autonomous operation is to quantify them into behaviors that can be executed algorithmically. The focus of the work in this paper is to do that, and to design a hybrid COLAV system that performs motion planning and generates maneuvers in compliance to rules 8 and 13–17 of the COLREGs.

A number of COLAV approaches considering the COLREGs have been proposed in the past. This includes algorithms using simulation-based model predictive control (Hagen et al., 2018), velocity obstacles (Kuwata et al., 2014), rule-based repairing A* (Campbell et al., 2014) and interval programming (Benjamin et al., 2006). All these approaches are single-layer approaches, where one algorithm solves the complete COLAV problem.

Another approach to the COLAV problem is to use a hybrid architecture, where the task of planning an obstacle-free path or trajectory, complying with the COLREGs and ultimately performing safe maneuvers is divided into layers in a control hierarchy. The idea of hybrid architectures is to divide the subtasks of the COLAV problem into multiple algorithms, exploiting their complementary strengths. This also has the side effect of making it easier for human operators or supervisors to understand the system. Most single-layer algorithms use sample-based approaches that consider a finite number of discrete control inputs, as opposed to conventional gradient-based search algorithms. The reason for this is that many gradient-based algorithms are not sufficiently numerically robust, not allowing a COLAV system to solely rely on such an algorithm. This issue can be handled in hybrid architectures, constrained by the bottom-level algorithm being numerically robust and able

[2] `https://www.marinemec.com/news/view,rollsroyce-and-finferries-demonstrate-worlds-first-fully-autonomous-ferry_56102.htm`, Accessed 2019-04-11.

[3] Rule 17 *allows* the stand-on vessel to maneuver when it becomes apparent that the give-way vessel does maneuver to avoid collision. If the vessels are so close that the give-way vessel cannot avoid collision by itself, Rule 17 *requires* the stand-on vessel to maneuver.

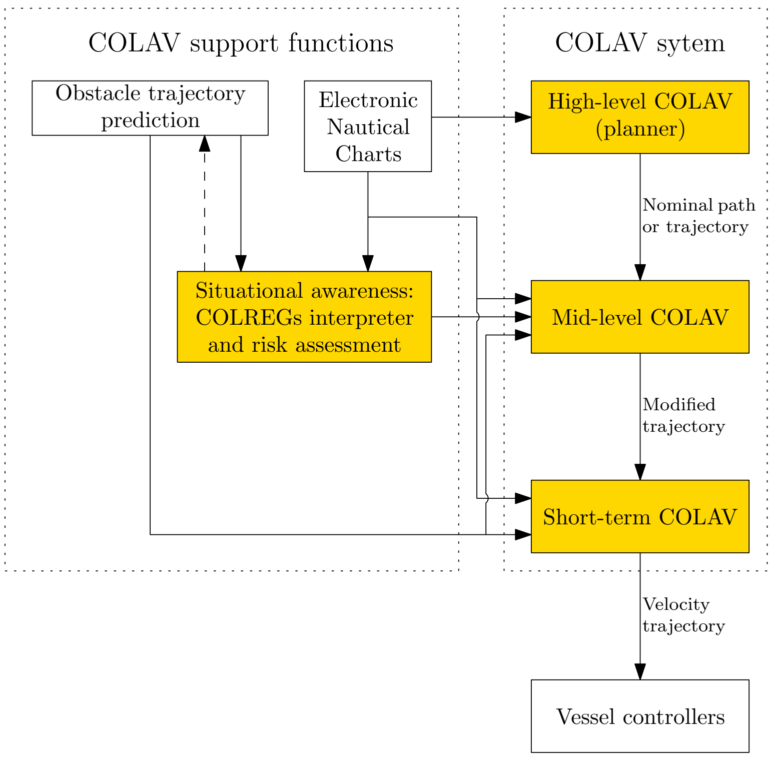

COLAV support functions
COLAV sytem
Obstacle trajectory prediction
Electronic Nautical Charts
High-level COLAV (planner)
Nominal path or trajectory
Situational awareness: COLREGs interpreter and risk assessment
Mid-level COLAV
Modified trajectory
Short-term COLAV
Velocity trajectory
Vessel controllers

Figure 1. Hybrid architecture with three COLAV layers, where the highlighted functions mark the areas of interest in this article. The COLAV system consists of a high-level planner, a mid-level COLAV algorithm and a short-term COLAV algorithm. The COLAV system is supported by data from electronic nautical charts, represented in a suitable manner for the algorithms, as well as situational awareness functions that track and predict obstacles, interpret the COLREGs and perform risk assessment.

to handle extraordinary situations where the other algorithms fail. Hence, hybrid architectures also allows using gradient-based algorithms, which are able to solve problems with large search spaces more efficiently than sample-based algorithms. The works by Švec et al. (2013) and Loe (2008) are examples of two-layered hybrid COLAV architectures. The top layers perform trajectory planning among static obstacles, while the bottom layers perform moving obstacle avoidance in compliance with COLREGs rules 13–16. Casalino et al. (2009) presents a three-layered hybrid COLAV system where the top layer also performs trajectory planning amongst static obstacles. The middle layer avoids moving obstacles, while the bottom layer implements safety functions for handling cases where the two other layers fail. This approach does, however, not consider the COLREGs.

Figure 1 shows a three-layered hybrid COLAV system for an autonomous surface vehicle (ASV). The authors have previously worked extensively on different components of this architecture. Examples include high-level COLAV algorithms (Bitar et al., 2018, 2019b), a mid-level algorithm (Eriksen and Breivik, 2017b; Bitar et al., 2019a), short-term algorithms (Eriksen et al., 2018, 2019; Eriksen and Breivik, 2019) and the development of high-performance vessel controllers (Eriksen and Breivik, 2017a, 2018).

In this paper, we demonstrate the three-layered hybrid COLAV shown in Figure 1 by combining and extending the COLAV algorithms developed in (Bitar et al., 2019a; Eriksen and Breivik, 2017b; Bitar et al., 2019b; Eriksen et al., 2019; Eriksen and Breivik, 2019). The high-level planner has a long temporal horizon, and finds an energy-optimized nominal trajectory from an initial to a goal position considering static obstacles. Since the high-level planner only considers static information, it is intended to be run offline, but it can also be run online, for instance if new static obstacles are detected. The mid-level algorithm attempts to follow this nominal trajectory, while performing COLAV of moving obstacles in compliance with COLREGs rules 8, 13–16 and the first part of Rule 17. The mid-level algorithm is run periodically with a shorter temporal horizon than the high-level algorithm, and produces a modified trajectory which is passed to the short-term layer. Both the high-level and mid-level algorithms use gradient-based optimization. The short-term algorithm attempts to follow the modified trajectory, while it in compliance with the second part of Rule 17 handles situations where obstacles ignore the COLREGs. This algorithm also handles other emergency situations, and uses sample-based optimization to achieve a high level of robustness, ensuring safe operation if the mid-level algorithm fails to find a solution. The following list summarizes our contributions:

- The high-level planner from (Bitar et al., 2019b) is modified to include the mathematical model of the *Telemetron* ASV in (Bitar et al., 2019a), including ocean currents.
- The development of a state-machine-based COLREGs interpretation scheme.
- The mid-level COLAV from (Bitar et al., 2019a) is modified to include rules 13–16 and the first part of Rule 17.
- The branching-course model predictive control (BC-MPC) algorithm for short-term COLAV is modified to reduce oscillatory behavior in turns.
- The three-layered COLAV system is verified in simulations and shown to be compliant with rules 8 and 13–17.

The rest of the paper has the following structure: The mathematical model of the ASV *Telemetron* is described in Section 2. The high-level planner, mid-level and short-term COLAV algorithms are described in sections 3 to 5, respectively. In Section 6 we present and discuss the simulation scenarios and results, and we conclude the paper in Section 7.

2 ASV MODELING

The vessel of interest in this article is the Telemetron ASV, which is owned and operated by the Norwegian company Maritime Robotics and shown in Figure 2. The Telemetron ASV is a high-speed dual-use vessel propelled by a steerable outboard engine, capable of speeds up to 18 m/s.

Eriksen and Breivik (2017a) presents a model of the Telemetron ASV, which is extended to include ocean currents in (Bitar et al., 2019a). The model has the form

$$\begin{aligned}\dot{\boldsymbol{\eta}} &= \boldsymbol{R}(\psi)\boldsymbol{x}_r + \begin{bmatrix}\boldsymbol{V}_c^\top & 0\end{bmatrix}^\top \\ \boldsymbol{M}(\boldsymbol{x}_r)\dot{\boldsymbol{x}}_r + \boldsymbol{\sigma}(\boldsymbol{x}_r) &= \boldsymbol{\tau}\,,\end{aligned} \tag{1}$$

where $\boldsymbol{\eta} = [x, y, \psi]^\top \in \mathbb{R}^2 \times S$ is the vessel pose and $\boldsymbol{V}_c = [V_x, V_y]^\top$ describes the ocean current, both in the Earth-fixed North-East-Down frame $\{n\}$. The vector $\boldsymbol{x}_r = [u_r, r]^\top \in \mathbb{X}_r \subset \mathbb{R}^2$ is the

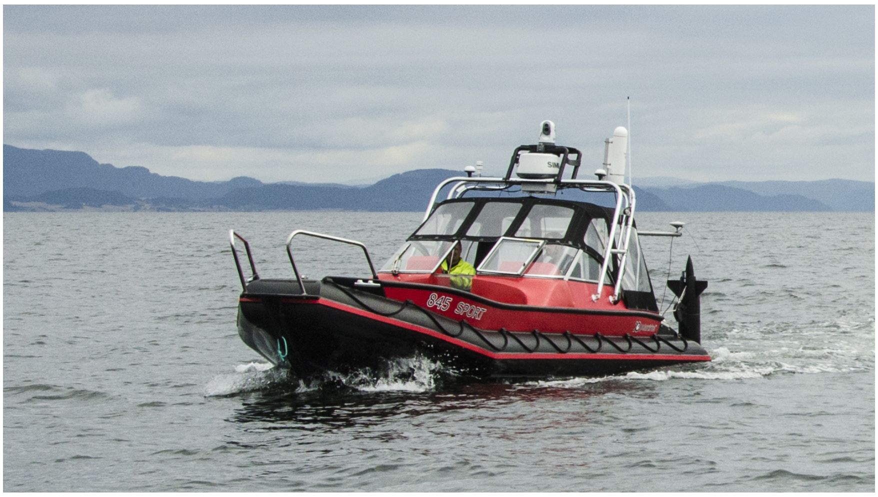


Figure 2. The Telemetron ASV, designed for both manned and unmanned operations. Courtesy of Maritime Robotics.

vessel velocity under the assumption of zero relative sway motion (Bitar et al., 2019a), where the set $\mathbb{X}_r$ describes the vessel-feasible steady-state velocities where (1) is valid. The transformation matrix $\boldsymbol{R}(\psi)$ is given by the heading $\psi \in S$ as

$$\boldsymbol{R}(\psi) = \begin{bmatrix} \cos\psi & 0 \\ \sin\psi & 0 \\ 0 & 1 \end{bmatrix} , \tag{2}$$

while $r \in \mathbb{R}$ describes the vessel yaw-rate. The matrix $\boldsymbol{M}(\boldsymbol{x}_r)$ is a state-dependent inertia matrix, while $\boldsymbol{\sigma}(\boldsymbol{x}_r)$ and $\boldsymbol{\tau} = [\tau_m, \tau_\delta]^\top \in \mathbb{U} \subset \mathbb{R}^2$ describe the vessel damping and control input, respectively. The set $\mathbb{U}$ describes the control inputs where (1) is valid.

3 HIGH-LEVEL PLANNER

To plan the ASV's nominal trajectory, we use a high-level trajectory planner developed in (Bitar et al., 2019b). This trajectory planner uses the ASV model described in Section 2 to generate an energy-optimized trajectory between the start and goal positions. The planning algorithm combines an A* implementation and an optimal control problem (OCP) solver to generate a feasible and optimized trajectory.

The high-level planning algorithm consists of three steps: First the A* implementation finds the shortest piecewise linear path between the start and goal position. Secondly, artificial temporal information is added to the path, converting it to a trajectory of states and inputs. Finally, the trajectory is used as an initial guess for an OCP solver, which finds a locally energy-optimized trajectory near the shortest path. All steps account for static obstacles in the form of elliptical boundaries.

3.1 Static obstacles

The elliptical boundaries are described with the inequality:

$$\left(\frac{x-x_c}{x_a}\right)^2 + \left(\frac{y-y_c}{y_a}\right)^2 \geq 1\,, \tag{3}$$

where x_c and y_c is the ellipsis center, and $x_a, y_a > 0$ are the ellipsis major and minor axes, respectively. To allow for angled obstacles, the ellipses are rotated clockwise by an angle α. We add a small constant $\epsilon > 0$ to each side of the inequality, and take the logarithm to arrive at the following obstacle representation:

$$\begin{aligned} h_o(x, y, x_c, y_c, x_a, y_a, \alpha) = -\log \Bigg[& \left(\frac{(x-x_c)\cos\alpha + (y-y_c)\sin\alpha}{x_a}\right)^2 \\ & + \left(\frac{-(x-x_c)\sin\alpha + (y-y_c)\cos\alpha}{y_a}\right)^2 + \epsilon\Bigg] + \log(1+\epsilon) \leq 0\,. \end{aligned} \tag{4}$$

The logarithm operation is applied to reduce the numerical range of the inequality, which helps with numerical stability of the subsequently described solver, and the constant ϵ is included to avoid singularities when (4) is evaluated for $(x, y) \to (x_c, y_c)$ (Bitar et al., 2019a).

3.2 Trajectory generation and optimization

From a scenario consisting of static obstacles, as mentioned in Section 3.1, we find the piecewise linear shortest path by performing an A* search on a uniformly decomposed grid. The resulting path is converted to a time-parametrized full-state trajectory by assuming a constant forward velocity, and connecting the shortest path with straight segments and circle arcs. The constant forward velocity is

$$u_{\text{nom}} = \frac{L_{\text{path}}}{t_{\text{max}}}\,, \tag{5}$$

where L_{path} is the length of the connected path, and t_{max} is the maximum allowed time to complete the trajectory. This full-state trajectory is then used as an initial guess to solve the OCP that gives the energy-optimized trajectory:

$$\min_{\boldsymbol{z}(\cdot),\boldsymbol{\tau}(\cdot)} \int_0^{t_{\text{max}}} F_{\text{hi}}(\boldsymbol{z}(t), \boldsymbol{\tau}(t))\mathrm{d}t \tag{6a}$$

subject to

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{f}(\boldsymbol{z}(t), \boldsymbol{\tau}(t))\ \forall t \in [0, t_{\text{max}}] \tag{6b}$$

$$\boldsymbol{h}_{\text{hi}}(\boldsymbol{z}(t), \boldsymbol{\tau}(t)) \leq \boldsymbol{0}\ \forall t \in [0, t_{\text{max}}] \tag{6c}$$

$$\boldsymbol{e}_{\text{hi}}(\boldsymbol{z}(0), \boldsymbol{z}(t_{\text{max}})) = \boldsymbol{0}\,. \tag{6d}$$

The solution of this OCP is a trajectory of states $\boldsymbol{z}(\cdot)$ and inputs $\boldsymbol{\tau}(\cdot)$ that minimizes the cost functional in (6a). The ASV model from Section 2 is rewritten as $\dot{\boldsymbol{z}} = \boldsymbol{f}(\boldsymbol{z}, \boldsymbol{\tau})$, where $\boldsymbol{z} = [\boldsymbol{\eta}^\top, \boldsymbol{x}_r^\top]^\top$ and $\boldsymbol{f}(\boldsymbol{z}, \boldsymbol{\tau})$ represents (1).

The cost functional (6a) is chosen to minimize energy. The cost-to-go function is

$$F_{\text{hi}}(\boldsymbol{z}, \boldsymbol{\tau}) = K_e F_e(\boldsymbol{z}, \boldsymbol{\tau}) + K_\delta \tau_\delta^2, \tag{7}$$

with tuning parameters $K_e, K_\delta > 0$. The first term consists of a function that is proportional to mechanical work performed by the ASV:

$$F_e(\boldsymbol{z}, \boldsymbol{\tau}) = |\underbrace{n(\tau_m)^2 \cdot \cos\delta(\tau_\delta)}_{\propto \text{ surge force}} \cdot u_r| + |\underbrace{n(\tau_m)^2 \cdot \sin\delta(\tau_\delta) \cdot L_m}_{\propto \text{ yaw moment}} \cdot r| \,. \tag{8}$$

The function $n : \mathbb{R}^+ \to \mathbb{R}^+$ maps the control input τ_m to propeller angular velocity. The function $\delta : \mathbb{R} \to \mathrm{S}$ maps the control input τ_δ to outboard motor angle. The second term in (8) is a quadratic cost to yaw control, included to avoid issues with singularity when solving the OCP.

The inequality constraints (6c) observe state boundaries as well as the static obstacles as represented in Section 3.1. The boundary conditions (6d) denote initial and final constraints, i.e. start and end states.

A detailed description of the transcription of the OCP (6) to a nonlinear program (NLP) using multiple shooting with N_{hi} shooting intervals is found in (Bitar et al., 2019b).

4 MID-LEVEL COLAV

The mid-level algorithm, initially presented in (Eriksen and Breivik, 2017b) and further developed in (Bitar et al., 2019a), is a model predictive control (MPC)-based algorithm intended for long-term COLAV. The algorithm utilizes gradient-based optimization, and takes both static and moving obstacles into account while attempting to follow an energy-optimized nominal trajectory from the high-level planner. The algorithm produces maneuvers complying with Rule 8 of the COLREGs, which requires maneuvers to be made in ample time and be readily observable for other vessels. The optimization problem is formulated as a NLP, which gives flexibility in designing the optimization problem.

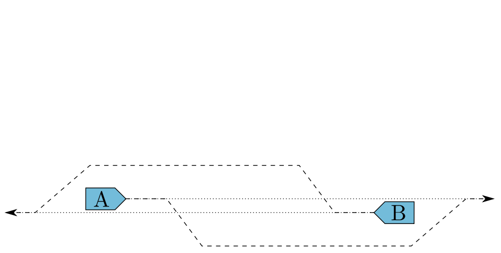
In this section, the algorithm is extended to also consider COLREGs rules 13–16 and the first part of Rule 17.

4.1 The International Regulations for Preventing Collisions at Sea (COLREGs)

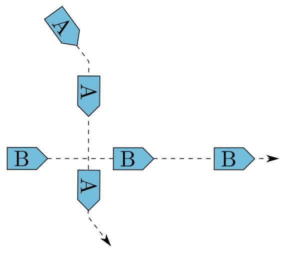
The COLREGs consists of a total of 37 rules and is divided into five parts (Cockcroft and Lameijer, 2004), where part B (rules 4–19) contains relevant rules on the conduct of vessels in proximity of each other. The most relevant rules for designing COLAV systems in part B are rules 8 and 13–17:

Rule 8 **Action to avoid collision.** This rule states that actions taken to avoid collision should be large enough to be readily observable of other ships, implying that series of small alternations in speed and/or course should not be applied. The rule also recommends that course changes should be prioritized over speed changes if there is enough free space available, and that maneuvers must be made in ample time.

Rule 13 **Overtaking.** This rule states that a vessel is overtaking another if it approaches the other vessel with a course more than 22.5° abaft her beam. The overtaking vessel has to



(A) Head-on situation.



(B) Crossing situation.

Figure 3. Illustration of head-on (A) and crossing (B) situations, and how they should be resolved.

stay clear of the overtaken vessel, but there is no statements on which side of the vessel one should pass.

Rule 14 Head on. When two power-driven vessels approach each other on reciprocal, or nearly reciprocal, courses, they are in a head-on situation. In such a situation, both vessels should change their course to starboard, passing each other port-to-port, as shown in Figure 3A. This rule states no explicit definition on what should be considered to be reciprocal, or nearly reciprocal, courses, but court decisions indicate head-on situations exist for opposing courses $\pm 6°$. Notice that the rule does not include sailing vessels, which are covered by Rule 12.

Rule 15 Crossing. When two vessels approach each other such that the situation is not a head on or an overtaking, it is a crossing situation. The vessel with the other one to her starboard side is deemed the give-way vessel, while the other vessel is deemed the stand-on vessel. As shown in Figure 3B, the give-way vessel should maneuver to avoid collision, preferably by passing behind the stand-on vessel, while the stand-on vessel should keep her speed and course.

Rule 16 Action by the give-way vessel. Every vessel which is required to keep out of the way of another vessel should take early and large enough action to safely avoid collision.

Rule 17 Action by the stand-on vessel. This rule requires that a stand-on vessel should keep its current speed and course. The stand-on vessel may, however, maneuver to avoid collision if it becomes apparent that the give-way vessel is not taking appropriate actions to avoid collision. Furthermore, if the stand-on vessel finds itself so close to the obstacle that collision can not be avoided by the give-way vessel alone, the stand-on vessel should take such action which best aids to avoid collision. In a crossing situation, the stand-on vessel should avoid maneuvering to port, since this could lead to a collision if the give-way vessel maneuvers to starboard.

In the hybrid architecture illustrated in Figure 1, the mid-level algorithm is given the task of strictly enforcing COLREGs rules 13–16 and the stand-on requirement of Rule 17, while also complying with Rule 8. In addition, we want the mid-level algorithm to comply with the first part of Rule 17, by not maneuvering to avoid collision in crossing situations if the ownship is the stand-on vessel. The COLAV system is inherently capable of adhering to the remaining requirement of Rule 17, where the stand-on vessel is allowed or required to maneuver, by having different prediction horizons and safety margins in the mid-level and short-term layers. The BC-MPC algorithm does not have any limitations of not maneuvering in stand-on situations, and will hence maneuver in stand-on situations if we come sufficiently close to the obstacle.

The mid-level algorithm as presented in (Bitar et al., 2019a) only complies with Rule 8. Further in this section, we therefore present improvements to the mid-level algorithm to make it comply with rules 13–16 and the stand-on requirement of Rule 17.

4.2 COLREGs interpretation

A commonly used concept for interpreting obstacles in COLAV algorithms is to use assign a spatial region to obstacles, which the ownship should not enter. This approach is commonly referred to as a *domain-based* approach. Specially designed ship domains are commonly used for interpreting the COLREGs in COLAV algorithms, where one usually require a larger clearance to obstacles if choosing maneuvers that violate the COLREGs (Eriksen et al., 2019; Szlapczynski and Szlapczynska, 2017). This approach is attractive since it continuously captures multiple COLREGs rules, and does not require logic or discrete decisions. However, such an approach does not strictly enforce the COLREGs rules, since it will allow maneuvers violating the rules if they are large enough. In addition, a ship-domain approach will not be able to strictly enforce the stand-on requirement of Rule 17, since a domain-based approach will avoid collision with all obstacles. One could ignore obstacles with give-way obligations, but this would require an explicit COLREGs interpretation which conflicts with domain-based approaches' core idea of implicit COLREGs interpretation. Therefore, we pursue an alternative approach to handling the COLREGs in the mid-level algorithm.

To simplify the COLREGs interpretation task, we look at the situation from a static perspective, assuming that the current COLREGs situations are valid throughout the entire prediction horizon of the mid-level algorithm. In reality, the COLREGs situations may, however, change during the prediction horizon depending on both the ownship's and obstacles future trajectory. For instance, an obstacle approaching from head on, but far enough away to not be considered as a danger may be put in a safe state. Hence, the mid-level algorithm will (for the current iteration) act like no COLREGs rule applies to this vessel for the entire prediction horizon, while the obstacle may get close enough during the prediction horizon to be considered as a head-on situation. An MPC scheme of only implementing a small part of the prediction horizon will reduce the implications of this, since the situation is reassessed each time mid-level algorithm is run, which justifies the assumption of considering the COLREGs from a static perspective. Investigating the possibilities for dynamically predicting future COLREGs situations as part of the MPC prediction will be considered as future work.

4.2.1 State machine

We propose to utilize a state machine in order to decide which COLREGs rule is active with respect to each obstacle in the vicinity of the ownship. The state machine contains the states:

SF Safe state. This implies that the COLREGs does not enforce any rule with respect to this obstacle.

OT Overtaking state. This implies that COLREGs Rule 13 applies with respect to this obstacle. The state machine does not discriminate on whether the ownship is overtaking another vessel or is being overtaken, but this can be done by looking at which vessel has the higher speed (Tam and Bucknall, 2010).

HO Head-on state. This implies that COLREGs Rule 14 applies with respect to this obstacle.

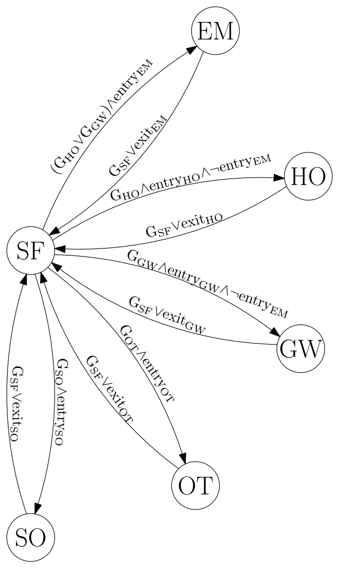

EM
HO
SF
GW
OT
SO
$(G_{HO} \vee G_{GW}) \wedge \text{entry}_{EM}$
$G_{SF} \vee \text{exit}_{EM}$
$G_{HO} \wedge \text{entry}_{HO} \wedge \neg\text{entry}_{EM}$
$G_{SF} \vee \text{exit}_{HO}$
$G_{GW} \wedge \text{entry}_{GW} \wedge \neg\text{entry}_{EM}$
$G_{SF} \vee \text{exit}_{GW}$
$G_{OT} \wedge \text{entry}_{OT}$
$G_{SF} \vee \text{exit}_{OT}$
$G_{SO} \wedge \text{entry}_{SO}$
$G_{SF} \vee \text{exit}_{SO}$

Figure 4. COLREGs state machine. The abbreviations "G_{SF}", "G_{SO}", "G_{OT}", "G_{GW}" and "G_{HO}" denote geometrical situations, while "$\mathrm{entry}_{\mathrm{xx}}$" and "$\mathrm{exit}_{\mathrm{xx}}$" denote additional state-dependent entry and exit criterias.

GW Give-way state. This implies that COLREGs Rule 15 applies with respect to this obstacle, and the ownship has to give way.

SO Stand-on state. This implies that COLREGs Rule 15 applies with respect to this obstacle, and the ownship has to stand on.

EM Emergency state. This implies that the obstacle is so close and/or behaves unpredictably, such that special considerations must be made.

As shown in Figure 4, all transitions have to go either from or to the safe state. This implies that when the state machine decides that a COLREGs (or emergency) situation exists with respect to an obstacle, it will not allow switching to another state without the situation being considered as safe first. One could argue that it should be able to transition between specific states, like e.g. from head-on, give-way and overtaking to emergency. This is an interesting topic, which should receive attention in the future. To control the transitions between the different states, we combine the time to and distance at the closest point of approach (CPA), a CPA-like measure of the time until a critical point and a geometrical interpretation of the situation.

4.2.2 Entry and exit criteria

CPA is a common concept in maritime risk assessment. Given the current speed and course of the ownship and an obstacle, CPA describes the time to the point where the two vessels are the closest, and the distance to the obstacle at this point. Given the position and velocity vector of the ownship $\boldsymbol{p}, \boldsymbol{v}$ and an obstacle $\boldsymbol{p}_o, \boldsymbol{v}_o$, the time to CPA is calculated as (Kufoalor et al., 2018)

$$t_{\mathrm{CPA}} = \begin{cases} 0 & \text{if } \|\boldsymbol{v} - \boldsymbol{v}_0\|_2 \leq \epsilon \\ \frac{(\boldsymbol{p}-\boldsymbol{p}_o)\cdot(\boldsymbol{v}-\boldsymbol{v}_o)}{\|\boldsymbol{v}-\boldsymbol{v}_o\|_2^2} & \text{else,} \end{cases} \tag{9}$$

where $\epsilon > 0$ is a small constant in order to avoid division by zero in the case where the relative velocity between the ownship and obstacle is zero. Given t_{CPA}, we calculate the distance between the vessels at CPA as

$$d_{\text{CPA}} = \|(\boldsymbol{p} + t_{\text{CPA}}\boldsymbol{v}) - (\boldsymbol{p}_o + t_{\text{CPA}}\boldsymbol{v}_o)\|_2. \tag{10}$$

While the CPA is the point where the distance to an obstacle is at its minimum, the critical point is where the distance to an obstacle crosses underneath a critical distance d_{crit}. This critical distance describes a minimum obstacle distance that the mid-level algorithm is designed for. The time to the critical point t_{crit} can be calculated by solving the equation

$$\|(\boldsymbol{p} + t_{\text{crit}}\boldsymbol{v}) - (\boldsymbol{p}_o + t_{\text{crit}}\boldsymbol{v}_o)\|_2 = d_{\text{crit}}\,. \tag{11}$$

In the cases where the distance between the ships does not fall below d_{crit}, t_{crit} is undefined. Otherwise, there are generally two solutions. The interesting solution is the one with the lowest t_{crit} value, as this is when the obstacle enters the d_{crit} boundary.

The state-machine entry criteria in Figure 4 are defined as

$$\begin{aligned}
\text{entry}_i &= \begin{cases} \mathit{true} & \text{if } d_{\text{CPA}} < \bar{d}_{\text{CPA}}^{i,\text{enter}} \wedge t_{\text{CPA}} \in [\underline{t}_{\text{CPA}}^{i,\text{enter}}, \bar{t}_{\text{CPA}}^{i,\text{enter}}] \\ \mathit{false} & \text{otherwise} \end{cases}, \quad \forall i \in \{\text{SO,OT,GW,HO}\} \\
\text{entry}_{\text{EM}} &= \begin{cases} \mathit{true} & \text{if } t_{\text{crit}} < \bar{t}_{\text{crit}}^{\text{EM,enter}} \wedge t_{\text{CPA}} > 0 \\ \mathit{false} & \text{otherwise,} \end{cases}
\end{aligned} \tag{12}$$

where $\bar{d}_{\text{CPA}}^{i,\text{enter}}$, $\underline{t}_{\text{CPA}}^{i,\text{enter}}$ and $\bar{t}_{\text{CPA}}^{i,\text{enter}}$ for $i \in \{\text{SO,OT,GW,HO}\}$ are tuning parameters denoting thresholds on d_{CPA} and t_{CPA} in order to satisfy the entry criteria for the stand-on, overtaking, give-way and head-on states. The tuning parameter $\bar{t}_{\text{crit}}^{\text{EM,enter}}$ denotes an upper limit on t_{crit} in order to enter the emergency state. The idea behind the stand-on, overtaking, give-way and head-on entry criterias are that in order for the obstacle to represent a risk, both t_{CPA} and d_{CPA} need to be within some tunable thresholds. Situations with a very low d_{CPA}, but with a high t_{CPA}, will not trigger the entry criteria, since the situations will not occur in the near future. Similarly, if t_{CPA} is within the thresholds, but d_{CPA} is large, this indicates a safe passing where risk of collision does not exist. The lower bound on t_{CPA} will typically be selected as zero, and is useful to distinguish between obstacles moving towards of away from the ownship. For the emergency state, the entry criteria is based on the critical point, at which we are so close that the mid-level algorithm may struggle with providing meaningful maneuvers. In addition to t_{crit} being under the threshold $\bar{t}_{\text{crit}}^{\text{EM,enter}}$, we require that t_{CPA} is positive, indicating that we are getting closer to the obstacle. Currently, we only allow entering the emergency state if the situation is a geometrical give-way or head-on, since an overtaking situation represents a smaller danger and has less requirement for special consideration.

The state-machine exit criterias in Figure 4 are defined as

$$\begin{aligned}
\text{exit}_i &= \begin{cases} \mathit{true} & \text{if } d_{\text{CPA}} \geq \underline{d}_{\text{CPA}}^{i,\text{exit}} \vee t_{\text{CPA}} \notin [\underline{t}_{\text{CPA}}^{i,\text{exit}}, \bar{t}_{\text{CPA}}^{i,\text{exit}}] \\ \mathit{false} & \text{otherwise} \end{cases}, \quad \forall i \in \{\text{SO,OT,GW,HO}\} \\
\text{exit}_{\text{EM}} &= \begin{cases} \mathit{true} & \text{if } t_{\text{crit}} \geq \underline{t}_{\text{crit}}^{\text{EM,exit}} \vee t_{\text{CPA}} \leq 0 \\ \mathit{false} & \text{otherwise,} \end{cases}
\end{aligned} \tag{13}$$

where $\underline{d}_{\text{CPA}}^{i,\text{exit}}, \underline{t}_{\text{CPA}}^{i,\text{exit}}$ and $\bar{t}_{\text{CPA}}^{i,\text{exit}}$ for $i \in \{\text{SO,OT,GW,HO}\}$ are tuning parameters denoting thresholds on d_{CPA} and t_{CPA} in order to satisfy the exit criteria for the stand-on, overtaking, give-way and head-on states. The exit criteria for the emergency state is satisfied if t_{crit} is larger than the tuning parameter $\bar{t}_{\text{exit}}^{\text{EM,enter}}$, or t_{CPA} is negative, implying that the obstacle is moving further away from the ownship. Note that the exit criterias are obtained by negating the entry criterias, but with other thresholds in order to implement hysteresis to avoid shattering. In general, we allow for different tuning parameters for the different states, but in our simulations we see that selecting the same tuning parameters for all states provides good results. Therefore, we define:

$$\begin{aligned} \bar{d}_{\text{CPA}}^{i,\text{enter}} &= \bar{d}_{\text{CPA}}^{\text{enter}} \\ \underline{t}_{\text{CPA}}^{i,\text{enter}} &= \underline{t}_{\text{CPA}}^{\text{enter}} \\ \bar{t}_{\text{CPA}}^{i,\text{enter}} &= \bar{t}_{\text{CPA}}^{\text{enter}}, \end{aligned} \tag{14}$$

and

$$\begin{aligned} \underline{d}_{\text{CPA}}^{i,\text{exit}} &= \underline{d}_{\text{CPA}}^{\text{exit}} \\ \underline{t}_{\text{CPA}}^{i,\text{exit}} &= \underline{t}_{\text{CPA}}^{\text{exit}} \\ \bar{t}_{\text{CPA}}^{i,\text{exit}} &= \bar{t}_{\text{CPA}}^{\text{exit}}. \end{aligned} \tag{15}$$

4.2.3 Geometrical situation interpretation

Tam and Bucknall (2010) present a geometrical interpretation scheme for deciding COLREGs situations based on the relative position, bearing and course of the obstacle with respect to the ownship. We base our geometrical interpretation on a slightly modified version of this scheme, where we include the sign of t_{CPA} to distinguish between situations where the obstacle moves closer towards or farther away from the ownship. The geometrical interpretation is shown in Figure 5, where the geometrical situation is obtained by finding which region the obstacle position and course resides in. Notice that the head-on region is larger than the threshold of $\pm 6°$ as described by the COLREGs. The reason for this is that Tam and Bucknall recommend using a larger region of 22.5° in order to increase the robustness of the geometrical COLREGs interpretation scheme.

4.3 Interface to the high-level planner

The high-level planner produces an energy-optimized nominal trajectory for the ownship to follow. However, since the high-level planner does not consider moving obstacles, the speed is the only time-relevant factor of the desired trajectory. In a case where the ownship for some reason, e.g. avoiding moving obstacles, lag behind the nominal trajectory, following the nominal trajectory in absolute time would cause a speed increase in order to catch up with it. Therefore, the mid-level algorithm performs *relative trajectory tracking*, where it tracks the nominal trajectory with a time offset $t_b \in \mathbb{R}$. This results in a relative nominal trajectory for the mid-level algorithm:

$$\bar{\boldsymbol{p}}_d(t) = \boldsymbol{p}_d(t + t_b), \tag{16}$$

where $\boldsymbol{p}_d = [N_d(t), E_d(t)]^\top$ is the nominal trajectory from the high-level planner. The time offset t_b is calculated each time the mid-level algorithm is run by solving a separate optimization problem, and is selected such that $\bar{\boldsymbol{p}}_d(t_0)$ is the point on the nominal trajectory closest to the ownship. See (Bitar et al., 2019a) for a detailed description of this concept.

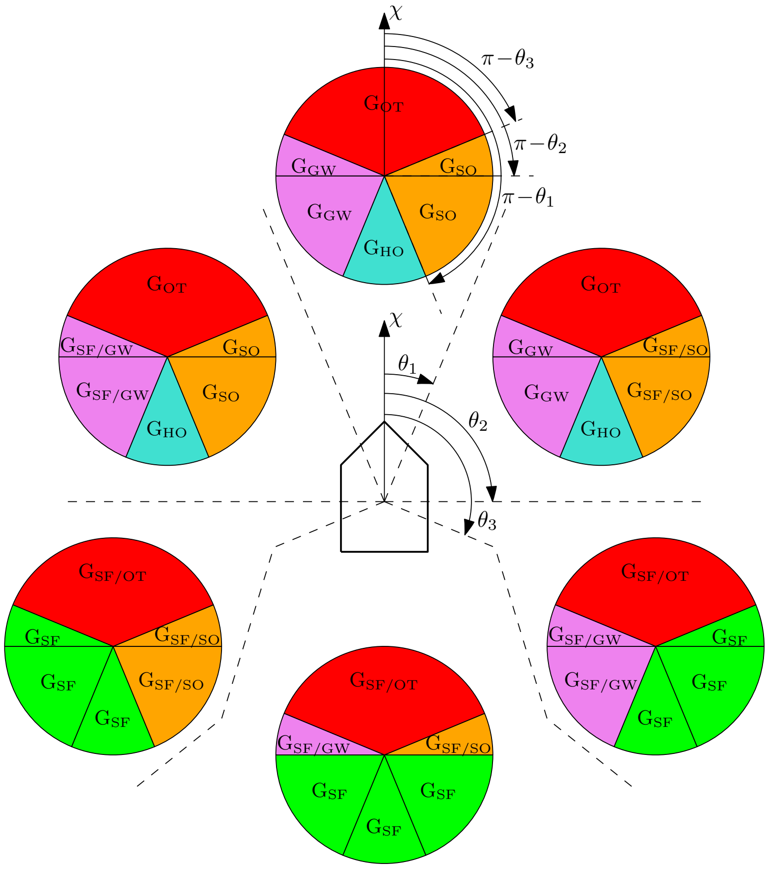


Figure 5. Illustration of the geometrical COLREGs interpretation, where the ownship course is denoted as χ and $\theta_1, \theta_2, \theta_3$ denote symmetrical regions given as $[22.5°, 90°, 112.5°]$. The circles illustrate obstacles in different relative bearing regions, and have a fixed orientation with respect to the ownship. The geometrical situations are color-coded and denoted as $G_i, i \in \{SF, SO, OT, GW, HO\}$ for safe, stand-on, overtaking, give-way and head-on situations, respectively. When two situations are given, like e.g. $G_{SF/SO}$, we use the former (SF) if $t_{CPA} < 0$ and the latter (SO) if $t_{CPA} \geq 0$, analogous to the obstacle moving away or towards the ownship. To decide the geometrical situation, we first find which relative bearing region the obstacle resides in, before finding which obstacle region the obstacle's course resides in. The figure is inspired by (Tam and Bucknall, 2010).

4.4 Optimization problem formulation

The mid-level algorithm is formalized as an OCP:

$$\min_{\boldsymbol{\eta}(\cdot), \boldsymbol{x}_r(\cdot)} \quad \phi(\boldsymbol{\eta}(\cdot), \boldsymbol{x}_r(\cdot)) \tag{17a}$$

subject to

$$\dot{\boldsymbol{\eta}}(t) = \boldsymbol{R}(\psi(t))\boldsymbol{x}_r(t) + \begin{bmatrix}\boldsymbol{V}_c \\ 0\end{bmatrix} \; \forall t \in [t_0, t_0 + T_h] \tag{17b}$$

$$\boldsymbol{h}_{\text{mid}}(\boldsymbol{\eta}(t), \boldsymbol{x}_r(t), t) \leq \boldsymbol{0} \; \forall t \in [t_0, t_0 + T_h] \tag{17c}$$

$$\boldsymbol{e}_{\text{mid}}(\boldsymbol{\eta}(t_0)) = \boldsymbol{0}\,, \tag{17d}$$

where $T_h > 0$ is the prediction horizon, $\phi(\cdot, \cdot)$ is the objective functional, (17b) contains a kinematic vessel model, (17c) contains inequality constraints and (17d) contains boundary constraints.

Analytical solutions of OCPs are in general not possible to find. A more common approach is to transcribe the OCP to an NLP, and solve that using a gradient optimization scheme. In our case, we transcribe (17) into an NLP with N_p samples using multiple shooting, where the vessel model is discretized using 4th order Runge Kutta and the cost functional is discretized using forward Euler. The resulting NLP is given as

$$\begin{aligned} &\min_{\boldsymbol{w},\boldsymbol{\omega},\boldsymbol{\mu},\boldsymbol{\xi}} \; \phi_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}) + \phi_c(\boldsymbol{w}) + \phi_{\text{COLREGs}}(\boldsymbol{w}) + \phi_\xi(\boldsymbol{\xi}) \\ &\text{subject to} \\ &\boldsymbol{g}(\boldsymbol{w}, \boldsymbol{\eta}(t_0)) = \boldsymbol{0} \\ &\boldsymbol{h}(\boldsymbol{w}, \boldsymbol{\xi}) \leq \boldsymbol{0} \\ &\bar{\boldsymbol{h}}_k(\boldsymbol{\eta}_k, \boldsymbol{\omega}_k, \boldsymbol{\mu}_k, \bar{\boldsymbol{p}}_{d,k}) \leq \boldsymbol{0} \; \forall k \in \{1, \dots, N_p\} \\ &\boldsymbol{\xi} \geq \boldsymbol{0}\,, \end{aligned} \tag{18}$$

where $\boldsymbol{w} = [\boldsymbol{\eta}_0^\top, \boldsymbol{x}_{r,0}^\top, \dots, \boldsymbol{\eta}_{N_p-1}^\top, \boldsymbol{x}_{r,N_p-1}^\top, \boldsymbol{\eta}_{N_p}^\top]^\top \in \mathbb{R}^{5N_p+3}$ is a vector of $5N_p + 3$ decision variables and $\bar{\boldsymbol{p}}_{d,1:N_p} = [\bar{\boldsymbol{p}}_{d,1}, \bar{\boldsymbol{p}}_{d,2}, \dots, \bar{\boldsymbol{p}}_{d,N_p}]$ is a sequence of desired positions. The vectors $\boldsymbol{\omega} \in \mathbb{R}^{2N_p}$, $\boldsymbol{\mu} \in \mathbb{R}^{2N_p}$ and $\boldsymbol{\xi} \in \mathbb{R}^{MN_p}$ contain slack variables, where M is the number of moving obstacles to be included in the constraints.

The vector $\boldsymbol{g}(\boldsymbol{w}, \boldsymbol{\eta}(t_0)) \in \mathbb{R}^{3N_p+3}$ contains shooting and boundary constraints, while $\boldsymbol{h}(\boldsymbol{w}) \in \mathbb{R}^{(M+D+4)N_p}$, where D is the number of static obstacles, contain inequality constraints ensuring COLAV and steady-state vessel velocity feasibility. The vectors $\bar{\boldsymbol{h}}_k(\boldsymbol{\eta}_k, \boldsymbol{\omega}_k, \boldsymbol{\mu}_k, \bar{\boldsymbol{p}}_{d,k}) \in \mathbb{R}^6$, $k \in \{1, N_p\}$ contain constraints on the slack variables $\boldsymbol{\omega}$ and $\boldsymbol{\mu}$.

In the following subsections, we describe the terms in (18) in more detail.

4.4.1 Objective function

The objective function contains four functions, where $\phi_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu})$ introduces cost on deviating from the relative nominal trajectory $\bar{\boldsymbol{p}}_d(t)$, $\phi_c(\boldsymbol{w})$ introduces cost on using control input, $\phi_{\text{COLREGs}}(\boldsymbol{w})$ is a COLREGs-specific function and $\phi_\xi(\boldsymbol{\xi})$ introduces slack variable cost.

To avoid that the NLP changes behavior when moving away from the nominal trajectory, we wish to have linear growth in the position error function $\phi_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu})$. This is achieved by instead of using quadratic terms in the position error function, we use the Huber loss function which is quadratic around the origin and resembles the absolute value function above a given threshold $\sigma > 0$:

$$H(\rho) = \begin{cases} \frac{1}{2}\rho^2 & |\rho| \leq \sigma \\ \sigma(|\rho| - \frac{1}{2}\sigma) & |\rho| > \sigma\,. \end{cases} \tag{19}$$

The Huber loss function has a discontinuous gradient, making it slightly complicated to implement in gradient-based optimization problems. It can, however, be implemented in a continuous fashion by utilizing lifting, where slack variables are introduced to create a problem of a higher dimensionality which is easier to solve. Using this technique, $\bar{\phi}_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu})$ is defined as

$$\bar{\phi}_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}) = K_p \sum_{k=1}^{N_p} \sigma \mathbf{1}^\top \boldsymbol{\omega}_k + \frac{1}{2}\boldsymbol{\mu}_k^\top \boldsymbol{\mu}_k, \tag{20}$$

where $K_p > 0$ is a tuning parameter, and $\boldsymbol{\omega}_k \in \mathbb{R}^2$ and $\boldsymbol{\mu}_k \in \mathbb{R}^2$ are slack variables constrained by

$$\bar{\boldsymbol{h}}_k(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}, \bar{\boldsymbol{p}}_{d,k}) = \begin{bmatrix} \boldsymbol{v_k} + \boldsymbol{\mu_k} + \boldsymbol{p}_k - \bar{\boldsymbol{p}}_{d,k} \\ \boldsymbol{v_k} + \boldsymbol{\mu_k} - (\boldsymbol{p}_k - \bar{\boldsymbol{p}}_{d,k}) \\ -\boldsymbol{\omega}_k \end{bmatrix} \leq \mathbf{0} \quad \forall k \in \{1, \ldots, N_p\}, \tag{21}$$

where $\boldsymbol{p}_k$ is the predicted vessel position at time step k, i.e. $\boldsymbol{\eta}_k = [\boldsymbol{p}_k^\top, \psi_k]^\top$. See (Bitar et al., 2019a) for more details.

Rule 8 of the COLREGs requires that maneuvers are readily observable for other vessels, implying that speed and course changes should have a sufficiently large magnitude, and not be performed as a sequence of small changes. In order to enforce this in the optimization problem, the control cost function $\phi_c(\boldsymbol{w})$ introduces a nonlinear cost on the change in speed and course, which makes the algorithm favor readily observable maneuvers. The function is defined as

$$\phi_c(\boldsymbol{w}) = \sum_{k=0}^{N_p - 1} K_{\dot{U}} q_{\dot{U}}(\dot{U}_k) + K_{\dot{\chi}} q_{\dot{\chi}}(\dot{\chi}_k), \tag{22}$$

where $K_{\dot{U}}, K_{\dot{\chi}} > 0$ are tuning parameters, while $q_{\dot{U}}(\dot{U}_k)$ and $q_{\dot{\chi}}(\dot{\chi}_k)$ are the nonlinear cost functions. Notice that neither the speed over ground (SOG) U nor the course χ are elements of the search space, but they can be computed as $U = \sqrt{u^2 + v^2}$ and $\chi = \psi + \arcsin \frac{v}{U}$. Their derivatives are then calculated by finite differencing. See (Eriksen and Breivik, 2017a; Bitar et al., 2019a) for more details on the control cost function.

The $\phi_{\text{COLREGs}}(\boldsymbol{w})$ function introduces a COLREGs-specific cost with respect to obstacles based on the rule currently applicable as defined by the state machine. We hence tailor the NLP to the current situation. The function is defined as

$$\begin{aligned} \phi_{\text{COLREGs}}(\boldsymbol{w}) = \sum_{k=1}^{N_p} \Bigg[& \sum_{i \in \mathcal{O}_{\text{HO}}} K_{\text{HO}} V_{\text{HO},i,k}(\boldsymbol{p}_k) + \sum_{i \in \mathcal{O}_{\text{GW}}} K_{\text{GW}} V_{\text{GW},i,k}(\boldsymbol{p}_k) \\ & + \sum_{i \in \mathcal{O}_{\text{SO}}} K_{\text{SO}} V_{\text{SO},k}(\boldsymbol{w}) + \sum_{i \in \mathcal{O}_{\text{EM}}} K_{\text{EM}} V_{\text{EM},k}(\boldsymbol{w}) \Bigg], \end{aligned} \tag{23}$$

where $\mathcal{O}_{\mathrm{HO}}, \mathcal{O}_{\mathrm{GW}}, \mathcal{O}_{\mathrm{SO}}$ and $\mathcal{O}_{\mathrm{EM}}$ contain obstacles which are in the head-on, give-way, stand-on and emergency states, respectively, and $K_{\mathrm{HO}}, K_{\mathrm{GW}}, K_{\mathrm{SO}}, K_{\mathrm{EM}} > 0$ are tuning parameters. The functions $V_{\mathrm{HO},i,k}(\boldsymbol{p}_k), V_{\mathrm{GW},i,k}(\boldsymbol{p}_k), V_{\mathrm{SO},k}(\boldsymbol{w})$ and $V_{\mathrm{EM},k}(\boldsymbol{w})$ describe functions capturing head-on, give-way, stand-on and emergency behavior with respect to obstacle i, respectively. Notice that the head-on and give-way functions vary with both the obstacle number and time step number, which is due to the functions depending on the the given obstacles position and course at time step k.

For head-on situations, we define a potential function with a positive value on the obstacle's starboard side, and a negative value on its port side. When used in the objective function, this will favor trajectories passing a head-on obstacle on its port side, in compliance with Rule 14 of the COLREGs. In addition, the potential function has an attenuation term, reducing the impact of the function when far away from an obstacle:

$$V_{\mathrm{HO},i,k}(\boldsymbol{p}) = \frac{\tanh\left(\alpha_{x,\mathrm{HO}}(x_{0,\mathrm{HO}} - x^{\{i,k\}})\right)}{2} \tanh(\alpha_{y,\mathrm{HO}} y^{\{i,k\}}) \in (-1, 1), \tag{24}$$

where $\alpha_{x,\mathrm{HO}}, \alpha_{y,\mathrm{HO}} > 0$ are tuning parameters controlling the steepness of the head-on potential function and $\bar{x}_{0,\mathrm{HO}} > 0$ is a tuning parameter controlling the influence of the attenuating potential. The coordinate $(x^{\{i,k\}}, y^{\{i,k\}})$ is $\boldsymbol{p}$ given in obstacles i's course-fixed frame (in which the x-axis points along the obstacle's course) at time step k, computed as

$$\begin{bmatrix} x^{\{i,k\}} \\ y^{\{i,k\}} \end{bmatrix} = \boldsymbol{R}(\chi_{i,k})^{\top} \left(\boldsymbol{p} - \boldsymbol{p}_{o,k,i}\right), \tag{25}$$

where $\boldsymbol{p}_{o,k,i}$ and $\psi_{i,k}$ are the position and heading of obstacle i at time step k. The head-on potential function with parameters $\alpha_{x,\mathrm{HO}} = 1/500, \alpha_{y,\mathrm{HO}} = 1/400$ and $x_{0,\mathrm{HO}} = 1000$ m is shown in Figure 6A.

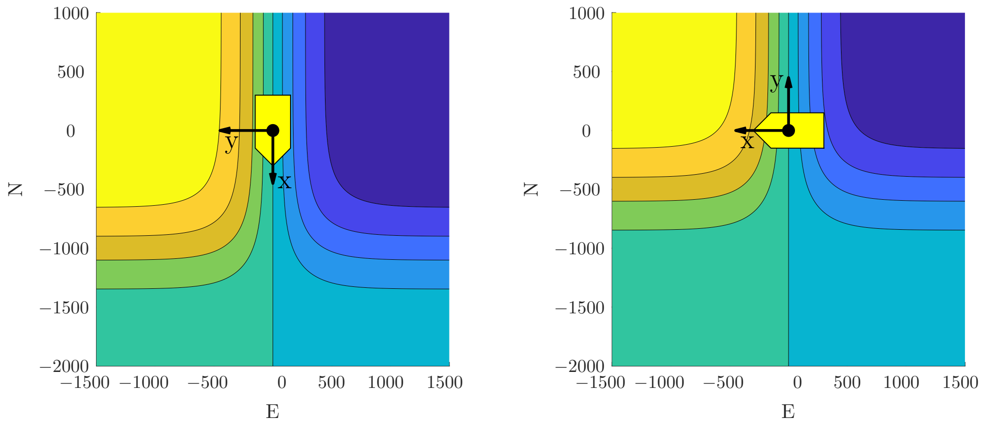
For give-way situations, we define a similar potential function, but rotated such that the function is positive in front of an obstacle and negative behind it. This will favor trajectories passing behind an obstacle, as desirable with respect to Rule 15 when a give-way obligation is active. The give-way potential function is defined as

$$V_{\mathrm{GW},i,k}(\boldsymbol{p}) = \frac{\tanh\left(\alpha_{y,\mathrm{GW}}(y^{i,k} - y_{0,\mathrm{GW}})\right)}{2} \tanh(\alpha_{x,\mathrm{GW}} x^{i,k}) \in (-1, 1), \tag{26}$$

where $\alpha_{x,\mathrm{GW}}, \alpha_{y,\mathrm{GW}} > 0$ control the steepness of the give-way potential function and $\bar{y}_{0,\mathrm{GW}} < 0$ control the attenuation on the port side of an obstacle. The give-way potential function with parameters $\alpha_{x,\mathrm{GW}} = 1/400, \alpha_{y,\mathrm{GW}} = 1/500$ and $y_{0,\mathrm{GW}} = -500$ m is shown in Figure 6B.

In stand-on situations, we want the mid-level algorithm to disregard the obstacle and keep the current speed and course in order to comply with the first part of Rule 17. One could simply constrain the algorithm to not maneuver, but this would be perilous in situations where the ownship simultaneously finds itself in a head-on or give-way situation. In such a situation it would be of extra importance to choose readily observable maneuvers, and we therefore design the stand-on cost with the same terms as used in the control cost (22) to amplify the effect:

$$V_{\mathrm{SO},k}(\boldsymbol{w}) = K_{\dot{U}} q_{\dot{U}}(\dot{U}_k) + K_{\dot{\chi}} q_{\dot{\chi}}(\dot{\chi}_k). \tag{27}$$



(A) Head-on potential function. **(B)** Give-way potential function.

Figure 6. Potential functions ensuring passing on the correct side in head-on and give-way situations. Yellow indicates a positive value, blue indicates a negative value, while the yellow patch and axis cross show the obstacle location and course-fixed coordinate system. Used in a minimization scheme, this will favor starboard maneuvers in head-on situations, and passing behind obstacles in give-way situations. Note that the obstacle here has zero sideslip, resulting in the heading and course pointing in the same direction.

If an obstacle is in an emergency state, the obstacle is disregarded in the mid-level algorithm and left for the short-term algorithm to handle. In such a situation, it is important that the mid-level algorithm behaves predictable, and we therefore use the same cost function as for stand-on situations:

$$V_{\mathrm{EM},k}(\boldsymbol{w}) = V_{\mathrm{SO},k}(\boldsymbol{w}). \tag{28}$$

The slack variable $\boldsymbol{\xi}$ is used in a homotopy scheme, which we introduce to avoid getting trapped in local minima around moving obstacles. The homotopy scheme is described in further detail in Section 4.5. The homotopy cost function $\phi_\xi(\boldsymbol{\xi})$ introduces slack cost on $\boldsymbol{\xi}$:

$$\phi_\xi(\boldsymbol{\xi}) = K_\xi \mathbf{1}^\top \boldsymbol{\xi}, \tag{29}$$

where $K_\xi > 0$ is iteratively increased as part of the homotopy scheme.

4.5 Obstacle handling and steady-state feasibility

The inequality constraint $\boldsymbol{h}(\boldsymbol{w}, \boldsymbol{\xi}) \leq \mathbf{0}$ ensures COLAV and steady-state feasibility with respect to actuator limitations.

Static obstacles are handled similarly as in the high-level algorithm, with (4) representing an elliptical obstacle with center (x_c, y_c), angle α and major and minor axes x_a and y_a, respectively. The constraint (4) needs to be enforced at each time step. Hence, for the i-th static obstacle, we define the constraint

$$\boldsymbol{h}_{s_i}(\boldsymbol{w}) = \begin{bmatrix} h_o(x_1, y_1, x_{c,i}, y_{c,i}, x_{a,i}, y_{a,i}, \alpha_i) \\ h_o(x_2, y_2, x_{c,i}, y_{c,i}, x_{a,i}, y_{a,i}, \alpha_i) \\ \vdots \\ h_o(x_{N_p}, y_{N_p}, x_{c,i}, y_{c,i}, x_{a,i}, y_{a,i}, \alpha_i) \end{bmatrix} \leq \boldsymbol{0}. \tag{30}$$

Moving obstacles are handled in a similar fashion, but letting the ellipsis center position and angle be time varying. Obstacles in stand-on situations should, however, not be included in the constraints, since the mid-level algorithm is supposed to stand on in such situations. Moreover, if an obstacle has entered an emergency state, the obstacle is so close and behaving unpredictably that the mid-level algorithm should disregard it and leave it for the short-term layer. Hence, for the i-th moving obstacle not in a stand-on or an emergency situation, we define the constraint

$$\boldsymbol{h}_{m_i}(\boldsymbol{w}) = \begin{bmatrix} h_o(x_1, y_1, x_{c,i,1}, y_{c,i,1}, x_{a,i}, y_{a,i}, \alpha_{i,1}) \\ \vdots \\ h_o(x_{N_p}, y_{N_p}, x_{c,i,N_p}, y_{c,i,N_p}, x_{a,i}, y_{a,i}, \alpha_{i,N_p}) \end{bmatrix} \leq \boldsymbol{0}, \tag{31}$$

where $x_{c,i,k}, y_{c,i,k}$ and $\alpha_{i,k}$ denote the position and course of the i-th moving obstacle at time step k.

Given D static obstacles and M obstacles not in stand-on or emergency situations, we define the constraint

$$\boldsymbol{h}_o(\boldsymbol{w}, \boldsymbol{\xi}) = \begin{bmatrix} \boldsymbol{h}_{s_1}(\boldsymbol{w}) \\ \vdots \\ \boldsymbol{h}_{s_D}(\boldsymbol{w}) \\ \boldsymbol{h}_{m_1}(\boldsymbol{w}) \\ \vdots \\ \boldsymbol{h}_{m_M}(\boldsymbol{w}) \end{bmatrix} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{\xi} \end{bmatrix}, \tag{32}$$

where we include slack variables $\boldsymbol{\xi} \geq \boldsymbol{0}$ on the moving obstacle constraints as part of a homotopy scheme. The reason for using homotopy is that NLP solvers in general only finds local minima, and can have issues with moving an initial guess "through" obstacles. Normally, this is not an issue, but for the mid-level algorithm the optimal solution can change drastically from one iteration to another. This can for instance happen if an obstacle enters a head-on or give-way state, where the solution can be trapped on the wrong side of an obstacle. In general, homotopy describes introducing an extra parameter which is iteratively adjusted in order to iteratively move a local solution towards a global solution (Deuflhard, 2011). In our homotopy scheme, we introduce slack variables on the moving obstacle constraints, which will allow solutions to travel through obstacles at the cost of a homotopy cost (29) scaled by the homotopy parameter K_ξ. Initially, this is selected as a low value to have a high amount of slack on the moving obstacles, while it is iteratively increased towards $K_\xi \to \infty$, which results in $\boldsymbol{\xi} = \boldsymbol{0}$ and hence no slack on moving obstacles. Currently, we only introduce slack on moving obstacles, but slack should also be introduced to static obstacles if they are small enough for the algorithm to be able to pass on both sides, like e.g. rocks, navigational marks, etc.

Similarly as in (Eriksen and Breivik, 2017b; Bitar et al., 2019a), we ensure steady-state feasible trajectories at each time step through a constraint $\boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,k}) \leq \boldsymbol{0} \in \mathbb{R}^4$, which captures the state constraint $\boldsymbol{x}_r \in \mathcal{X}_r$ at time step k. To ensure stead-state feasibility for the entire prediction horizon, we define the constraint

$$\boldsymbol{h}_{\boldsymbol{x}_r}(\boldsymbol{w}) = \begin{bmatrix} \boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,0}) \\ \boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,1}) \\ \vdots \\ \boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,N_p-1}) \end{bmatrix} \leq \boldsymbol{0}. \tag{33}$$

Finally, the inequality constraints are combined as

$$\boldsymbol{h}(\boldsymbol{w}, \boldsymbol{\xi}) = \begin{bmatrix} \boldsymbol{h}_o(\boldsymbol{w}, \boldsymbol{\xi}) \\ \boldsymbol{h}_{\boldsymbol{x}_r}(\boldsymbol{w}) \end{bmatrix} \in \mathbb{R}^{(M+D+4)N_p}. \tag{34}$$

5 SHORT-TERM COLAV

For the short-term layer, the branching-course model predictive control (BC-MPC) algorithm is used, which is a sample-based MPC algorithm intended for short-term ASV COLAV. The BC-MPC algorithm was initially developed in (Eriksen et al., 2019), extended to also consider static obstacles in (Eriksen and Breivik, 2019) and is experimentally validated in several full-scale experiments using a radar-based system for detecting and tracking obstacles. The algorithm complies with COLREGs rules 8, 13 and the second part of Rule 17, while favoring maneuvers complying with the maneuvering aspects of rules 14 and 15. Notice that Rule 17 allows a ship to ignore the maneuvering aspects of rules 14 and 15 in situations where the give-way vessel does not maneuver. The obstacle clearance will be larger if the algorithm ignores the maneuvering aspects of rules 14 and 15, like e.g. passing in front of an obstacle in a crossing situation where the ownship is the give-way vessel. Moving obstacles are in general handled by the mid-level algorithm, making this applicable only in emergency situations and for obstacles detected so late that the mid-level algorithm is unable to avoid them.

The algorithm constructs a search space consisting of a finite number of trajectories, which each contain a sequence of maneuvers. The maneuvers are constructed using a dynamic model of the ownship and a set of acceleration motion primitives, resulting in feasible trajectories being specified to the vessel controller. For each maneuver, a discrete set of SOG and course accelerations are created as

$$\begin{aligned} \dot{\boldsymbol{U}}_{\text{samples}} &= \left\{\dot{U}_1, \dot{U}_2, \dots, \dot{U}_{N_U}\right\} \\ \ddot{\boldsymbol{\chi}}_{\text{samples}} &= \left\{\ddot{\chi}_1, \ddot{\chi}_2, \dots, \ddot{\chi}_{N_\chi}\right\}, \end{aligned} \tag{35}$$

where $\dot{U}_i, i \in [1, N_U]$ and $\ddot{\chi}_i, i \in [1, N_\chi]$ denote $N_U \in \mathbb{N}$ and $N_\chi \in \mathbb{N}$ vessel-feasible speed and course accelerations. Given the acceleration samples (35) and motion primitives for each maneuver in a trajectory, we create a set of desired SOG and course trajectories $\mathcal{U}_d$. These trajectories have continuous acceleration, and is designed in an open-loop fashion by using the current reference tracked by the vessel controller for initialization, rather than the current vessel SOG and course. The reason for this is that the reference to the vessel controller should be continuous in order to avoid jumps in the actuator commands. To include feedback in the trajectory prediction, a set of feedback-corrected SOG and course trajectories $\bar{\mathcal{U}}_d$ is predicted using a simplified error model of the vessel and vessel controller. Finally, the feedback-corrected SOG and course trajectories are used to compute a set of feedback-corrected pose trajectories:

$$\bar{\mathcal{H}} = \left\{ \bar{\boldsymbol{\eta}}(\cdot) \middle| (\bar{U}(\cdot), \bar{\chi}(\cdot)) \in \bar{\mathcal{U}} \right\}, \tag{36}$$

where $\bar{\boldsymbol{\eta}}(\cdot)$ denotes a kinematic simulation procedure that given SOG and course trajectories, $\bar{U}(\cdot)$ and $\bar{\chi}(\cdot)$, in $\bar{\mathcal{U}}_d$ computes the vessel pose. See (Eriksen et al., 2019; Eriksen and Breivik, 2019) for more details on the trajectory generation procedure.

In order to converge towards the trajectory specified by the mid-level algorithm, a desired acceleration is computed based on a line-of-sight guidance scheme. In (Eriksen et al., 2019) and (Eriksen and Breivik, 2019), the samples closest to the desired acceleration in (35) are replaced with the desired acceleration, given that this is vessel-feasible. A problem with this, is that when operating at high speeds, the possible acceleration may not be symmetric, resulting in that zero acceleration (hence keeping a constant speed and course), may not be part of the search space. This can cause undesirable behavior, since the BC-MPC algorithm will be unable to keep the speed and course constant, which can cause oscillatory behavior. In this paper, we therefore propose to move the acceleration samples closest to zero, and adding the desired acceleration as a separate sample, given that it is vessel feasible. This will make sure that keeping a constant speed and course, as well as a trajectory converging towards the desired trajectory is included in the search space.

Given the predicted trajectories, the algorithm finds the optimal desired SOG and course trajectory for the vessel controller $\boldsymbol{u}_d^*(\cdot) = [U_d(\cdot)^*, \chi_d(\cdot)^*]$ as

$$\boldsymbol{u}_d^*(\cdot) = \underset{(\bar{\boldsymbol{\eta}}_k(\cdot), \boldsymbol{u}_{d,k}(\cdot)) \in (\bar{\mathcal{H}}, \mathcal{U}_d)}{\operatorname{argmin}} G(\bar{\boldsymbol{\eta}}_k(\cdot), \boldsymbol{u}_{d,k}(\cdot); \boldsymbol{p}_d^{\text{mid}}(\cdot)), \tag{37}$$

where the objective function is given as

$$\begin{aligned} G(\bar{\boldsymbol{\eta}}(\cdot), \boldsymbol{u}_d(\cdot); \boldsymbol{p}_d^{\text{mid}}(\cdot)) = {} & w_{\text{al}} \text{align}(\bar{\boldsymbol{\eta}}(\cdot); \boldsymbol{p}_d^{\text{mid}}(\cdot)) + w_{\text{av,m}} \text{avoid}_{\text{m}}(\bar{\boldsymbol{\eta}}(\cdot)) + w_{\text{av,s}} \text{avoid}_{\text{s}}(\bar{\boldsymbol{\eta}}(\cdot)) \\ & + w_{\text{t},U} \text{tran}_U(\boldsymbol{u}_d(\cdot)) + w_{\text{t},\chi} \text{tran}_\chi(\boldsymbol{u}_d(\cdot)). \end{aligned} \tag{38}$$

The variables $w_{\text{al}}, w_{\text{av,m}}, w_{\text{av,s}}, w_{\text{t},U}, w_{\text{t},\chi} > 0$ are tuning parameters, while $\text{align}(\bar{\boldsymbol{\eta}}(\cdot); \boldsymbol{p}_d^{\text{mid}}(\cdot))$ measures the alignment between a candidate trajectory $\bar{\boldsymbol{\eta}}(\cdot)$ and the desired trajectory from the mid-level algorithm $\boldsymbol{p}_d^{\text{mid}}(\cdot)$. The function $\text{avoid}_{\text{m}}(\bar{\boldsymbol{\eta}}(\cdot))$ ensures COLAV of moving obstacles by penalizing trajectories close to obstacles, using a non-symmetric obstacle ship domain designed with the COLREGs in mind. The function $\text{avoid}_{\text{s}}(\bar{\boldsymbol{\eta}}(\cdot))$ ensures COLAV of static obstacles by introducing an occupancy grid, while $\text{tran}_U(\boldsymbol{u}_d(\cdot))$ and $\text{tran}_\chi(\boldsymbol{u}_d(\cdot))$ introduces transitional costs to avoid shattering. The transitional terms penalize deviations from the planned trajectory of the previous iteration, unless changing to the trajectory corresponding by the desired acceleration. See (Eriksen et al., 2019) and (Eriksen and Breivik, 2019) for more details and descriptions of the terms.

6 SIMULATION RESULTS

The hybrid COLAV system is verified through simulations, which are present in this section. The simulations include ocean current and both static and moving obstacles. We include moving obstacles both acting in compliance with the COLREGs, and violating the COLREGs.

Table 1. Tuning parameters for the high-level algorithm.

Param.	Value	Comment
$t_{\max}$		Maximum trajectory time
Scenario 1	1420 s	
Scenario 2	1420 s	
Scenario 3	725 s	
N_{hi}	1000	Number of prediction steps
K_e	$1.0\,\text{s}^3/\text{m}$	Energy penalty gain
K_δ	1.0	Quadratic yaw control penalty gain
L_m	4.0 m	Length between control origin and outboard motor

The simulations are performed on a computer with an 2.8 GHz Intel Core i7 processor, running macOS Mojave.

6.1 Simulation setup

The simulations are performed in MATLAB using CasADi (Andersson et al., 2019) and IPOPT (Wächter and Biegler, 2005) for implementing the high-level and mid-level algorithms. The simulator is built upon the mathematical model of the Telemetron ASV described in Section 2, and the model-based speed and course controller in (Eriksen and Breivik, 2018) is used as the vessel controller.

The parameters of the high-level algorithm are listed in Table 1. The mid-level algorithm is implemented using the parameters in Table 2. The slack variable cost K_ξ has five elements, implying that we use five steps in our homotopy scheme. The mid-level NLP is initially warm started with the solution from the previous iteration, while each step in the homotopy scheme is warm started with the solution from the previous step of the homotopy scheme, converging towards the solution without slack on the constraints. To reduce the computational load and increase the predictability of the mid-level algorithm, we utilize six steps of each planned mid-level trajectory, only running the mid-level algorithm every 60 s. This implies that six steps of the predicted solution will be implemented before computing a new solution, which further implies that the state machine is also only run every 60 s. If the mid-level algorithm fails in finding a feasible solution, the algorithm will re-use the solution from the last iteration. This may for instance happen if the algorithm tries to compute a solution while being inside a moving obstacle ellipse, which sometimes can be the case when an obstacle is exiting an emergency or stand-on state. The BC-MPC algorithm is run every 5 s, with parameters as described in (Eriksen and Breivik, 2019). Static obstacles are padded with a safety margin of 150 m for the high-level and mid-level algorithms, while the BC-MPC algorithm uses a safety margin of 100 m for static obstacles. The reason for having a smaller static obstacle safety margin for the BC-MPC algorithm is that it tends to struggle with following trajectories on the static obstacle boundaries. The BC-MPC algorithm would hence not be able to follow the nominal trajectory if the static obstacle safety margin was the same as for the mid-level and high-level algorithms.

The simulations are performed without any noise on the obstacle estimates, providing the algorithms with exact information about the obstacles position, course and speed. The BC-MPC algorithm has previously been shown to perform well with noisy and uncertain obstacle estimates in full-scale experiments using radar-based detection and tracking of obstacles (Eriksen et al., 2019; Eriksen and Breivik, 2019). The mid-level algorithm is likely to have a larger requirement to low noise

Table 2. Tuning parameters for the mid-level algorithm.

Param.	Value	Comment
$\bar{d}^{\text{enter}}_{\text{CPA}}$	900 m	State machine d_{CPA} entry criteria
$\underline{d}^{\text{exit}}_{\text{CPA}}$	2000 m	State machine d_{CPA} exit criteria
$[\underline{t}^{\text{enter}}_{\text{CPA}}, \bar{t}^{\text{enter}}_{\text{CPA}}]$	$[0, 270]$ s	State machine t_{CPA} entry criteria
$[\underline{t}^{\text{exit}}_{\text{CPA}}, \bar{t}^{\text{exit}}_{\text{CPA}}]$	$[-20, 290]$ s	State machine t_{CPA} exit criteria
$\bar{t}^{\text{EM,enter}}_{\text{crit}}$	20 s	Emergency state t_{crit} entry criteria
$\underline{t}^{\text{EM,exit}}_{\text{crit}}$	25 s	Emergency state t_{crit} exit criteria
h	10 s	Step size
N_p	36	Number of prediction steps
K_p	0.02	Position error scaling
σ	1	Huber loss function threshold
$K_{\dot{U}}$	0.3	SOG-derivative penalty term scaling
$K_{\dot{\chi}}$	2.5	Course-derivative penalty term scaling
K_{HO}	40	Head-on potential function scaling
$[\alpha_{x,\text{HO}}, \alpha_{y,\text{HO}}]$	$[1/500, 1/400]$	Head-on potential function steepness parameters
$x_{0,\text{HO}}$	1000 m	Head-on potential function attenuation parameter
K_{GW}	40	Give-way potential function scaling
$[\alpha_{x,\text{GW}}, \alpha_{y,\text{GW}}]$	$[1/400, 1/500]$	Give-way potential function steepness parameters
$y_{0,\text{GW}}$	-500 m	Give-way potential function attenuation parameter
K_{SO}	3	Stand-on function scaling
K_{EM}	3	Emergency function scaling
K_{ξ}	$[0.1, 1, 10, 100, \infty]$	Iterative slack variable cost
x_a	600 m	Moving obstacle ellipsis major axis size
y_a	225 m	Moving obstacle ellipsis minor axis size

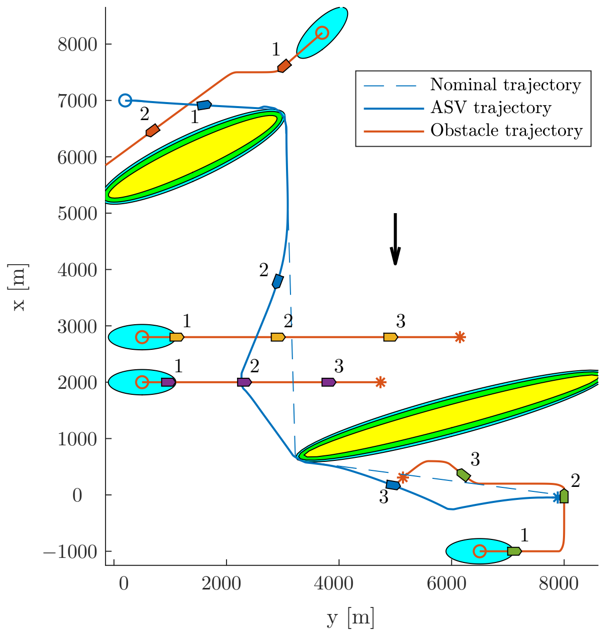
levels on the obstacle estimates, since the state machine in the mid-level algorithm depends on logic and discrete switching. However, the algorithm is also run less frequently, reducing the required bandwidth of the obstacle estimates, possibly allowing using smoothing or tracking filters with a lower process noise if necessary. It may also be feasible to make the mid-level algorithm depend on data from the automatic identification system, which typically have much lower noise levels than radar-based tracking systems, while being subject to robustness issues (Harati-Mokhtari et al., 2007).

We present three scenarios, which demonstrate different important properties of the hybrid COLAV system:

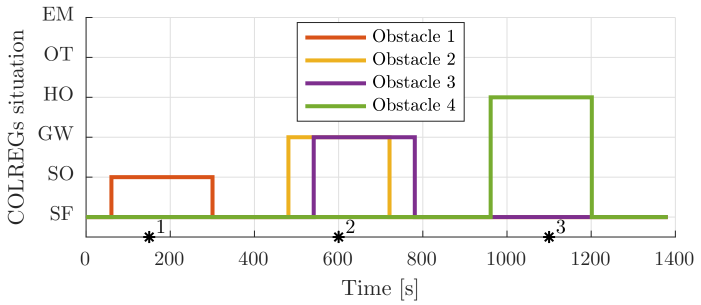
Scenario 1 This scenario contains two static obstacles, and four moving obstacles of which all comply with the COLREGs. The moving obstacles demonstrate stand-on, give-way and head-on situations.

Scenario 2 This scenario contains one static and five moving obstacles. The moving obstacles demonstrate stand-on with an obstacle ignoring the COLREGs, an overtaking and a simultaneous head-on, give-way and stand-on situation with obstacles complying with the COLREGs.

Scenario 3 This scenario contains two moving obstacles, which suddenly perform dangerous maneuvers close to the ownship, displaying the use of the emergency state.



(A) Trajectory plot. The initial position of the ownship and obstacles are shown with circles, with the blue ellipses illustrating the moving obstacle ellipse size. The vessel patches, which are overexaggerated for visualization, mark the ownship and obstacle poses at given time stamps. The static obstacles are shown in yellow, with the BC-MPC and mid-level safety margins enclosed around. The black arrow indicates the ocean current direction.

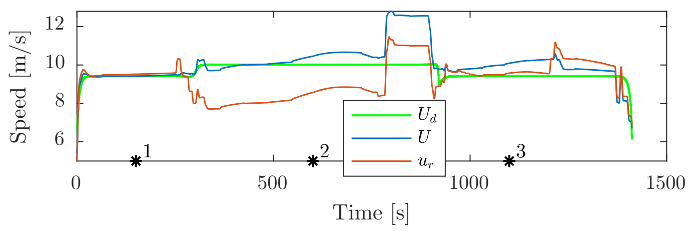


(B) Output from the state machine for each obstacle. The asterisks mark time stamps, and the colors correspond to the obstacle patch colors in the trajectory plot.

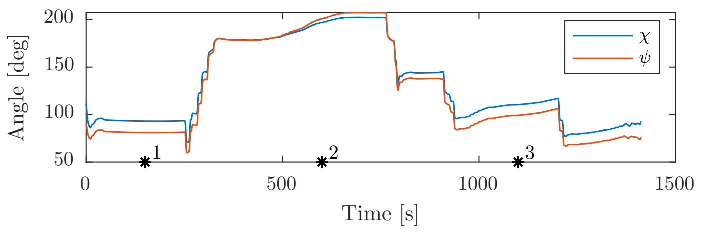
Figure 7. Scenario 1: Trajectory and COLREGs interpretation. The text marks denote the time steps $[150, 600, 1100]$ s.

6.2 Scenario 1

Scenario 1 contains two static obstacles, four moving obstacles, an ocean current of $[-2, 0]^\top$ m/s and is shown in Figure 7. The high-level planner plans a nominal trajectory between the initial and goal positions at $[7000, 200]^\top$ m and $[0, 7900]^\top$ m, respectively. The first obstacle is in a stand-on



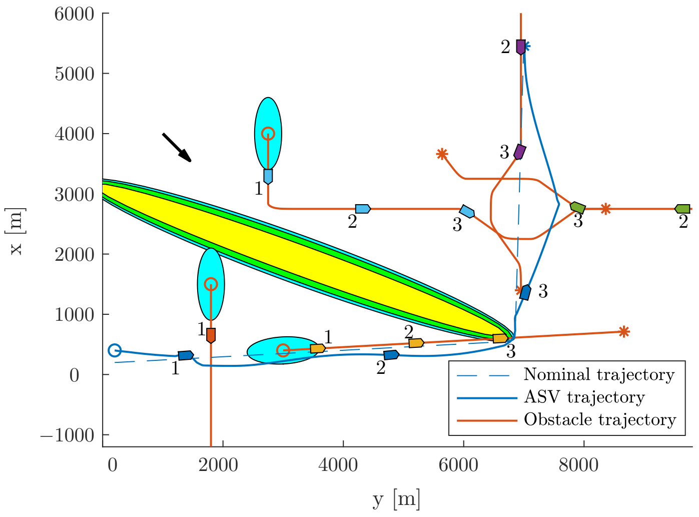
(A) Speed trajectories



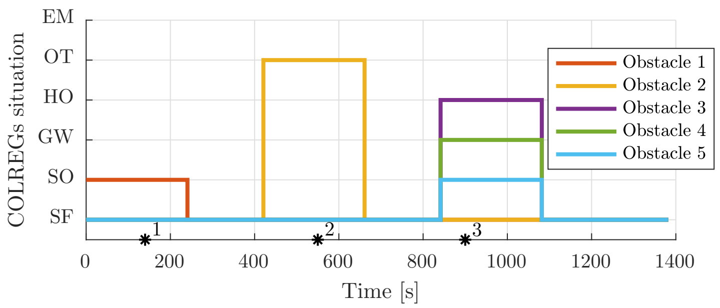
(B) Angular trajectories

Figure 8. Scenario 1: Speed and angular trajectories. The asterisks mark the same time samples as in Figure 7.

situation, where it is required to maneuver in order to avoid collision with the ownship, which is required to stand on. As shown in Figure 7B, the first obstacle is quickly considered as a stand-on situation, at which the mid-level algorithm disregards the obstacle and continues with the current speed and course. Following this, the obstacle maneuvers in accordance to the COLREGs, and we avoid collision. After the first static obstacle, we encounter two crossing vessels where the ownship is deemed the give-way vessel. In accordance with the COLREGs, we maneuver to starboard in order to pass behind both obstacles. Notice that the second give-way obstacle is detected as a give-way situation later than the first, since the entry criteria in the state machine includes the time to CPA, which is higher for the second give-way obstacle. After avoiding the two give-way obstacles, we converge towards the nominal trajectory and encounter a head-on situation. This is correctly identified by the state machine as head on, and we maneuver to starboard in order to avoid collision. Notice that even though the obstacle maneuvers, we keep the obstacle in the head-on state until we have passed it. Figure 8 shows the speed and angular trajectories during Scenario 1, where the desired speed is calculated as the nominal speed at the closest point on the nominal trajectory given the ownship position. From this, we see that the mid-level and BC-MPC algorithms manage to track the desired nominal speed before and after the first static obstacle, where no obstacles require maneuvering away from the nominal trajectory. Notice that when encountering the two crossing obstacles, the mid-level algorithm chooses to slowly change the course, which is due to the attenuation of the give-way potential function and the large distance between the vessels. It would be better to make a clear course change, which is a subject of tuning. After passing the two crossing obstacles, the mid-level algorithm increases the speed in order to get back to the nominal trajectory, which is due to the algorithm attempting to keep the speed projected on the nominal trajectory equal as the desired nominal speed. Furthermore, notice that the mid-level algorithm actively controls the relative surge speed in order achieve the desired SOG, which is clearly seen when passing the first static obstacle.



(A) Trajectory plot. The initial position of the ownship and obstacles are shown with circles, with the blue ellipses illustrating the moving obstacle ellipse size. The vessel patches, which are overexaggerated for visualization, mark the ownship and obstacle poses at given time stamps. The static obstacles are shown in yellow, with the BC-MPC and mid-level safety margins enclosed around. The black arrow indicates the ocean current direction.

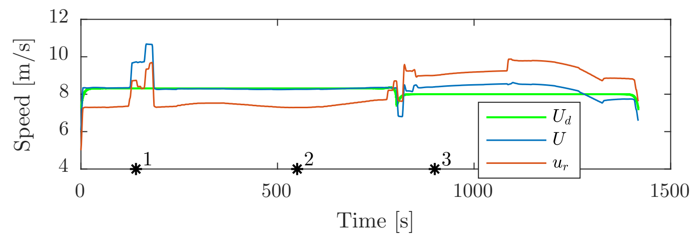


(B) Output from the state machine for each obstacle. The asterisks mark time stamps, and the colors correspond to the obstacle patch colors in the trajectory plot.

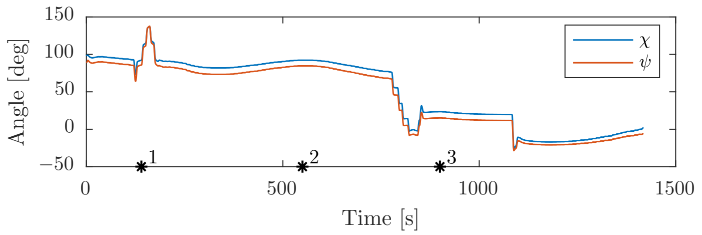
Figure 9. Scenario 2: Trajectory and COLREGs interpretation. The text marks denote the time steps $[140, 550, 900]$ s.

6.3 Scenario 2

Scenario 2, shown in Figure 9, is more complex than Scenario 1, with a total of five moving obstacles, and has an ocean current of $[-1, 1]^\top$ m/s. The high-level planner plans a nominal trajectory between the initial and goal positions at $[200, 200]^\top$ m and $[5500, 7000]^\top$ m, respectively. The first obstacle is a crossing vessel, which similarly as in Scenario 1 is deemed to give way for the ownship, which should keep the current speed and course. However, in this scenario, the obstacle violates the COLREGs by not maneuvering in order to avoid collision. Therefore, the BC-MPC algorithm maneuvers to avoid collision when the obstacle gets so close that the safety margins of the BC-MPC



(A) Speed trajectories



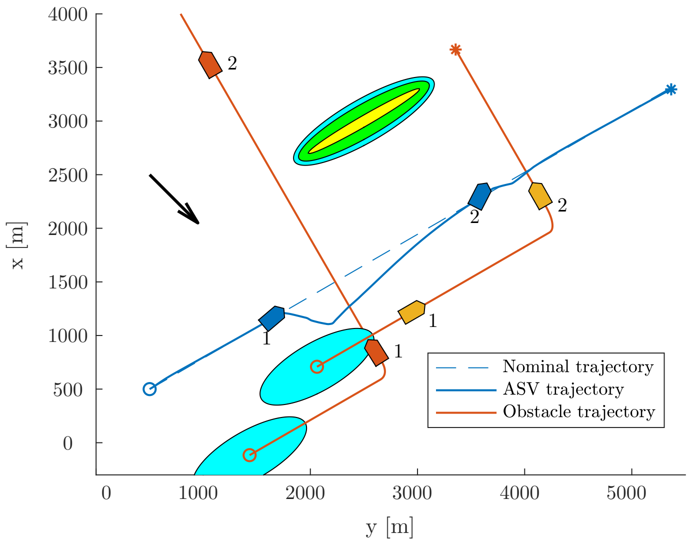
(B) Angular trajectories

Figure 10. Scenario 2: Speed and angular trajectories. The asterisks mark the same time samples as in Figure 9.

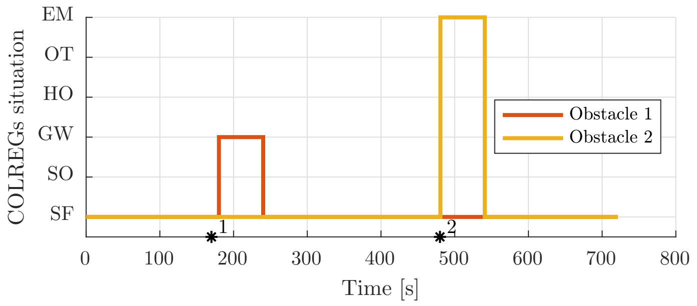
algorithms is violated. The BC-MPC algorithm maneuvers to port, as advised by COLREGs Rule 17 for crossing situations where the stand-on vessel has to maneuver, and safely avoid the first obstacle. The second obstacle is overtaken by the ownship, and correctly considered as an overtaking situation by the state machine. For such an situation, there is no requirement on how the ownship should maneuver, except keeping clear from the overtaken vessel. After passing the second obstacle, we encounter a complex situation with simultaneous head-on, give-way and stand-on obligations. In this situation, each vessel, including the ownship, finds itself in a situation where a head-on and a give-way situation require starboard maneuvers, while a stand-on situation requires the vessel to keep the current speed and course. However, head-on and give-way obligations should be prioritized higher than stand-on situations, and the situation is quite easily solved by each vessel maneuvering to starboard and passing behind the vessel crossing from starboard. The mid-level algorithm solves this situation with the desirable behavior, and converges towards the nominal trajectory after the situation is resolved. As shown in Figure 9B, the state machine interprets the situations correctly. From the speed trajectory in Figure 10 it is clear that the mid-level algorithm follows the desired nominal speed also when overtaking the second obstacle.

6.4 Scenario 3

Scenario 3, shown in Figure 11, contains two moving obstacles on parallel courses with the ownship, and has an ocean current of $[-1, 1]^\top$ m/s. The high-level planner plans a nominal trajectory between the initial and goal positions at $[500, 500]^\top$ m and $[3328, 5399]^\top$ m, respectively, which results in a straight line trajectory with a course angle of 60°. The first obstacle travels at a higher speed than the ownship, while the second one travels at a lower speed and will be overtaken by the ownship. Since the obstacles are on parallel paths with the obstacle, the time to CPA is sufficiently high such that the obstacles are in the safe state, even though the the vessels are quite close. However, both obstacles make sudden maneuvers to port dangerously close to the ownship and enters on a



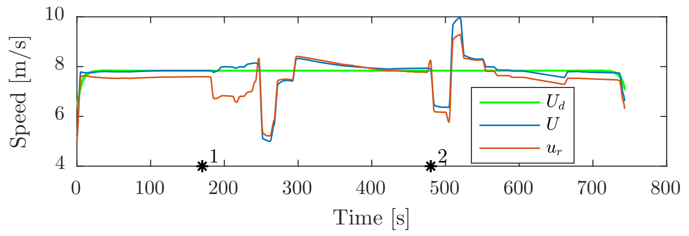
(A) Trajectory plot. The initial position of the ownship and obstacles are shown with circles, with the blue ellipses illustrating the moving obstacle ellipse size. The vessel patches, which are overexaggerated for visualization, mark the ownship and obstacle poses at given time stamps. The static obstacles are shown in yellow, with the BC-MPC and mid-level safety margins enclosed around. The black arrow indicates the ocean current direction.



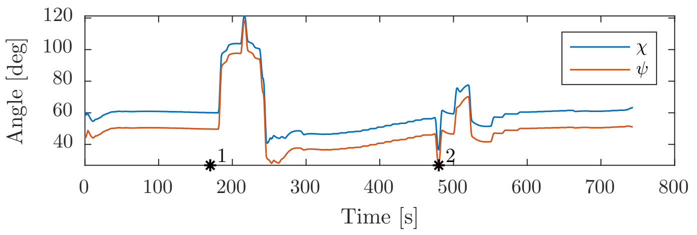
(B) Output from the state machine for each obstacle. The asterisks mark time stamps, and the colors correspond to the obstacle patch colors in the trajectory plot.

Figure 11. Scenario 3: Trajectory and COLREGs interpretation. The text marks denote the time steps $[170, 480]$ s.

crossing course with the ownship. With respect to the COLREGs, the ownship is required to give way to both obstacles since they are crossing from the ownship's starboard side. One can, however, argue that the maneuvers displayed by the obstacles are dangerous and displays poor seamanship, such that the ownship should not be held accountable if a collision occurred. Nevertheless, the hybrid COLAV system manages to avoid both obstacles. As seen in Figure 11B, the first obstacle is sufficiently far away from the ownship to be considered as a give-way situation when the state machine interprets the situation, and the mid-level algorithm plans a trajectory passing behind the first obstacle. The second obstacle maneuvers to port even closer to the ownship, resulting in



(A) Speed trajectories



(B) Angular trajectories

Figure 12. Scenario 3: Speed and angular trajectories. The asterisks mark the same time samples as in Figure 11.

Table 3. Minimum distance to static and moving obstacles for the simulation scenarios.

Scenario	**Minimum distance to static obstacles**	**Minimum distance to moving obstacle number**				
		1	**2**	**3**	**4**	**5**
Scenario 1	93.7 m	634.3 m	596.3 m	522.7 m	726.8 m	–
Scenario 2	118.2 m	185.5 m	228.3 m	1097.2 m	575.6 m	842.3 m
Scenario 3	1123.8 m	326.4 m	106.6 m	–	–	–

the distance to the critical point being within the threshold for entering the emergency situation when the state machine interprets the situation. In this situation, the mid-level algorithm disregards the obstacle and leaves it to the BC-MPC algorithm to avoid collision. As seen in Figure 12, the mid-level algorithm both reduces the speed and changes the course to avoid the first obstacle. When approaching the second obstacle, the BC-MPC algorithm initiates a speed reduction, and after some time also maneuver to starboard in order to pass behind the obstacle and resolve the situation.

6.5 Simulation summary

The simulation results show that the hybrid COLAV system is able to handle a wide range of situations, while also behaving in an energy-optimal way when moving obstacles are not interfering with the ownship trajectory. Table 3 shows the minimum distance to static and moving obstacles for the scenarios. The minimum distance to static obstacles is in Scenario 1 below the safety region size of the BC-MPC algorithm, which is intentional and caused by the algorithm using a smooth penalty function for interpreting static obstacles. The penalty function value increases linearly when moving further into the safety region, see (Eriksen and Breivik, 2019) for more details. The minimum distance to moving obstacles is a bit difficult to interpret, since the obstacle ship domains are non-circular, implying that the required clearance depends on relative position of the ownship with respect to the moving obstacles. However, we see that we have a larger clearance in head-on, give-way and stand-on situations where the obstacles comply with the COLREGs, and do not perform dangerous maneuvers (as in Scenario 3), compared to overtaking situations. The reason for this is that when overtaking (obstacle 2 in Scenario 2), we pass the obstacle on a parallel course, resulting in the minor axis of the moving obstacle ellipsis indicating the required clearance. Furthermore, we see that obstacle 1 in Scenario 2, which ignores its give-way obligation, comes significantly closer than other crossing obstacles except for those in Scenario 3. The reason for this is that the BC-MPC algorithm, which handles this situation, has a lower clearance requirement than the mid-level algorithm, which still should be considered as safe. In Scenario 3, the two obstacles display poor seamanship, and behave dangerously. Obstacle 1 is handled by the mid-level algorithm and passed with a clearance lower than the major axis of the mid-level algorithm, which is caused by the BC-MPC algorithm "cutting the corner". The clearance should still be considered safe since we are behind the obstacle, and the clearance requirements of the BC-MPC algorithm is enforced. Obstacle 2, which is placed in the emergency state and handled by the BC-MPC algorithm, is passed with a clearance of only 106.6 m. This is lower than the clearance to Obstacle 1 in Scenario 2 (which violated its stand-on requirement), and is due to the BC-MPC algorithm having a non-symmetric obstacle ship domain function allowing for a smaller clearance when passing behind an obstacle than in front.

For the three scenarios, the high-level planner used an average of 67 s with a maximum of 93 s to compute the solution. Since the high-level planner is intended to be run off-line, this is well within reasonable limits. The mid-level algorithm used 0.60 s on average, and a maximum of 2.1 s, which we consider to be real-time feasible since the mid-level algorithm only is run every 60 s. The BC-MPC algorithm used 0.29 s on average, and a maximum of 0.63 s, which we also consider to be real-time feasible when the BC-MPC algorithm is run every 5 s. The BC-MPC algorithm is highly parallelizable, which could reduce the BC-MPC runtime by a large magnitude if required.

7 CONCLUSION

In this paper, we have presented a three-layered hybrid COLAV system, compliant with COLREGs rules 8 and 13–17. As part of this, we have further developed the MPC-based mid-level COLAV algorithm in (Eriksen and Breivik, 2017b; Bitar et al., 2019a) to comply with COLREGs rules 13–16 and parts of Rule 17, which includes developing a state machine for COLREGs interpretation. The hybrid COLAV system has a well-defined division of labor, including an inherent understanding of COLREGs Rule 17, where the mid-level algorithm obeys stand-on situations, while the BC-MPC algorithm handles situations where give-way vessels does not maneuver.

The hybrid COLAV system is verified through simulations, where we in three scenarios challenge the system with a number of different situations. The scenarios include multi-obstacle situations with multiple simultaneously active COLREGs rules, and situations where obstacles violate the COLREGs. Collision is avoided in all the scenarios, and we show that the ownship follows an energy-optimized trajectory generated by the high-level planner when moving obstacles does not interfere with this trajectory.

For further work, we suggest to:

- Investigate if using situation-dependent entry and exit criteria parameters in the state machine improves the performance.

- Expand the state machine with the possibility of transitioning from head-on, give-way and overtaking states to the emergency state for situations where obstacles behave dangerously or hostile.
- Develop a methodology for deciding tuning parameters.
- Perform simulations with noisy obstacle estimates to investigate how the state machine and mid-level algorithm respond to this.
- Explore the possibilities for integrating the COLREGs interpretation in the mid-level NLP, relaxing the assumption of the current COLREGs situation being valid for the entire prediction horizon.
- Simulate scenarios where multiple vessels running the hybrid COLAV system interact with each other.
- Validate the hybrid COLAV system in full-scale experiments.

CONFLICT OF INTEREST STATEMENT

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

AUTHOR CONTRIBUTIONS

The work in this article is the result of a collaboration between BE and GB, supervised by MB and AL. The contributions to the mid-level and short-term algorithms are made by BE, while the COLREGs interpreter is developed in collaboration between GB and BE. GB has implemented the high-level planner, and prepared this for integration with the developed simulator. BE has taken lead on writing the paper, in collaboration with GB. MB and AL have provided valuable feedback in the writing process.

FUNDING

This work was supported by the Research Council of Norway through project number 269116, project number 244116, as well as the Centers of Excellence funding scheme with project number 223254.

REFERENCES

Andersson, J. A. E., Gillis, J., Horn, G., Rawlings, J. B., and Diehl, M. (2019). CasADi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation* 11, 1–36. doi:10.1007/s12532-018-0139-4

Benjamin, M. R., Leonard, J. J., Curcio, J. A., and Newman, P. M. (2006). A method for protocol-based collision avoidance between autonomous marine surface craft. *Journal of Field Robotics* 23, 333–346. doi:10.1002/rob.20121

Bitar, G., Breivik, M., and Lekkas, A. M. (2018). Energy-optimized path planning for autonomous ferries. In *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)* (Opatija, Croatia), 389–394. doi:10.1016/j.ifacol.2018.09.456

Bitar, G., Eriksen, B.-O. H., Lekkas, A. M., and Breivik, M. (2019a). Energy-optimized hybrid collision avoidance for ASVs. In *Proc. of the 17th IEEE European Control Conference (ECC)* (Naples, Italy), 2522–2529

Bitar, G., Vestad, V. N., Lekkas, A. M., and Breivik, M. (2019b). Warm-started optimized trajectory planning for ASVs. In *Proc. of the 12th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles (CAMS)*. In press. Preprint available at `https://arxiv.org/abs/1907.02696`

Campbell, S., Abu-Tair, M., and Naeem, W. (2014). An automatic COLREGs-compliant obstacle avoidance system for an unmanned surface vehicle. *Proc. of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment* 228, 108–121. doi:10.1177/1475090213498229

Casalino, G., Turetta, A., and Simetti, E. (2009). A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields. In *Proc. of the 2009 IEEE OCEANS-EUROPE Conference* (Bremen, Germany). doi:10.1109/oceanse.2009.5278104

Chauvin, C. (2011). Human factors and maritime safety. *Journal of Navigation* 64, 625–632. doi:10.1017/S0373463311000142

Cockcroft, A. N. and Lameijer, J. N. F. (2004). *A Guide to the Collision Avoidance Rules* (Elsevier)

Deuflhard, P. (2011). *Newton Methods for Nonlinear Problems* (Springer)

Eriksen, B.-O. H. and Breivik, M. (2017a). *Modeling, Identification and Control of High-Speed ASVs: Theory and Experiments* (Springer International Publishing). 407–431. doi:10.1007/978-3-319-55372-6_19

Eriksen, B.-O. H. and Breivik, M. (2017b). MPC-based mid-level collision avoidance for ASVs using nonlinear programming. In *Proc. of the 1st IEEE Conference on Control Technology and Applications (CCTA)* (Kohala Coast, Hawai'i, USA), 766–772. doi:10.1109/CCTA.2017.8062554

Eriksen, B.-O. H. and Breivik, M. (2018). A model-based speed and course controller for high-speed ASVs. In *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)* (Opatija, Croatia), 317–322. doi:10.1016/j.ifacol.2018.09.504

Eriksen, B.-O. H. and Breivik, M. (2019). Short-term ASV collision avoidance with static and moving obstacles Submitted to Modeling, Identification and Control, preprint available at `https://arxiv.org/abs/1907.04877`

Eriksen, B.-O. H., Breivik, M., Wilthil, E. F., Flåten, A. L., and Brekke, E. F. (2019). The branching-course MPC algorithm for maritime collision avoidance. *Journal of Field Robotics* In press, preprint available at `https://arxiv.org/abs/1907.00039`

Eriksen, B.-O. H., Wilthil, E. F., Flåten, A. L., Brekke, E. F., and Breivik, M. (2018). Radar-based maritime collision avoidance using dynamic window. In *Proc. of the 2018 IEEE Aerospace Conference* (Big Sky, Montana, USA), 1–9. doi:10.1109/AERO.2018.8396666

Hagen, I. B., Kufoalor, D. K. M., Brekke, E., and Johansen, T. A. (2018). MPC-based collision avoidance strategy for existing marine vessel guidance systems. In *Proc. of the 2018 IEEE International Conference on Robotics and Automation (ICRA)* (Brisbane, Australia), 7618–7623. doi:10.1109/ICRA.2018.8463182

Harati-Mokhtari, A., Wall, A., Brooks, P., and Wang, J. (2007). Automatic identification system (AIS): Data reliability and human error implications. *Journal of Navigation* 60, 373–389. doi:10.1017/S0373463307004298

Kufoalor, D. K. M., Brekke, E. F., and Johansen, T. A. (2018). Proactive collision avoidance for ASVs using a dynamic reciprocal velocity obstacles method. In *Proc. of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2402–2409. doi:10.1109/IROS.2018.8594382

Kuwata, Y., Wolf, M. T., Zarzhitsky, D., and Huntsberger, T. L. (2014). Safe maritime autonomous navigation with COLREGS, using velocity obstacles. *IEEE J. Oceanic Eng.* 39, 110–119. doi:10.1109/joe.2013.2254214

Levander, O. (2017). Autonomous ships on the high seas. *IEEE Spectrum* 54, 26–31. doi:10.1109/MSPEC.2017.7833502

Loe, Ø. A. G. (2008). *Collision Avoidance for Unmanned Surface Vehicles*. Master's thesis, Norwegian University of Science and Technology, Trondheim, Norway

Szlapczynski, R. and Szlapczynska, J. (2017). Review of ship safety domains: Models and applications. *Ocean Engineering* 145, 277–289. doi:https://doi.org/10.1016/j.oceaneng.2017.09.020

Tam, C. and Bucknall, R. (2010). Collision risk assessment for ships. *Journal of Marine Science and Technology* 15, 257–270. doi:10.1007/s00773-010-0089-7

Švec, P., Shah, B. C., Bertaska, I. R., Alvarez, J., Sinisterra, A. J., von Ellenrieder, K., et al. (2013). Dynamics-aware target following for an autonomous surface vehicle operating under COLREGs in civilian traffic. In *Proc. of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Tokyo, Japan), 3871–3878. doi:10.1109/IROS.2013.6696910

Wächter, A. and Biegler, L. T. (2005). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* 106, 25–57. doi:10.1007/s10107-004-0559-y

Appendix A

PSEUDOCODE FOR MODEL IDENTIFICATION OF HIGH-SPEED ASVS

This appendix contains pseudocode for the model identification procedure presented in Paper B. Equation and section numbers in this appendix refer to corresponding equations and sections in Paper B.

Algorithm 1 describes how the regularization parameter weight λ in (11) is determined using leave-one-out cross validation (CV), in accordance with Section 3.4.2.

Algorithm 1 Pseudocode for selecting the regularization parameter using leave-one-out CV

```
Given a set of regularization parameters $\lambda \in \{\lambda_1, \lambda_2, \ldots, \lambda_I\}$ and a data set $\mathcal{D}$ of size $N_{\mathcal{D}}$
$\boldsymbol{\epsilon} \leftarrow \mathbf{0}_{1\times I}$
Divide the data set $\mathcal{D}$ into $N_{\mathcal{D}}$ combinations of training sets $\mathcal{D}_t$ and verification sets $\mathcal{D}_v$
for $i = 1 : I$ do
    for all $N_{\mathcal{D}}$ combinations of $\mathcal{D}_t$ and $\mathcal{D}_v$ do
        $\boldsymbol{\beta} \leftarrow$ solution from minimizing (11) using $R(\cdot) = \|\cdot\|_1$, $\lambda = \lambda_i$ and $(\boldsymbol{x}, y) \in \mathcal{D}_t$
        $\bar{\epsilon} \leftarrow$ evaluate (9) using $(\boldsymbol{x}, y) \in \mathcal{D}_v$
        $\epsilon_i \leftarrow \epsilon_i + \frac{1}{N_{\mathcal{D}}}\bar{\epsilon}$
    end for
end for
Find $i$ s.t. $\epsilon_i = \min(\boldsymbol{\epsilon})$, and choose $\lambda = \lambda_i$
```

Algorithm 2 describes how the damping parameter vectors $\boldsymbol{\beta}_{\sigma_U}$ and $\boldsymbol{\beta}_{\sigma_r}$ are found, in accordance with Section 3.4.3.

Algorithm 3 describes how the hyperparameters of the asymptotic inertia basis functions are found using leave-one-out CV, in accordance with Section 3.4.4.

Algorithm 4 describes how the inertia parameter vectors $\boldsymbol{\beta}_{m_U}$ and $\boldsymbol{\beta}_{m_r}$ are found, in accordance with Section 3.4.4.

Algorithm 2 Pseudocode for identifying parameters for the damping term

```
Given the measurement set $\mathcal{D}_{\boldsymbol{\sigma}}$
$\lambda_{\sigma_U} \leftarrow$ Algorithm 1 with $\mathcal{D} = \{(\boldsymbol{x}, y) = (\boldsymbol{x}, \sigma_U) \in \mathcal{D}_{\boldsymbol{\sigma}}\}$
$\boldsymbol{\beta}_{\sigma_U} \leftarrow$ solution from minimizing (11) using $R(\cdot) = \|\cdot\|_1$, $\lambda = \lambda_{\sigma_U}$, $(\boldsymbol{x}, y) = (\boldsymbol{x}, \sigma_U) \in \mathcal{D}_{\boldsymbol{\sigma}}$ and $\boldsymbol{\phi}(\boldsymbol{x})$ from (12)
$\lambda_{\sigma_r} \leftarrow$ Algorithm 1 with $\mathcal{D} = \{(\boldsymbol{x}, y) = (\boldsymbol{x}, \sigma_r) \in \mathcal{D}_{\boldsymbol{\sigma}}\}$
$\boldsymbol{\beta}_{\sigma_r} \leftarrow$ solution from minimizing (11) using $R(\cdot) = \|\cdot\|_1$, $\lambda = \lambda_{\sigma_r}$, $(\boldsymbol{x}, y) = (\boldsymbol{x}, \sigma_r) \in \mathcal{D}_{\boldsymbol{\sigma}}$ and $\boldsymbol{\phi}(\boldsymbol{x})$ from (12)
```

Algorithm 3 Pseudocode for identifying hyperparameters for the asymptotic basis function

```
Given sets of hyperparameters $a \in \{a_1, a_2, \dots, a_I\}$, $b \in \{b_1, b_2, \dots, b_J\}$ and a data set $\mathcal{D}$ of size $N_{\mathcal{D}}$
$\lambda_{hyp} \leftarrow$ Regularization parameter for identifying $(a, b)$          ▷ Tuning parameter
$\boldsymbol{\epsilon} \leftarrow \mathbf{0}_{I \times J}$
Divide the data set $\mathcal{D}$ into $N_{\mathcal{D}}$ combinations of training sets $\mathcal{D}_t$ and verification sets $\mathcal{D}_v$
for $i = 1 : I$ do
    for $j = 1 : J$ do
        for all $N_{\mathcal{D}}$ combinations of $\mathcal{D}_t$ and $\mathcal{D}_v$ do
            $\boldsymbol{\beta} \leftarrow$ solution from minimizing (9) using $(\boldsymbol{x}, y) \in \mathcal{D}_t$
            $\bar{\epsilon} \leftarrow$ evaluate (9) using $(\boldsymbol{x}, y) \in \mathcal{D}_v$
            $\epsilon_{ij} \leftarrow \epsilon_{ij} + \frac{1}{N_{\mathcal{D}}}\left(\bar{\epsilon} + \lambda_{hyp}\left\|\begin{bmatrix} a & b \end{bmatrix}^T\right\|_1\right)$
        end for
    end for
end for
Find $(i, j)$ s.t. $\epsilon_{ij} = \min(\boldsymbol{\epsilon})$, and choose $a = a_i$ and $b = b_j$
```

Algorithm 4 Pseudo code for identifying parameters for the inertia term

```
Given the measurement sets $\mathcal{D}_{m_U}$ and $\mathcal{D}_{m_r}$
$(a_{m_U}, b_{m_U}) \leftarrow$ Algorithm 3 with $\mathcal{D} = \{(\boldsymbol{x}, y) = (\boldsymbol{x}, m_U) \in \mathcal{D}_{m_U}\}$
$\lambda_{m_U} \leftarrow$ Algorithm 1 with $\mathcal{D} = \{(\boldsymbol{x}, y) = (\boldsymbol{x}, m_U) \in \mathcal{D}_{m_U}\}$, $\boldsymbol{\phi}(\boldsymbol{x})$ from (13) and $(a, b) = (a_{m_U}, b_{m_U})$
$\boldsymbol{\beta}_{m_U} \leftarrow$ solution from minimizing (11) using $R(\cdot) = \|\cdot\|_1$, $\lambda = \lambda_{m_U}$, $\boldsymbol{\phi}(\boldsymbol{x})$ from (13) with $(a, b) = (a_{m_U}, b_{m_U})$ and $(\boldsymbol{x}, y) = (\boldsymbol{x}, \sigma_{m_U}) \in \mathcal{D}_{m_U}$
$(a_{m_r}, b_{m_r}) \leftarrow$ Algorithm 3 with $\mathcal{D} = \{(\boldsymbol{x}, y) = (\boldsymbol{x}, m_r) \in \mathcal{D}_{m_r}\}$
$\lambda_{m_r} \leftarrow$ Algorithm 1 with $\mathcal{D} = \{(\boldsymbol{x}, y) = (\boldsymbol{x}, m_r) \in \mathcal{D}_{m_r}\}$, $\boldsymbol{\phi}(\boldsymbol{x})$ from (13) and $(a, b) = (a_{m_r}, b_{m_r})$
$\boldsymbol{\beta}_{m_r} \leftarrow$ solution from minimizing (11) using $R(\cdot) = \|\cdot\|_1$, $\lambda = \lambda_{m_r}$, $\boldsymbol{\phi}(\boldsymbol{x})$ from (13) with $(a, b) = (a_{m_r}, b_{m_r})$ and $(\boldsymbol{x}, y) = (\boldsymbol{x}, \sigma_{m_r}) \in \mathcal{D}_{m_r}$
```

REFERENCES

[1] S. Bennett, *A history of control engineering 1800–1930.* The Institution of Electrical Engineers, 1979, ISBN: 0-906048-07-9.

[2] ——, "Nicholas Minorsky and the automatic steering of ships", *IEEE Control Systems Magazine*, vol. 4, no. 4, pp. 10–15, 1984. DOI: 10.1109/MCS.1984.1104827.

[3] M. Breivik and G. Sand, "Jens Glad Balchen: A Norwegian pioneer in engineering cybernetics", *Modeling, Identification and Control*, vol. 30, no. 3, pp. 101–125, 2009. DOI: 10.4173/mic.2009.3.2.

[4] M. Breivik, "Topics in guided motion control of marine vehicles", Doctoral dissertation, Norwegian University of Science and Technology (NTNU), 2010, ISBN: 978-82-471-2085-9.

[5] R. Hussain and S. Zeadally, "Autonomous cars: Research results, issues and future challenges", *IEEE Communications Surveys Tutorials*, 2018, Accepted for publication in a future issue of the journal, but has not been fully edited. DOI: 10.1109/COMST.2018.2869360.

[6] J. Larson, M. Bruch, and J. Ebken, "Autonomous navigation and obstacle avoidance for unmanned surface vehicles", in *Proc. of the 2006 Defense and Security Symposium*, (Orlando, Florida, USA), 2006. DOI: 10.1117/12.663798.

[7] Leidos, *Sea hunter reaches new milestone for autonomy*, https://www.leidos.com/insights/sea-hunter-reaches-new-milestone-autonomy, Accessed: 2019-05-16, 2019.

[8] Rachel Courtland, *DARPA's self-driving submarine hunter steers like a human*, https://spectrum.ieee.org/automaton/robotics/military-robots/darpa-actuv-self-driving-submarine-hunter-steers-like-a-human, Accessed: 2019-05-15, 2016.

[9] U. Skoglund, *Driverless ferries to replace footbridges*, https://gemini-researchnews.com/2018/06/driverless-ferries-to-replace-foot-bridges/, Accessed: 2019-05-30, 2018.

[10] C. Paris, *Norway takes lead in race to build autonomous cargo ships*, https://www.wsj.com/articles/norway-takes-lead-in-race-to-build-autonomous-cargo-ships-1500721202, Accessed: 2019-05-22, 2017.

[11] C. Jallal, *Rolls-Royce and Finferries demonstrate world's first fully autonomous ferry*, https://www.marinemec.com/news/view,rollsroyce-and-finferries-demonstrate-worlds-first-fully-autonomous-ferry_56102.htm, Accessed: 2019-04-11, 2018.

[12] The Maritime Executive, *Wärtsilä conducts autonomous ferry voyage and docking*, https://www.maritime-executive.com/article/waertsilae-conducts-autonomous-ferry-voyage-and-docking, Accessed: 2019-05-16, 2018.

[13] C. Chauvin, "Human factors and maritime safety", *Journal of Navigation*, vol. 64, no. 4, pp. 625–632, 2011. DOI: 10.1017/S0373463311000142.

[14] O. Levander, "Autonomous ships on the high seas", *IEEE Spectrum*, vol. 54, no. 2, pp. 26–31, 2017. DOI: 10.1109/MSPEC.2017.7833502.

[15] A. N. Cockcroft and J. N. F. Lameijer, *A Guide to the Collision Avoidance Rules*. Elsevier, 2004, ISBN: 0-7506-6179-8.

[16] E. F. Wilthil, "Maritime target tracking with varying sensor performance", Under evaluation, Doctoral dissertation, Norwegian University of Science and Technology (NTNU), 2019.

[17] E. F. Wilthil, A. L. Flåten, and E. F. Brekke, "A target tracking system for ASV collision avoidance based on the PDAF", in *Sensing and Control for Autonomous Vehicles: Applications to Land, Water and Air Vehicles*, T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer, Eds. Springer International Publishing, 2017, pp. 269–288, ISBN: 978-3-319-55372-6. DOI: 10.1007/978-3-319-55372-6_13.

[18] B.-O. H. Eriksen and M. Breivik, "Modeling, identification and control of high-speed ASVs: Theory and experiments", in *Sensing and Control for Autonomous Vehicles: Applications to Land, Water and Air Vehicles*, T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer, Eds. Springer International Publishing, 2017, pp. 407–431, ISBN: 978-3-319-55372-6. DOI: 10.1007/978-3-319-55372-6_19.

[19] B.-O. H. Eriksen, M. Breivik, E. F. Wilthil, A. L. Flåten, and E. F. Brekke, "The branching-course MPC algorithm for maritime collision avoidance", 2019, Accepted for publication in Journal of Field Robotics, available at https://arxiv.org/abs/1907.00039.

[20] B.-O. H. Eriksen, G. Bitar, M. Breivik, and A. M. Lekkas, "Hybrid collision avoidance for ASVs compliant with COLREGs rules 8 and 13–17", 2019, Submitted to Frontiers in Robotics and AI, available at https://arxiv.org/abs/1907.00198.

[21] B.-O. H. Eriksen, M. Breivik, K. Y. Pettersen, and M. S. Wiig, "A modified dynamic window algorithm for horizontal collision avoidance for AUVs", in *Proc. of the 2016 IEEE Conference on Control Applications (CCA)*, (Buenos Aires, Argentina), 2016, pp. 499–506. DOI: 10.1109/CCA.2016.7587879.

[22] B.-O. H. Eriksen and M. Breivik, "MPC-based mid-level collision avoidance for ASVs using nonlinear programming", in *Proc. of the 1st IEEE Conference on Control Technology and Applications (CCTA)*, (Mauna Lani, Hawai'i, USA), 2017, pp. 766–772. DOI: 10.1109/CCTA.2017.8062554.

[23] B.-O. H. Eriksen, E. F. Wilthil, A. L. Flåten, E. F. Brekke, and M. Breivik, "Radar-based maritime collision avoidance using dynamic window", in *Proc. of the 2018 IEEE Aerospace Conference*, (Big Sky, Montana, USA), 2018, pp. 1–9. DOI: 10.1109/AERO.2018.8396666.

[24] B.-O. H. Eriksen and M. Breivik, "A model-based speed and course controller for high-speed ASVs", in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*, (Opatija, Croatia), 2018, pp. 317–322. DOI: 10.1016/j.ifacol.2018.09.504.

[25] G. Bitar, B.-O. H. Eriksen, A. M. Lekkas, and M. Breivik, "Energy-optimized hybrid collision avoidance for ASVs", in *Proc. of the 17th IEEE European Control Conference (ECC)*, (Naples, Italy), 2019, pp. 2522–2529.

[26] B.-O. H. Eriksen and M. Breivik, "Short-term ASV collision avoidance with static and moving obstacles", 2019, Submitted to Modeling, Identification and Control, available at https://arxiv.org/abs/1907.04877.

[27] S. Hexeberg, A. L. Flåten, B.-O. H. Eriksen, and E. F. Brekke, "AIS-based vessel trajectory prediction", in *Proc. of the 20th IEEE International Conference on Information Fusion (FUSION)*, (Xi'an, China), 2017, pp. 1–8. DOI: 10.23919/ICIF.2017.8009762.

[28] M. E. N. Sørensen, M. Breivik, and B.-O. H. Eriksen, “A ship heading and speed control concept inherently satisfying actuator constraints”, in *Proc. of the 1st IEEE Conference on Control Technology and Applications (CCTA)*, (Mauna Lani, Hawai’i, USA), 2017, pp. 323–330. DOI: 10.1109/CCTA.2017.8062483.

[29] B. R. Dalsnes, S. Hexeberg, A. L. Flåten, B.-O. H. Eriksen, and E. F. Brekke, “The neighbor course distribution method with Gaussian mixture models for AIS-based vessel trajectory prediction”, in *Proc. of the 21st IEEE International Conference on Information Fusion (FUSION)*, (Cambridge, UK), 2018, pp. 580–587. DOI: 10.23919/ICIF.2018.8455607.

[30] E. Serigstad, B.-O. H. Eriksen, and M. Breivik, “Hybrid collision avoidance for autonomous surface vehicles”, in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles (CAMS) 2018*, (Opatija, Croatia), 2018, pp. 1–7. DOI: 10.1016/j.ifacol.2018.09.460.

[31] M. E. N. Sørensen, O. N. Lyngstadaas, B.-O. H. Eriksen, and M. Breivik, “A dynamic window-based controller for dynamic positioning satisfying actuator magnitude constraints”, in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*, (Opatija, Croatia), 2018, pp. 140–146. DOI: 10.1016/j.ifacol.2018.09.483.

[32] H. Hanafusa, T. Yoshikawa, and Y. Nakamura, “Analysis and control of articulated robot arms with redundancy”, in *Proc. of the 8th IFAC World Congress on Control Science and Technology for the Progress of Society*, (Kyoto, Japan), 1981, pp. 1927–1932. DOI: 10.1016/S1474-6670(17)63754-6.

[33] O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots”, in *Proc. of the 1985 IEEE International Conference on Robotics and Automation (ICRA)*, (St. Louis, Missouri, USA), 1985, pp. 500–505. DOI: 10.1109/ROBOT.1985.1087247.

[34] ——, “Real-time obstacle avoidance for manipulators and mobile robots”, *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986. DOI: 10.1177/027836498600500106.

[35] J. Borenstein and Y. Koren, “Real-time obstacle avoidance for fast mobile robots”, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 5, pp. 1179–1187, 1989. DOI: 10.1109/21.44033.

[36] ——, "The vector field histogram-fast obstacle avoidance for mobile robots", *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 278–288, 1991. DOI: 10.1109/70.88137.

[37] O. Brock and O. Khatib, "High-speed navigation using the global dynamic window approach", in *Proc. of the 1999 IEEE International Conference on Robotics and Automation (ICRA)*, (Detroit, Michigan), 1999, pp. 341–346. DOI: 10.1109/ROBOT.1999.770002.

[38] P. Ögren and N. Leonard, "A convergent dynamic window approach to obstacle avoidance", *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 188–195, 2005. DOI: 10.1109/tro.2004.838008.

[39] E. J. Rodríguez-Seda, C. Tang, M. W. Spong, and D. M. Stipanovic, "Trajectory tracking with collision avoidance for nonholonomic vehicles with acceleration constraints and limited sensing", *The International Journal of Robotics Research*, vol. 33, no. 12, pp. 1569–1592, 2014. DOI: 10.1177/0278364914537130.

[40] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation", in *Proc. of the 1991 IEEE International Conference on Robotics and Automation (ICRA)*, (Sacramento, California, USA), 1991, pp. 1398–1404. DOI: 10.1109/robot.1991.131810.

[41] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance", *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997. DOI: 10.1109/100.580977.

[42] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles", *International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998. DOI: 10.1177/027836499801700706.

[43] Ø. A. G. Loe, "Collision avoidance for unmanned surface vehicles", Master's thesis, Norwegian University of Science and Technology (NTNU), 2008.

[44] G. Casalino, A. Turetta, and E. Simetti, "A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields", in *Proc. of the 2009 IEEE OCEANS-EUROPE Conference*, (Bremen, Germany), 2009. DOI: 10.1109/oceanse.2009.5278104.

[45] P. Švec, B. C. Shah, I. R. Bertaska, J. Alvarez, A. J. Sinisterra, K. von Ellenrieder, M. Dhanak, and S. K. Gupta, "Dynamics-aware target following for an autonomous surface vehicle operating under COLREGs in civilian traffic", in *Proc. of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Tokyo, Japan), 2013, pp. 3871–3878. DOI: 10.1109/IROS.2013.6696910.

[46] J. D. Luse, "Collision avoidance systems and the rules of the nautical road", *Journal of Navigation*, vol. 19, no. 1, pp. 80–88, 1972. DOI: 10.1002/j.2161-4296.1972.tb00129.x.

[47] J. F. Kemp, "Two hundred years of the collision regulations", *Journal of Navigation*, vol. 29, no. 4, pp. 341–349, 1976. DOI: 10.1017/S0373463300039308.

[48] J. Armstrong and D. M. Williams, "The steamboat, safety and the state: Government reaction to new technology in a period of laissez-faire", *The Mariner's Mirror*, vol. 89, no. 2, pp. 167–184, 2003. DOI: 10.1080/00253359.2003.10659284.

[49] A. E. Fiore, R. E. Anderson, and L. J. Kapanka, "Historical approach to marine collision avoidance", *Navigation*, vol. 18, no. 1, pp. 116–133, 1971. DOI: 10.1002/j.2161-4296.1971.tb00079.x.

[50] P. D. L. Williams, "Civil marine radar – a review and a way ahead", *Journal of Navigation*, vol. 51, no. 3, pp. 394–403, 1998. DOI: 10.1017/S0373463398007942.

[51] C. Tam, R. Bucknall, and A. Greig, "Review of collision avoidance and path planning methods for ships in close range encounters", *Journal of Navigation*, vol. 62, no. 03, pp. 455–476, 2009. DOI: 10.1017/s0373463308005134.

[52] IMO, *AIS transponders*, http://www.imo.org/en/OurWork/Safety/Navigation/Pages/AIS.aspx, Accessed: 2018-08-31, 2018.

[53] A. Harati-Mokhtari, A. Wall, P. Brooks, and J. Wang, "Automatic identification system (AIS): Data reliability and human error implications", *Journal of Navigation*, vol. 60, no. 3, pp. 373–389, 2007. DOI: 10.1017/S0373463307004298.

[54] T. Statheros, G. Howells, and K. M. Maier, "Autonomous ship collision avoidance navigation concepts, technologies and techniques", *Journal of Navigation*, vol. 61, no. 01, pp. 129–142, 2008. DOI: 10.1017/s037346330700447x.

[55] S. Campbell, W. Naeem, and G. Irwin, “A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres”, *Annual Reviews in Control*, vol. 36, no. 2, pp. 267–283, 2012. DOI: 10.1016/j.arcontrol.2012.09.008.

[56] O. Mitrofanov, “An anti-collision indicator”, *Journal of Navigation*, vol. 21, no. 2, pp. 163–170, 1968. DOI: 10.1017/S0373463300030319.

[57] C. E. Moore and J. D. Elpi, “Optimum collision avoidance for merchant ships”, *IEEE Transactions on Industry Applications*, vol. IA-9, no. 6, pp. 640–647, 1973. DOI: 10.1109/TIA.1973.349987.

[58] K. G. Bjerva, *Norcontrol: maritim innovasjon siden 1965*. Norcontrol, Kongsberg maritim pensjonistforening, 2006.

[59] I. Höivold, “Norwegian research and development in the field of ship automation”, *Modeling, Identification and Control*, vol. 5, no. 3, pp. 171–178, 1984. DOI: 10.4173/mic.1984.3.4.

[60] R. F. Riggs, “A modern collision avoidance display technique”, *Journal of Navigation*, vol. 28, no. 2, pp. 143–155, 1975. DOI: 10.1017/S0373463300037681.

[61] S. L. Gravely jr., “Collision avoidance devices on board U.S. Navy ships”, Destroyer Development Group, Tech. Rep., 1975.

[62] K. Ravenna, “Collision avoidance systems - the evolution continues”, in *Proc. of the 1980 IEEE OCEANS Conference*, (Seattle, Washington, USA), 1980, pp. 193–197. DOI: 10.1109/OCEANS.1980.1151385.

[63] T. Miloh and S. D. Sharma, “Maritime collision avoidance as a differential game”, *Schiffstechnik*, vol. 24, no. 116, pp. 69–88, 1977. DOI: 10.15480/882.476.

[64] A. W. Merz and J. S. Karmarkar, “Collision avoidance systems and optimal turn manoeuvres”, *Journal of Navigation*, vol. 29, no. 2, pp. 160–174, 1976. DOI: 10.1017/S0373463300030150.

[65] C. de Wit and J. Oppe, “Optimal collision avoidance in unconfined waters”, *Journal of Navigation*, vol. 26, no. 4, pp. 296–303, 1979. DOI: 10.1002/j.2161-4296.1979.tb01389.x.

[66] R. Hooke and T. A. Jeeves, ““Direct search” solution of numerical and statistical problems”, *Journal of the ACM*, vol. 8, no. 2, pp. 212–229, 1961. DOI: 10.1145/321062.321069.

[67] T. Degré and X. Lefèvre, “A collision avoidance system”, *Journal of Navigation*, vol. 34, no. 2, pp. 294–302, 1981. DOI: 10.1017/S0373463300021408.

[68] R. Kalafus, “Radio/radar collision avoidance aid”, in *Proc. of the IEEE OCEANS 81 Conference*, (Boston, Massachusetts, USA), 1981, pp. 988–991. DOI: 10.1109/OCEANS.1981.1151559.

[69] M. J. Dove, R. S. Burns, and C. T. Stockel, “An automatic collision avoidance and guidance system for marine vehicles in confined waters”, *Journal of Navigation*, vol. 39, no. 2, pp. 180–190, 1986. DOI: 10.1017/S0373463300000059.

[70] K. Kwik, “Calculation of ship collision avoidance manoeuvres: A simplified approach”, *Ocean Engineering*, vol. 16, no. 5, pp. 475–491, 1989. DOI: 10.1016/0029-8018(89)90048-6.

[71] F. P. Coenen, G. P. Smeaton, and A. G. Bole, “Knowledge-based collision avoidance”, *Journal of Navigation*, vol. 42, no. 1, pp. 107–116, 1989. DOI: 10.1017/S0373463300015125.

[72] Y. Yavin, C. Frangos, and T. Miloh, “Computation of feasible control trajectories for the navigation of a ship around an obstacle in the presence of a sea current”, *Mathematical and Computer Modelling*, vol. 21, no. 3, pp. 99–117, 1995. DOI: 10.1016/0895-7177(94)00218-D.

[73] A. Miele, T. Wang, C. S. Chao, and J. B. Dabney, “Optimal control of a ship for collision avoidance maneuvers”, *Journal of Optimization Theory and Applications*, vol. 103, no. 3, pp. 495–519, 1999. DOI: 10.1023/A:1021775722287.

[74] J. Lisowski and M. Mohamed-Seghir, “Application of fuzzy set theory in the safe ship control process”, in *Proc. of the 1998 IFAC Conference on Control Applications in Marine Systems (CAMS)*, (Fukuoka, Japan), 1998, pp. 181–185. DOI: 10.1016/S1474-6670(17)38452-5.

[75] M. Ito, Feifei Zhnng, and N. Yoshida, “Collision avoidance control of ship with genetic algorithm”, in *Proc. of the 1999 IEEE International Conference on Control Applications*, (Kohala Coast, Hawai’i, USA), 1999, pp. 1791–1796. DOI: 10.1109/CCA.1999.801243.

[76] R. Smierzchalski and Z. Michalewicz, “Modeling of ship trajectory in collision situations by an evolutionary algorithm”, *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 3, pp. 227–241, 2000. DOI: 10.1109/4235.873234.

[77] Y. Sato and H. Ishii, “Study of a collision-avoidance system for ships”, *Control Engineering Practice*, vol. 6, no. 9, pp. 1141–1149, 1998. DOI: 10.1016/S0967-0661(98)00107-5.

[78] M. R. Benjamin and J. A. Curcio, “COLREGS-based navigation of autonomous marine vehicles”, in *Proc. of the 2004 IEEE/OES Autonomous Underwater Vehicles Conference*, (Sebasco, ME, USA), 2004, pp. 32–39. DOI: 10.1109/AUV.2004.1431190.

[79] M. R. Benjamin, J. J. Leonard, J. A. Curcio, and P. M. Newman, “A method for protocol-based collision avoidance between autonomous marine surface craft”, *Journal of Field Robotics*, vol. 23, no. 5, pp. 333–346, 2006. DOI: 10.1002/rob.20121.

[80] M. R. Benjamin, H. Schmidt, P. M. Newman, and J. J. Leonard, “Nested autonomy for unmanned marine vehicles with MOOS-IvP”, *Journal of Field Robotics*, vol. 27, no. 6, pp. 834–875, 2010. DOI: 10.1002/rob.20370.

[81] J. Larson, M. Bruch, R. Halterman, J. Rogers, and R. Webster, “Advances in autonomous obstacle avoidance for unmanned surface vehicles”, in *Proc. of the 2007 Unmanned Systems North America Conference*, (Washington, D.C., USA), 2007.

[82] L. Elkins, D. Sellers, and W. R. Monach, “The autonomous maritime navigation (AMN) project: Field tests, autonomous and cooperative behaviors, data fusion, sensors, and vehicles”, *Journal of Field Robotics*, vol. 27, no. 6, pp. 790–818, 2010. DOI: 10.1002/rob.20367.

[83] T. Huntsberger, H. Aghazarian, A. Castano, G. Woodward, C. Padgett, and D. Gaines, “Intelligent autonomy for unmanned sea surface and underwater vehicles”, in *Proc of the 2008 AUVSI Unmanned Systems North America Conference*, (San Diego, California), 2008, pp. 943–958.

[84] T. Huntsberger, H. Aghazarian, A. Howard, and D. C. Trotz, “Stereo vision-based navigation for autonomous surface vessels”, *Journal of Field Robotics*, vol. 28, no. 1, pp. 3–18, 2011. DOI: 10.1002/rob.20380.

[85] Y. Kuwata, M. T. Wolf, D. Zarzhitsky, and T. L. Huntsberger, “Safe maritime autonomous navigation with COLREGS, using velocity obstacles”, *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 110–119, 2014. DOI: 10.1109/joe.2013.2254214.

[86] M. Schuster, M. Blaich, and J. Reuter, “Collision avoidance for vessels using a low-cost radar sensor”, in *Proc. of the 19th IFAC World Congress*, (Cape Town, South Africa), 2014, pp. 9673–9678. DOI: 10.3182/20140824-6-ZA-1003.01872.

[87] M. Blaich, M. Rosenfelder, M. Schuster, O. Bittel, and J. Reuter, "Fast grid based collision avoidance for vessels using A* search algorithm", in *Proc. of the 17th IEEE International Conference on Methods Models in Automation Robotics (MMAR)*, (Miedzyzdrojie, Poland), 2012, pp. 385–390. DOI: 10.1109/MMAR.2012.6347858.

[88] E. M. Goodwin, "A statistical study of ship domains", *Journal of Navigation*, vol. 28, no. 3, pp. 328–344, 1975. DOI: 10.1017/S0373463300041230.

[89] T. A. Johansen, T. Perez, and A. Cristofaro, "Ship collision avoidance and COLREGS compliance using simulation-based control behavior selection with predictive hazard assessment", *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3407–3422, 2016. DOI: 10.1109/tits.2016.2551780.

[90] T. Barton, *The DARPA ACTUV program*, Presented at the 2016 Autonomous Ship Technology Symposium, Amsterdam, Netherlands, 2016.

[91] G. Bitar, M. Breivik, and A. M. Lekkas, "Energy-optimized path planning for autonomous ferries", in *Proc. of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)*, (Opatija, Croatia), 2018, pp. 389–394. DOI: 10.1016/j.ifacol.2018.09.456.

[92] S. Hexeberg, "AIS-based vessel trajectory prediction for ASV collision avoidance", Master's thesis, Norwegian University of Science and Technology (NTNU), 2017.

[93] B. R. Dalsnes, "Long-term vessel prediction using AIS data", Master's thesis, Norwegian University of Science and Technology (NTNU), 2018.

[94] G. Bitar, V. N. Vestad, A. M. Lekkas, and M. Breivik, "Warm-started optimized trajectory planning for ASVs", in *Proc. of the 12th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles (CAMS)*, Available at https://arxiv.org/abs/1907.02696, 2019, In press.

ISBN 978-82-326-4056-0 (printed version)
ISBN 978-82-326-4057-7 (electronic version)
ISSN 1503-8181

Doctoral Thesis

Bjørn-Olav Holtung Eriksen

Doctoral theses at NTNU, 2019:230

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

NTNU

Doctoral theses at NTNU, 2019:230

Bjørn-Olav Holtung Eriksen

Collision Avoidance and Motion Control for Autonomous Surface Vehicles

NTNU
Norwegian University of
Science and Technology