Received: 16 November 2018 | Revised: 8 July 2019 | Accepted: 10 July 2019

DOI: 10.1002/rob.21900

REGULAR ARTICLE

WILEY

The branching-course model predictive control 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

Correspondence
Bjørn-Olav H. Eriksen, Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), Trondheim 7491, Norway.
Email: bjorn-olav.h.eriksen@ieee.org

Funding information
Norges Forskningsråd, Grant/Award Numbers: 244116, 223254

Abstract

This article presents a new algorithm for short-term maritime collision avoidance (COLAV) named the branching-course model predictive control (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, for example, radars for obstacle detection and tracking. Exteroceptive sensors do not require vessel-to-vessel communication, which enables COLAV toward vessels not equipped with, for example, automatic identification system 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.

KEYWORDS
control, marine robotics, planning

1 | INTRODUCTION

Today's society moves rapidly toward an increased level of automation. The development of autonomous cars is spearheading this trend, as exemplified by the efforts made by, for example, 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 40,000 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 vehicle (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 & 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.

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, Sutton, & Chudley, 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, and so forth. 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 (Fiorini & Shiller, 1998; Kuwata, Wolf, Zarzhitsky, & Huntsberger, 2014) and the dynamic window (DW); (Fox, Burgard, & Thrun, 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* (Blaich, Rosenfelder, Schuster, Bittel, & Reuter, 2012; Hart, Nilsson, & Raphael, 1968) and the rapidly exploring random tree (La Valle, 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 Figure 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

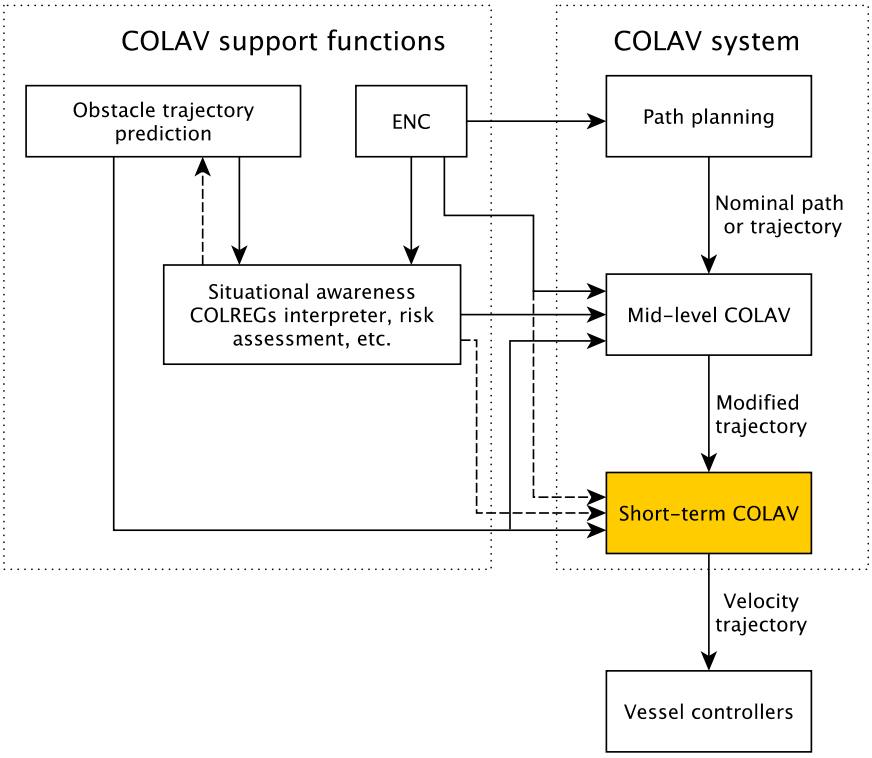

FIGURE 1 A hybrid COLAV architecture with three levels. The support functions provide relevant information for the COLAV algorithms, including obstacle trajectories, static obstacles from electronic nautical charts (ENC), and situational awareness in the form of COLREGs situations. The short-term layer does not currently utilize information from ENC or situational awareness [Color figure can be viewed at wileyonlinelibrary.com]

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 & Breivik, 2017b). Furthermore, the short-term layer should be able to avoid collision in emergency situations, for example, when obstacles do not maneuver in accordance with COLREGs.

COLAV algorithms depend on information about obstacle position, speed, and course 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, Wall, Brooks, & Wang, 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 to detect them. A commonly used exteroceptive sensor at sea is radar. However, the data from a radar usually include a fair amount of noise, which makes this sensor more complex and difficult to work with than AIS (Eriksen, Wilthil, Flåten, Brekke, & Breivik, 2018). On-board radars have been used for full-scale COLAV experiments based on the A* algorithm in Schuster, Blaich, and Reuter (2014), and using a modified version of the DW algorithm in Eriksen et al. (2018). In Elkins, Sellers, and Monach (2010) and 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, for example, ground and automotive robots (Gray, Ali, Gao, Hedrick, & Borrelli, 2013; Keller, Haß, Seewald, & Bertram, 2015; Ögren & Leonard, 2005), aerospace applications (Kuwata & How, 2011), and underwater vehicles (Caldwell, Dunlap, & Collins, 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 (Hagen, Kufoalor, Brekke, & Johansen, 2018; Johansen, Perez, & Cristofaro, 2016; Švec et al., 2013) and conventional gradient-based search algorithms (Abdelaal & Hahn, 2016; Eriksen & 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 & Breivik, 2017b). This makes sample-based approaches well suited for short-term COLAV. In Benjamin, Leonard, Curcio, and Newman (2006, 2010), a protocol-based COLAV algorithm using interval programming is presented. The algorithm optimizes over multiple functions considering different behaviors, for example, 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 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

- **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.
- **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 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.
- **14:** In a head-on situation, where two vessels approaches each other on reciprocal or nearly reciprocal courses (a margin of $\pm$ 6° is often used), both vessels are required to do starboard maneuvers and pass the other vessel on her port side.
- **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.
- **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.
- **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 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.

Figure 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.

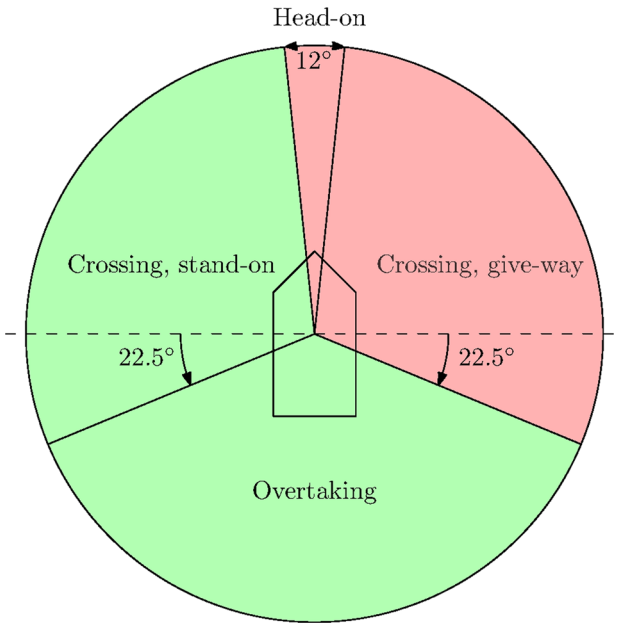

FIGURE 2 Graphical illustration of COLREGs regions as seen from the ownship. The light red regions show areas where the ownship is required to maneuver, while the light green regions show areas where the ownship should keep the current speed and course [Color figure can be viewed at wileyonlinelibrary.com]

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, Breivik, Pettersen, & Wiig, 2016). This algorithm was adapted for use with high-speed ASVs, and tested in conjunction with a radar-based tracking system (Wilthil, Flåten, & Brekke, 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 underactuated 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 multiobstacle 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 study is the Telemetron ASV shown in Figure 3, which is owned and operated by Maritime Robotics. The vessel is 8.45 m long, and uses a single steerable outboard engine for propulsion, which makes the vessel underactuated.

FIGURE 3 The Telemetron ASV, designed for both manned and unmanned operations. Courtesy of Maritime Robotics [Color figure can be viewed at wileyonlinelibrary.com]

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 planning region. In between the displacement and planning region, we have the semidisplacement region. The Telemetron ASV is a high-speed vessel, capable of speeds up to 18 m/s, which combined with the vessel length of 8.45 m makes for a vessel operating in the displacement, semidisplacement, and planning regions (Faltinsen, 2005; Fossen, 2011).

The conventional approach to modeling ASVs is by using the three-degree-of-freedom (3DOF) model (Fossen, 2011):

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

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

where $\boldsymbol{\eta} = [N \;\; E \;\; \psi]^\intercal$ is the vessel pose in an earth-fixed North-East–Down reference frame, $\boldsymbol{\nu} = [u \;\; v \;\; r]^\intercal$ is the vessel velocity, and $\boldsymbol{\tau} = [X \;\; Y \;\; N]^\intercal$ is a vector of forces and torque, both given in the body-fixed reference frame. See Figure 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 m/s (Eriksen & Breivik, 2017a). This is quite a big limitation, and we therefore rather use a control-oriented nonfirst principles model developed for high-speed ASVs (Eriksen & Breivik, 2017a), valid for the displacement, semidisplacement, and planning regions:

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

where $\boldsymbol{x} = [U \;\; r]^\intercal$ 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 $\boldsymbol{\tau} = [\tau_m \;\; \tau_\delta]^\intercal$ 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, while $\boldsymbol{\sigma}(\boldsymbol{x}) = [\sigma_U(\boldsymbol{x}) \;\; \sigma_r(\boldsymbol{x})]^\intercal$ 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 & Breivik, 2017a).

2.2 | ASV control design

As shown in Figure 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 & 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 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 & Breivik, 2017a; Eriksen et al., 2018), 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 $\boldsymbol{x}_d = [U_d \;\; r_d]^\intercal$, $\boldsymbol{K}_p > 0$ 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 Υ: $\mathbb{R} \to S^1$ maps an angle to the domain $[-\pi, \pi)$.

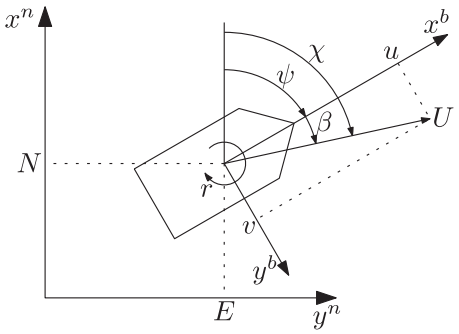

FIGURE 4 Vessel variables. The superscripts $(\cdot)^n$ and $(\cdot)^b$ denote the NED and body reference frames (Fossen, 2011), respectively. The variables N, E, and ψ represent the vessel pose, u, v, and r represent the body-fixed vessel velocity and U is the vessel speed over ground. The course χ is the sum of the heading ψ and the sideslip β.

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 to ensure feasible maneuvers, which is ideal for short-term COLAV. The algorithm is based on 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, for example, 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 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 Figure 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 subtrajectories 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 subtrajectories, each containing one maneuver, are computed given an initial vessel configuration, initial time, and some step-specific parameters:

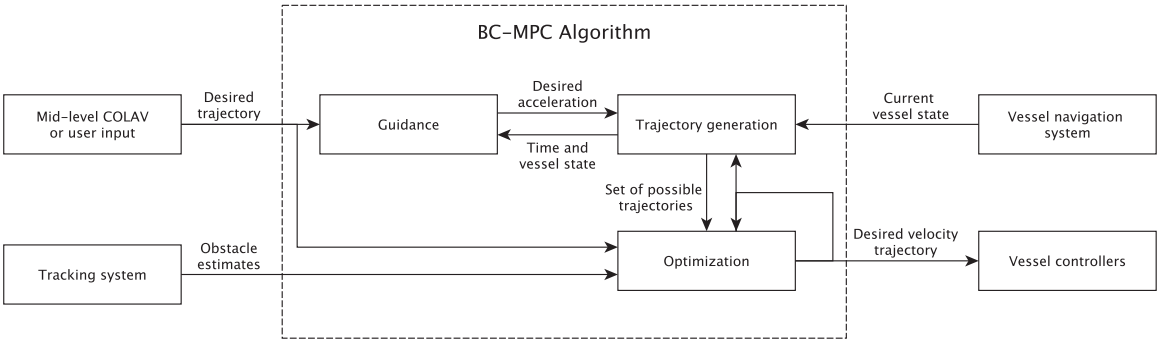

FIGURE 5 BC-MPC algorithm overview. The algorithm inputs a desired trajectory from a mid-level COLAV algorithm or an operator, obstacle estimates from a tracking system, and the current vessel state from a navigation system, and outputs a desired velocity trajectory for the vessel controllers

- 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

$$\begin{aligned}\dot{\boldsymbol{X}}_{\max} &= \boldsymbol{M}^{-1}(\boldsymbol{\tau}_{\max} - \boldsymbol{\sigma}(\boldsymbol{X}_0)),\\ \dot{\boldsymbol{X}}_{\min} &= \boldsymbol{M}^{-1}(\boldsymbol{\tau}_{\min} - \boldsymbol{\sigma}(\boldsymbol{X}_0)),\end{aligned} \tag{7}$$

where $\dot{\boldsymbol{X}}_{\max} = [\dot{U}_{\max} \quad \dot{r}_{\max}]^\intercal$, $\dot{\boldsymbol{X}}_{\min} = [\dot{U}_{\min} \quad \dot{r}_{\min}]^\intercal$, $\boldsymbol{X}_0$ is the current vessel velocity, and

$$\begin{aligned}\boldsymbol{\tau}_{\max} &= \text{sat}(\boldsymbol{\tau}_0 + T_{\text{ramp}}\dot{\boldsymbol{\tau}}_{\max}, \boldsymbol{\tau}_{\min}, \boldsymbol{\tau}_{\max}),\\ \boldsymbol{\tau}_{\min} &= \text{sat}(\boldsymbol{\tau}_0 + T_{\text{ramp}}\dot{\boldsymbol{\tau}}_{\min}, \boldsymbol{\tau}_{\min}, \boldsymbol{\tau}_{\max}),\end{aligned} \tag{8}$$

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 $\text{sat}(\boldsymbol{a}, \boldsymbol{a}_{\min}, \boldsymbol{a}_{\max})$ 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 and $(\cdot)_i$ denoting element i of a vector. Following this, we create a set of possible accelerations as

$$A_d = \{(\dot{U}, \dot{r}) \in \mathbb{R} \times \mathbb{R} \;\big|\; \dot{U} \in [\dot{U}_{\min}, \dot{U}_{\max}], \dot{r} \in [\dot{r}_{\min}, \dot{r}_{\max}]\}. \tag{10}$$

The set of possible accelerations is then sampled uniformly to create a discrete set of candidate maneuver accelerations:

$$\begin{aligned}\dot{\boldsymbol{U}}_{\text{samples}} &= \{\dot{U}_1, \dot{U}_2, ..., \dot{U}_{N_U}\},\\ \dot{\boldsymbol{r}}_{\text{samples}} &= \{\dot{r}_1, \dot{r}_2, ..., \dot{r}_{N_\chi}\},\end{aligned} \tag{11}$$

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, for example, 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 $\dot{\boldsymbol{U}}_{\text{samples}} \times \dot{\boldsymbol{r}}_{\text{samples}}$. This concept is illustrated in Figure 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:

$$\dot{U}_{d,i}(t) = \begin{cases} k_{U,i}t, & 0 \le t < T_{\text{ramp}},\\ \dot{U}_i, & T_{\text{ramp}} \le t < T_U - T_{\text{ramp}},\\ \dot{U}_i - k_{U,i}(t - (T_U - T_{\text{ramp}})), & T_U - T_{\text{ramp}} \le t < T_U,\\ 0, & T_U \le t \le T,\end{cases} \tag{12}$$

where $k_{U,i} = (\dot{U}_i / T_{\text{ramp}})$, $\dot{U}_i$ 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:

$$\dot{r}_{d,i}(t) = \begin{cases} k_{r,i}t, & 0 \le t < T_{\text{ramp}},\\ 2\dot{r}_i - k_{r,i}t, & T_{\text{ramp}} \le t < 2T_{\text{ramp}},\\ 0, & 2T_{\text{ramp}} \le t < T_\chi - 2T_{\text{ramp}},\\ -k_{r,i}(t - (T_\chi - 2T_{\text{ramp}})), & T_\chi - 2T_{\text{ramp}} \le t < T_\chi - T_{\text{ramp}},\\ -2\dot{r}_i + k_{r,i}(t - (T_\chi - T_{\text{ramp}})), & T_\chi - T_{\text{ramp}} \le t < T_\chi,\\ 0, & 2T_\chi \le t < T,\end{cases} \tag{13}$$

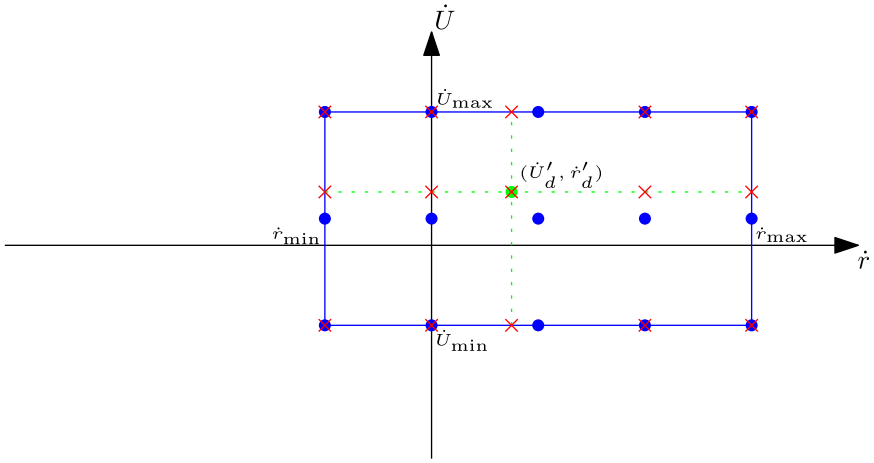

FIGURE 6 Set of possible accelerations shown with the blue line, with initial samples shown as blue circles. The desired acceleration $(\dot{U}'_d, \dot{r}'_d)$ is shown as a green circle, while the final samples are shown as red crosses [Color figure can be viewed at wileyonlinelibrary.com]

where $k_{r,i} = (\dot{r}_i/T_{\text{ramp}})$, $\dot{r}_i$ 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 Figure 7.

Notice that the integral of the course acceleration maneuvers is 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 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 the expressions (12) and (13) as

$$\begin{aligned} U_{d,i}(t) &= U_{d,0} + \int_{t_0}^{t} \dot{U}_{d,i}(\gamma)\mathrm{d}\gamma, \; i \in [1, N_U], \\ r_{d,i}(t) &= r_{d,0} + \int_{t_0}^{t} \dot{r}_{d,i}(\gamma)\mathrm{d}\gamma, \quad i \in [1, N_\chi], \\ \chi_{d,i}(t) &= \chi_{d,0} + \int_{t_0}^{t} r_{d,i}(\gamma)\mathrm{d}\gamma, \; i \in [1, N_\chi]. \end{aligned} \tag{14}$$

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 subtrajectory, 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 five speed and five 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

$$\mathcal{U}_d = \left\{U_{d,1}(t), U_{d,2}(t), \ldots, U_{d,N_U}(t)\right\} \times \left\{\chi_{d,1}(t), \chi_{d,2}(t), \ldots, \chi_{d,N_\chi}(t)\right\}, \tag{15}$$

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.

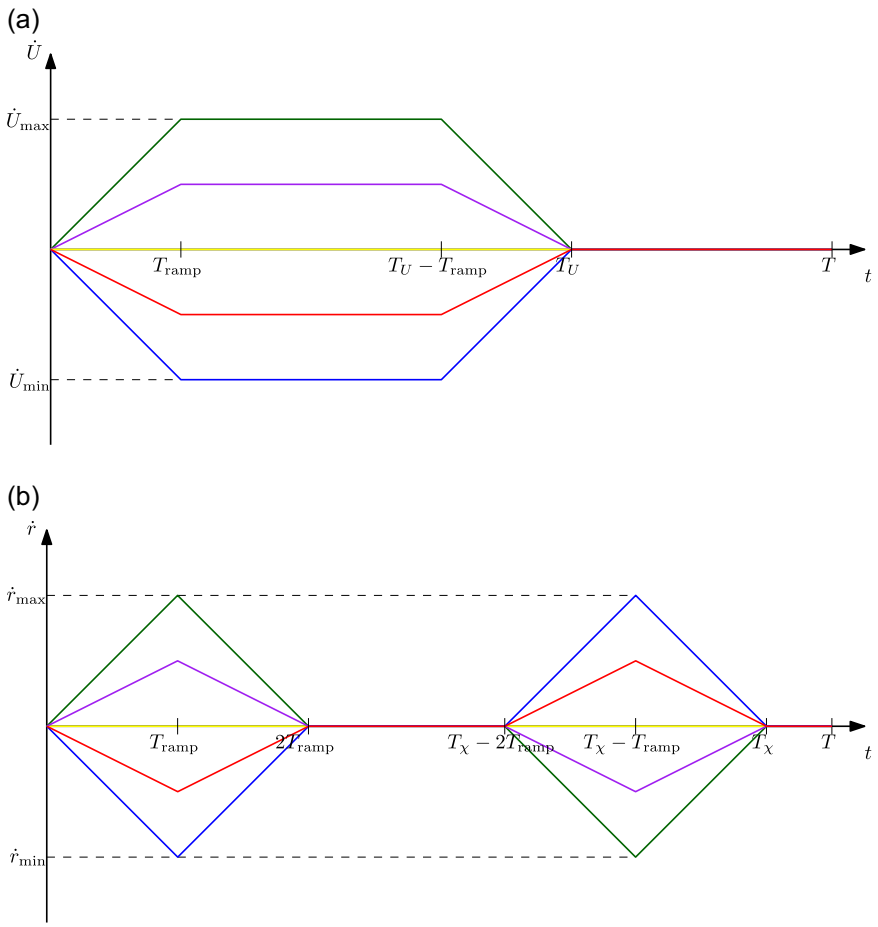

FIGURE 7 Acceleration motion primitives, where T is the step time, T_{ramp} denotes the ramp time, while T_U and T_χ are the speed and course maneuver time lengths, respectively. (a) Speed acceleration motion primitives and (b) course acceleration motion primitives. Note that the integral of each course acceleration trajectory is zero [Color figure can be viewed at wileyonlinelibrary.com]

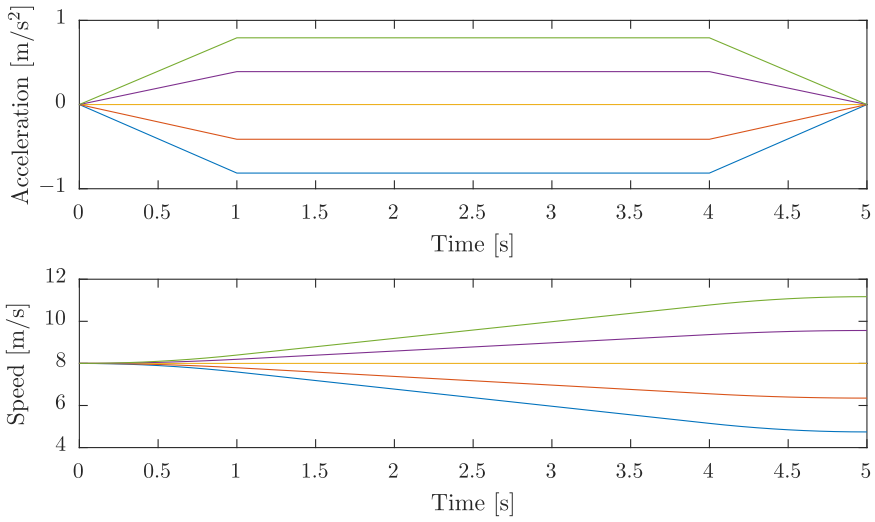

FIGURE 8 Example of $N_U = 5$ speed trajectories with $T_{\text{ramp}} = 1\,\text{s}$ and $T = T_U = 5\,\text{s}$. Acceleration in the top plot and speed in the bottom plot [Color figure can be viewed at wileyonlinelibrary.com]

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 & 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

$$\dot{\tilde{U}} = \frac{1}{T_{\tilde{U}}}\tilde{U}, \dot{\tilde{\chi}} = \frac{1}{T_{\tilde{\chi}}}\tilde{\chi}, \tag{16}$$

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

$$\begin{aligned} \bar{U}_i(t) &= \tilde{U}_0 e^{-(1/T_{\tilde{U}})(t-t_0)} + U_{d,i}(t), \quad i \in [1, N_U], \\ \bar{\chi}_i(t) &= \tilde{\chi}_0 e^{-(1/T_{\tilde{\chi}})(t-t_0)} + \chi_{d,i}(t), \quad i \in [1, N_\chi], \end{aligned} \tag{17}$$

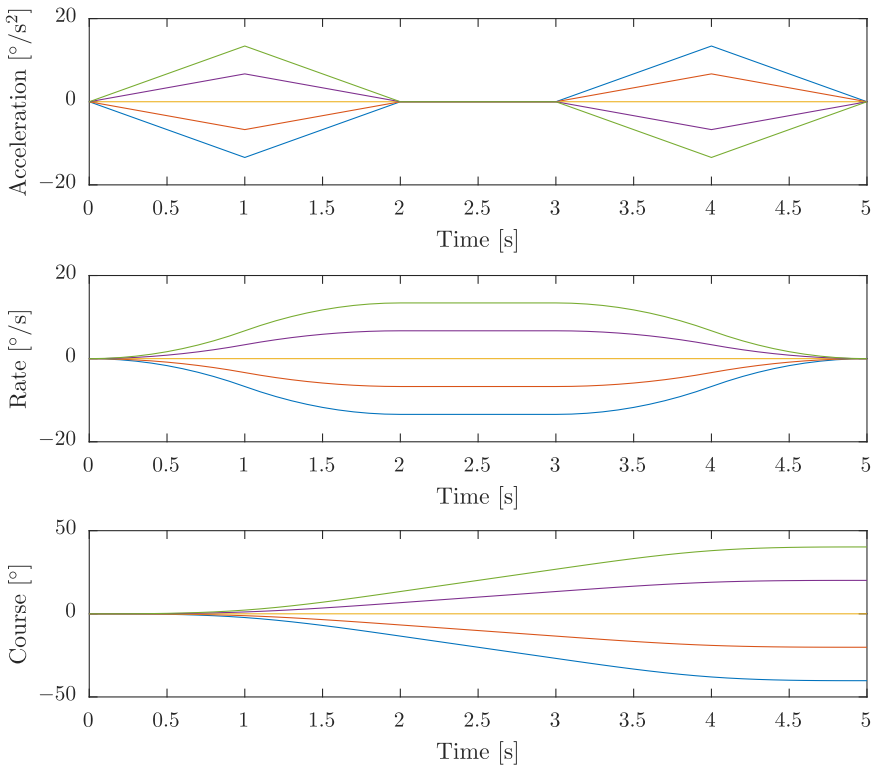

FIGURE 9 Example of $N_\chi = 5$ course trajectories with $T_{\text{ramp}} = 1\,\text{s}$ and $T = T_\chi = 5\,\text{s}$. Acceleration in the top plot, rate in the middle and course in the bottom plot [Color figure can be viewed at wileyonlinelibrary.com]

where $\tilde{U}_0 = U_0 - U_{d,0}$ and $\tilde{\chi}_0 = \Upsilon(\chi_0 - \chi_{d,0})$ 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:

$$\bar{\mathcal{U}} = \{\bar{U}_1(t), \bar{U}_2(t), ..., \bar{U}_{N_U}(t)\} \times \{\bar{\chi}_1(t), \bar{\chi}_2(t), ..., \bar{\chi}_{N_\chi}(t)\}. \quad (18)$$

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 $\bar{\boldsymbol{p}}(t) = [\bar{N}(t) \quad \bar{E}(t)]^T$ are calculated from the predicted velocity trajectories using a kinematic model:

$$\dot{\bar{\boldsymbol{p}}} = \begin{bmatrix} \cos(\bar{\chi}) \\ \sin(\bar{\chi}) \end{bmatrix} \bar{U}, \quad (19)$$

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

$$\bar{\mathcal{H}} = \{\bar{\boldsymbol{\eta}}(t; \bar{U}(t), \bar{\chi}(t)) | (\bar{U}(t), \bar{\chi}(t)) \in \bar{\mathcal{U}}\}, \quad (20)$$

where $\bar{\boldsymbol{\eta}} = [\bar{N}(t) \quad \bar{E}(t) \quad \bar{\chi}(t)]^T$.

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 subtrajectories, 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 subtrajectories. The depth of the tree will be equal to the desired number of maneuvers in each trajectory. The tree is initialized with the initial state as the root node, which the single-step procedure is performed on, generating a number of subtrajectories and leaf nodes. Following this, the single-step procedure is performed on each of the leaf nodes, adding the next subtrajectory 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 $\boldsymbol{T} = [T_1 \quad T_2 \quad ... \quad T_B]$, 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 $\boldsymbol{N}_U = [N_{U,1} \quad N_{U,2} \quad ... \quad N_{U,B}]$.
- The number of course maneuvers at each level $\boldsymbol{N}_\chi = [N_{\chi,1} \quad N_{\chi,2} \quad ... \quad N_{\chi,B}]$.

A set of predicted vessel pose trajectories with $B = 3$ levels is shown in Figure 10. The step time is chosen as $\boldsymbol{T} = [5\,\text{s} \quad 10\,\text{s} \quad 10\,\text{s}]$, making the trajectories 25 s long in total. The trajectories have five course maneuvers at the first level and three in the later levels, while there for illustrational purposes only is one speed maneuver at each step. Hence, $\boldsymbol{N}_U = [1 \quad 1 \quad 1]$ and $\boldsymbol{N}_\chi = [5 \quad 3 \quad 3]$. The ramp time and maneuver lengths are chosen as $T_{\text{ramp}} = 1\,\text{s}$ and $T_U = T_\chi = 5\,\text{s}$, 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

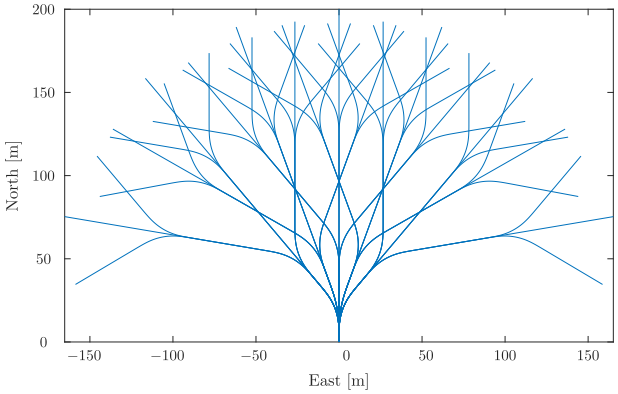

FIGURE 10 A set of predicted pose trajectories with three levels [Color figure can be viewed at wileyonlinelibrary.com]

maneuvering out from a desired trajectory, turning parallel to the trajectory and returning toward the trajectory. The step time of each maneuver $\boldsymbol{T}$ controls the length of the prediction 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 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 toward 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 & 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

$$\chi_{d,\text{LOS}} = \chi_{\text{path}} + \arctan\left(-\frac{e}{\Delta}\right), \tag{21}$$

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 PP velocity along the path is stated as

$$U_{\text{PP}} = U\cos(\chi - \chi_{\text{path}}) + \gamma_s s, \tag{22}$$

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 Figure 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 PP, letting the vessel converge toward 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 PP at the desired position on the trajectory, given the current time, and by reformulating (22) we obtain a desired vessel speed given the trajectory velocity:

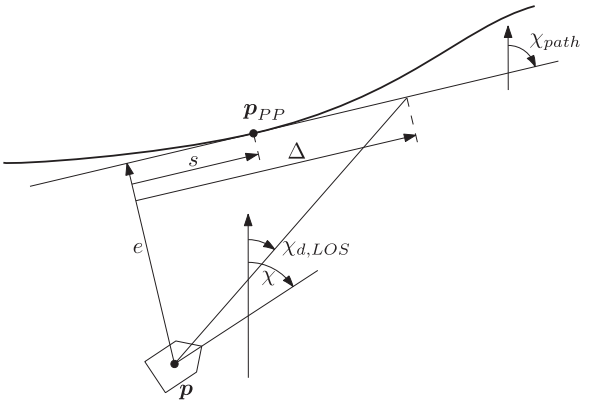

FIGURE 11 LOS guidance scheme. The path particle propagates along the path with the speed U_{pp}. The vessel course is denoted as χ, while χ_{path} denotes the current tangential path course and $\chi_{d,\text{LOS}}$ denote the desired course. The variables e, s, and Δ are the cross-track error, the along-track distance, and the lookahead distance, respectively

$$U_{d,\text{LOS}} = \begin{cases} \text{sat}\left(\dfrac{U_t - \gamma_s s}{\cos(\chi - \chi_{\text{path}})}, 0, U_{\text{max,LOS}}\right) & \text{if } |\cos(\chi - \chi_{\text{path}})| > \epsilon, \\ \text{sat}\left(\dfrac{U_{\text{pp}} - \gamma_s s}{\epsilon}, 0, U_{\text{max,LOS}}\right) & \text{else,} \end{cases} \tag{23}$$

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 $[0, U_{\text{max}}]$, where $U_{\text{max}} > 0$ is the maximum vessel operating speed. Given a desired speed and course, we compute the desired speed and course acceleration:

$$\begin{aligned} \dot{U}'_d &= \frac{U_{d,\text{LOS}} - U_{d,0}}{T_U - T_{\text{ramp}}}, \\ \dot{r}'_d &= \frac{\chi_{d,\text{LOS}} - \chi_{d,0}}{T_{\text{ramp}}(T_\chi - 2T_{\text{ramp}})}, \end{aligned} \tag{24}$$

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 $\cos(\chi - \chi_{\text{path}} = 0)$ 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 & Breivik, 2017a). Hence, the simplicity of (21), (22), and (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 $\text{tran}(\boldsymbol{u}_d(t))$ 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 $w_{\text{av}} \gg w_{\text{al}}$ 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:

$$\boldsymbol{u}_d^*(t) = \underset{(\bar{\boldsymbol{\eta}}_k(t), \boldsymbol{u}_{d,k}(t)) \in (\mathcal{H}, \mathcal{U}_d)}{\text{argmin}} G(\bar{\boldsymbol{\eta}}_k(t), \boldsymbol{u}_{d,k}(t); \boldsymbol{p}_d(t)), \tag{26}$$

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 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 $\boldsymbol{p}_d(t) = [N_d(t) \quad E_d(t)]^\intercal$. Given this, we define a weighted metric of Euclidean distance and orientation error as

$$\text{align}(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_d(t)) = \int_{t_0}^{t_0+T_{\text{full}}} \left(w_p \left\| \begin{bmatrix} \bar{N}(\gamma) \\ \bar{E}(\gamma) \end{bmatrix} - \boldsymbol{p}_d(\gamma) \right\|_2 + w_\chi |\Upsilon(\bar{\chi}(\gamma) - \chi_d(\gamma))| \right) \mathrm{d}\gamma, \tag{28}$$

where $w_p, w_\chi > 0$ are weights controlling the influence of the Euclidean and angular error, respectively, while $T_{\text{full}} = \sum_{i=1}^{B} T_i$ 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, namely, the collision, safety, and margin regions. The idea behind this is to make it possible to 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 $\boldsymbol{r}_i = [r_{N,i} \quad r_{E,i}]^\intercal$ 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 (Eriksen et al., 2018; Johansen et al., 2016; Kuwata et al., 2014), 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 AIS data (Dalsnes, Hexeberg, Flåten, Eriksen, & Brekke, 2018) or by estimating the turn rate of the obstacles (Flåten & Brekke, 2017). Using (29), we define the distance and relative bearing to obstacle i given a predicted vessel trajectory $\bar{\boldsymbol{\eta}}(t)$ as

$$\begin{aligned} d_i(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_i(t)) &= \|\boldsymbol{r}_i(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_i(t))\|_2, \\ \beta_i(\bar{\boldsymbol{\eta}}(t); \boldsymbol{p}_i(t)) &= \Upsilon(\text{atan2}(r_{E,i}(\bar{\boldsymbol{\eta}}(t)), r_{N,i}(\bar{\boldsymbol{\eta}}(t))) - \chi_i(t)), \end{aligned} \tag{30}$$

where $\chi_i(t)$ is the course of obstacle i, calculated as $\chi_i(t) = \text{atan2}(\dot{E}_i(t), \dot{N}_i(t))$ with $\boldsymbol{p}_i(t) = [N_i(t) \quad E_i(t)]^\intercal$. The distance d_i and relative bearing β_i are illustrated in Figure 12.

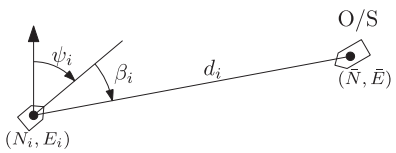

FIGURE 12 Distance d_i and relative bearing β_i to obstacle i. The ownship is marked O/S

The obstacle distance and relative bearing is used to calculate a penalty function, which we use to define the avoidance function as

$$\text{avoid}(\bar{\boldsymbol{\eta}}(t)) = \sum_{i=1}^{M} \int_{t=t_0}^{t_0+T_{\text{full}}} w_i(\gamma)\text{penalty}_i(\bar{\boldsymbol{\eta}}(\gamma))\mathrm{d}\gamma, \tag{31}$$

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 multiobstacle situations where properties like vessel type, size, speed, and so forth 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

$$\text{penalty}_{i,\text{circular}}(\bar{\boldsymbol{\eta}}(t)) = \begin{cases} 1 & \text{if } d_i < D_0, \\ 1 + \dfrac{\gamma_1 - 1}{D_1 - D_0}(d_i - D_0) & \text{if } D_0 \le d_i < D_1, \\ \gamma_1 - \dfrac{\gamma_1}{D_2 - D_1}(d_i - D_1) & \text{if } D_1 \le d_i < D_2, \\ 0 & \text{else,} \end{cases} \tag{32}$$

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 Figure 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 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 toward 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 revokes the stand-on requirement. However, the algorithm should choose maneuvers compliant with Rule 13, and Rules 14 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 & Szlapczynska, 2017). Each region is defined by a combination of three elliptical and one circular segment as

$$D_k(\beta_i) = \begin{cases} b_k & \text{if } \beta_i < -\dfrac{\pi}{2}, \\ \dfrac{a_k b_k}{\sqrt{(b_k \cos\beta_i)^2 + (a_k \sin\beta_i)^2}} & \text{if } -\dfrac{\pi}{2} \le \beta_i < 0, \\ \dfrac{a_k c_k}{\sqrt{(c_k \cos\beta_i)^2 + (a_k \sin\beta_i)^2}} & \text{if } 0 \le \beta_i < \dfrac{\pi}{2}, \\ \dfrac{b_k c_k}{\sqrt{(c_k \cos\beta_i)^2 + (b_k \sin\beta_i)^2}} & \text{if } \dfrac{\pi}{2} \le \beta_i, \end{cases} \tag{33}$$

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.

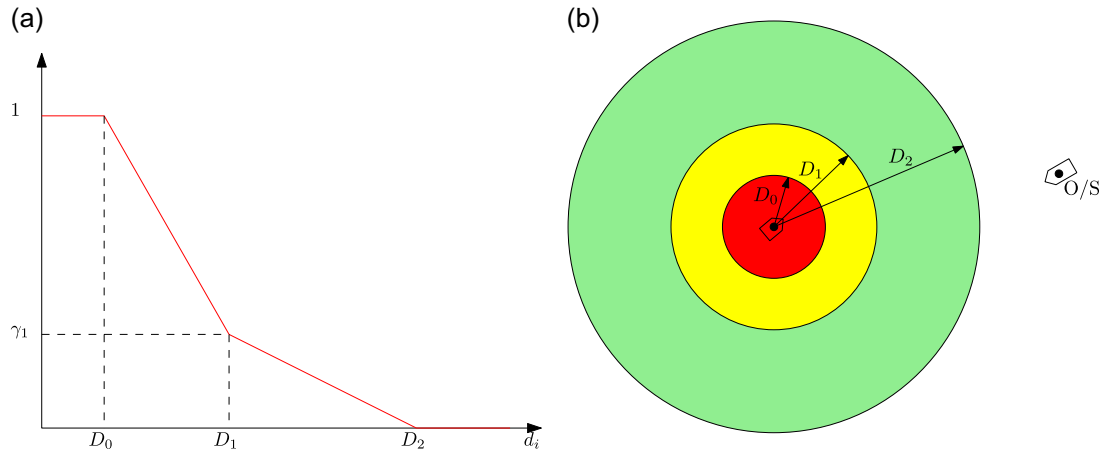

FIGURE 13 The circular penalty function value and regions. (a) Function value and (b) function regions. The red region is the collision region, yellow is the safety region, and green is the margin region, given by the radiuses D_0, D_1, and D_2, respectively [Color figure can be viewed at wileyonlinelibrary.com]

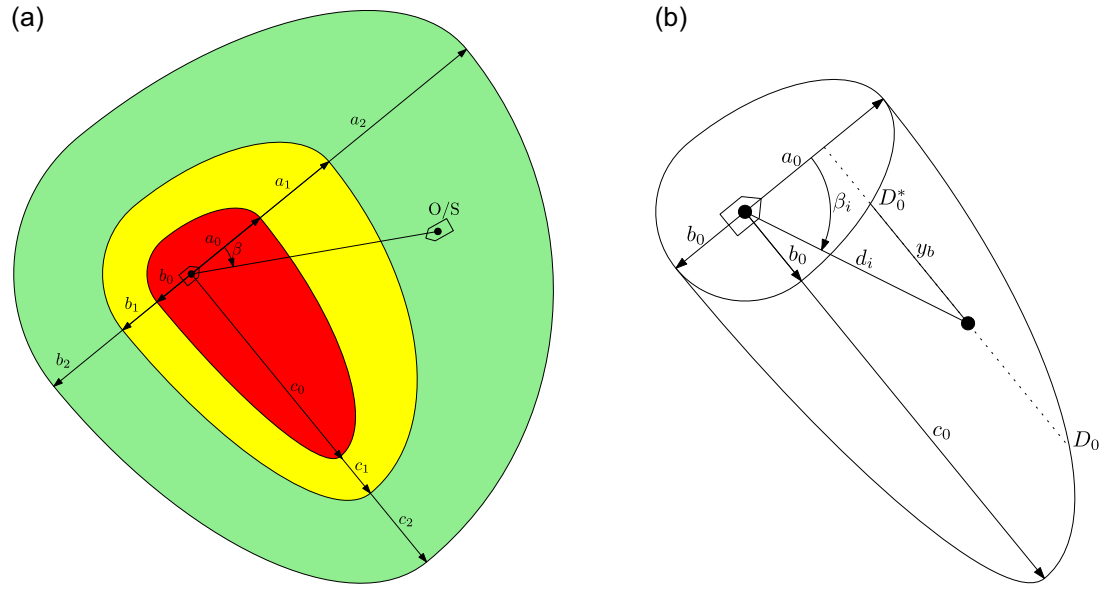

FIGURE 14 Elliptical COLREGs penalty function regions and illustration of y_b for the inner penalty function. (a) Function regions, each constructed by one circular and three elliptical segments. At a given distance from the obstacle, the elliptical form imposes a higher cost on the starboard side and in front of an obstacle, compared to the port side 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 port-side collision region boundary mirrored about the obstacle surge axis [Color figure can be viewed at wileyonlinelibrary.com]

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 nonconstant cost inside the collision region since this is rather large. We therefore define the elliptical COLREGs penalty function as

$$\begin{aligned}\text{penalty}_{i,\text{COLREGs}}(\bar{\boldsymbol{\eta}}(t)) &= \text{inner\_penalty}_i(\bar{\boldsymbol{\eta}}(t)) \\ &+ \begin{cases} 1 & \text{if } d_i < D_0, \\ 1 + \dfrac{\gamma_1 - 1}{D_1 - D_0}(d_i - D_0) & \text{if } D_0 \le d_i < D_1, \\ \gamma_1 - \dfrac{\gamma_1}{D_2 - D_1}(d_i - D_1) & \text{if } D_1 \le d_i < D_2, \\ 0 & \text{else,} \end{cases}\end{aligned} \tag{34}$$

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

$$\text{inner\_penalty}_i(\bar{\boldsymbol{\eta}}(t)) = \begin{cases} 1 & \text{if } d_i < D_0^*, \\ 1 - \dfrac{y_b(d_i, \beta_i)}{d_{\text{COLREGs}}} & \text{if } D_0^* \le d_i < D_0, \\ 0 & \text{else,} \end{cases} \tag{35}$$

where D_0^* is given as

$$D_0^*(\beta_i) = \begin{cases} \dfrac{a_0 b_0}{\sqrt{(b_0 \cos \beta_i)^2 + (a_0 \sin \beta_i)^2}} & \text{if } |\beta_i| < \dfrac{\pi}{2}, \\ b_0 & \text{else,} \end{cases} \tag{36}$$

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 toward 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

$$\text{tran}(\boldsymbol{u}_d(t)) = \begin{cases} 1 \quad \text{if } \int_{t_0}^{t_0+T_1} |U_d(\gamma) - U_d^-(\gamma)| \, \mathrm{d}\gamma \\ \qquad > e_{U,\min} \text{ or } \int_{t_0}^{t_0+T_1} |\chi_d(\gamma) - \chi_d^-(\gamma)| \, \mathrm{d}\gamma > e_{\chi,\min}, \\ 0 \quad \text{else}, \end{cases} \tag{37}$$

with $\boldsymbol{u}_d(t) = [U_d(t) \;\; \chi_d(t)]^\intercal$, $\boldsymbol{u}_d^-(t) = [U_d^-(t) \;\; \chi_d^-(t)]^\intercal$ 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:

$$\begin{aligned} e_{U,\min} &= \min_{\boldsymbol{u}_d(t) \in \mathcal{U}_d} \int_{t_0}^{t_0+T_1} |U_d(\gamma) - U_d^-(\gamma)| \, \mathrm{d}\gamma, \\ e_{\chi,\min} &= \min_{\boldsymbol{u}_d(t) \in \mathcal{U}_d} \int_{t_0}^{t_0+T_1} |\chi_d(\gamma) - \chi_d^-(\gamma)| \, \mathrm{d}\gamma. \end{aligned} \tag{38}$$

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 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 October 12, 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™ 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-

TABLE 1 Telemetron ASV specifications

Component	Description
Vessel hull	Polarcirkel Sport 845
Length	8.45 m
Width	2.71 m
Weight	1,675 kg
Propulsion system	Yamaha 225 HP outboard engine
Motor control	Electromechanical actuation of throttle valve
Rudder control	Hydraulic actuation of outboard engine angle with proportional–derivative 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

Abbreviation: ASV, autonomous surface vehicle.

TABLE 2 BC-MPC algorithm parameters

Parameter	Value	Description
$\boldsymbol{T}$	[5 20 30] s	Prediction horizon
$\boldsymbol{N}_U$	[5 1 1]	Number of speed maneuvers
$\boldsymbol{N}_\chi$	[5 3 3]	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}	6,000	Avoid weight
w_{t}	4,200	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

Abbreviation: BC-MPC, branching-course model predictive control; LOS, line of sight.

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). Figure 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 algorithm 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 interface. The on-board control system has an electromechanical 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 Figure 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 m/s (five 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 (HO). 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 (CS). 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.
4. Crossing from port (CP). 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.

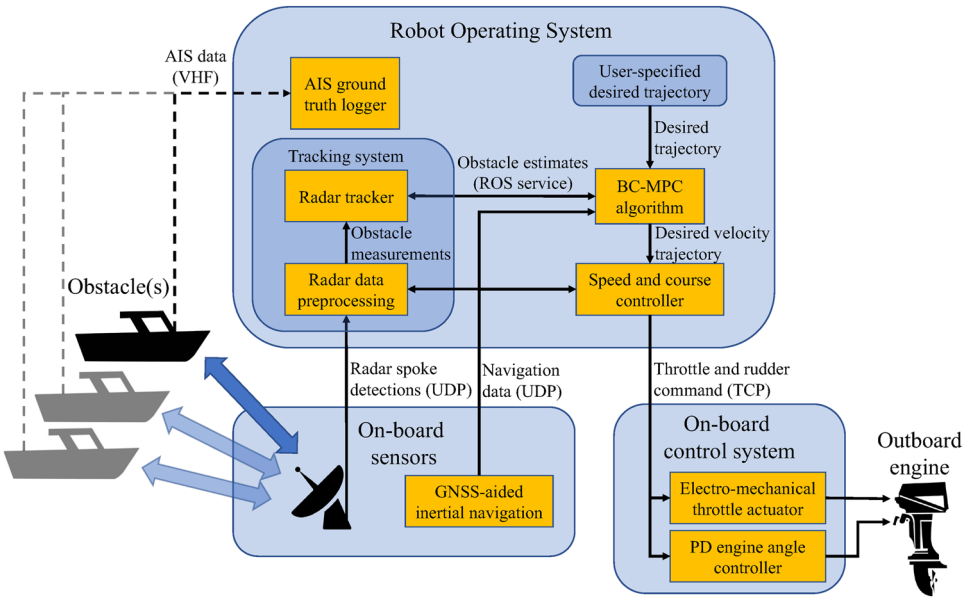

FIGURE 15 Architecture of the COLAV implementation on the Telemetron ASV [Color figure can be viewed at wileyonlinelibrary.com]

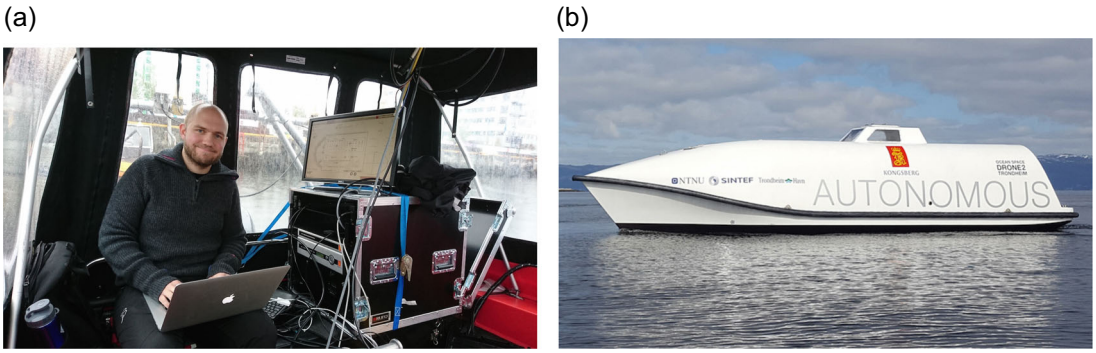

FIGURE 16 The inside of the Telemetron ASV and the Kongsberg Seatex Ocean Space Drone 2. (a) Erik Wilthil in the back of the Telemetron ASV with the navigation system and the processing platform in the rack to the right. (b) The Kongsberg Seatex Ocean Space Drone 2, which is identical to the Ocean Space Drone 1. Courtesy of Kongsberg Seatex [Color figure can be viewed at wileyonlinelibrary.com]

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 Figure 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 nonconventional 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

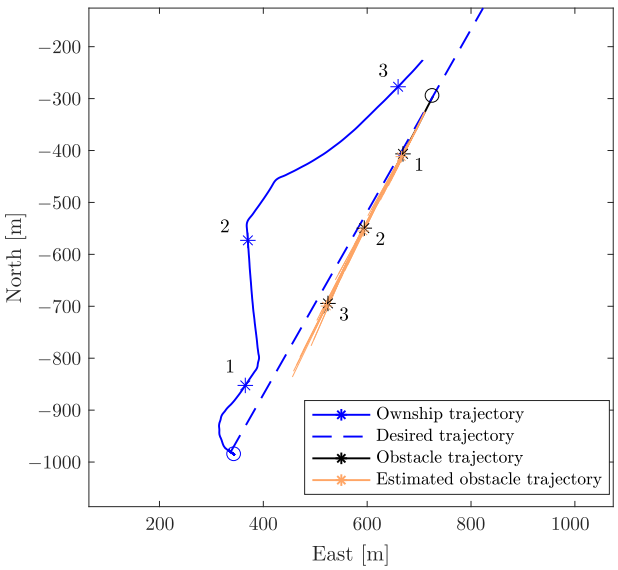

FIGURE 17 Experiment 1.1: Head-on scenario using AIS for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, the estimated obstacle trajectory is shown with the thick orange line, while predicted future trajectory for the obstacle at each timestep is shown 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 [Color figure can be viewed at wileyonlinelibrary.com]

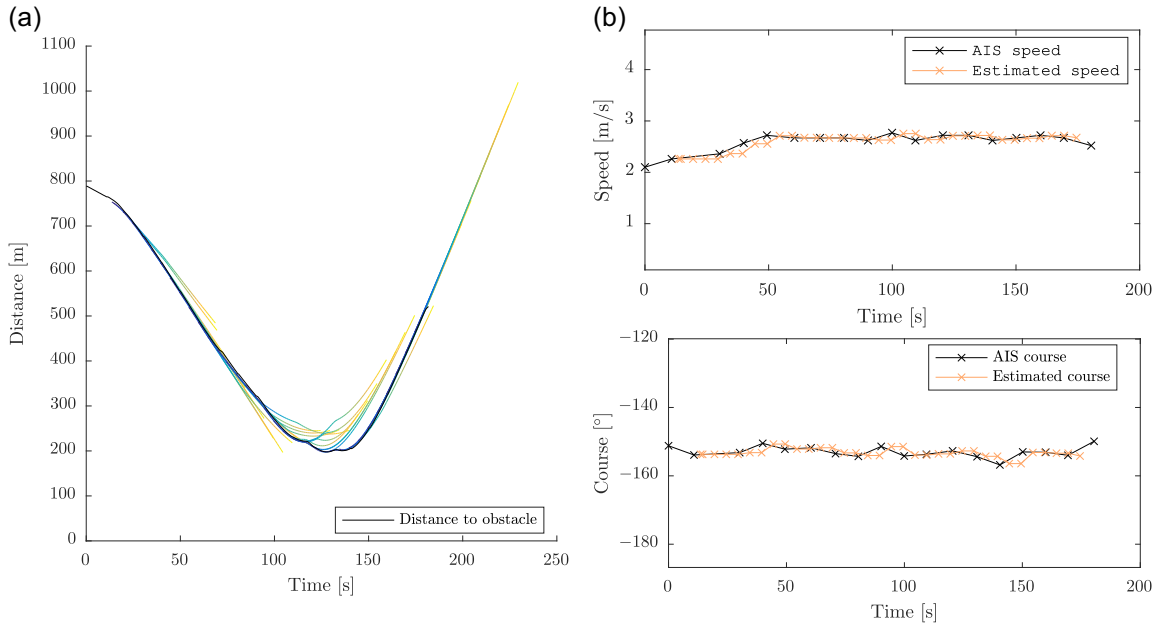

FIGURE 18 Distance to the OSD1 (a) and the estimated speed and course (b) during Experiment 1.1. (a) The black line shows the actual distance between the vessels, while the colored lines show the predicted future distance at each BC-MPC iteration. Blue represents the start of the predictions while yellow represents the end. (b) Speed and course estimates. The black crosses represent received AIS messages, while the orange crosses represent BC-MPC iterations [Color figure can be viewed at wileyonlinelibrary.com]

the ground truth. The OSD1 does, however, transmit AIS messages quite seldom, introducing some delay in the estimated speed and course.

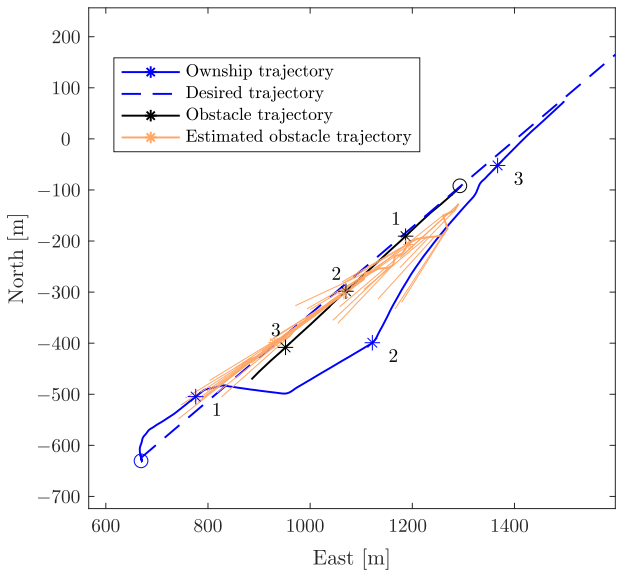

FIGURE 19 Experiment 1.2: Head-on scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, the estimated obstacle trajectory is shown with the thick orange line, while predicted future trajectory for the obstacle at each timestep are shown as the thin orange lines. The numbers represent time markers for each 60 s [Color figure can be viewed at wileyonlinelibrary.com]

Following this experiment, we performed several experiments using the radar-based tracking system for providing obstacle estimates. Figure 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 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 Figure 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 Figure 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 Figure 21, where we approach the OSD1 from northeast. 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 northeast, behind the ownship. The leisure craft did not have AIS, and we do therefore not have a ground-truth trajectory for this vessel. Figure 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

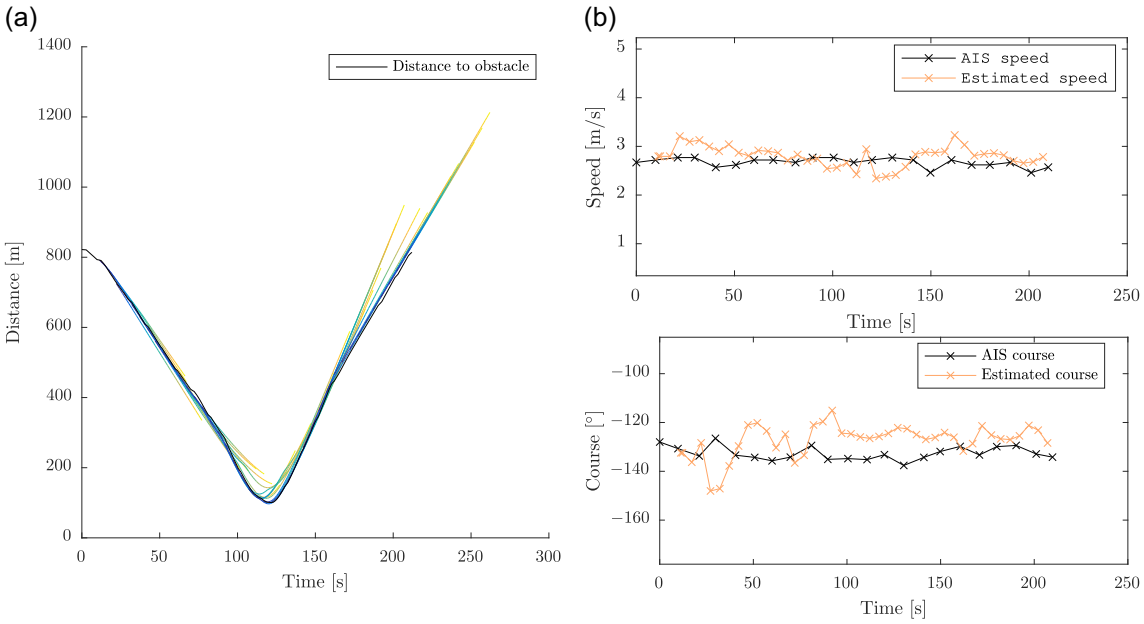

FIGURE 20 Distance to the OSD1 (a), and the estimated speed and course (b) during Experiment 1.2. (a) The black line shows the actual distance between the vessels, while the colored lines show the predicted future distance at each BC-MPC iteration. Blue represents the start of the predictions while yellow represents the end. (b) Speed and course estimates. The black crosses represent received AIS messages, while the orange crosses represent BC-MPC iterations [Color figure can be viewed at wileyonlinelibrary.com]

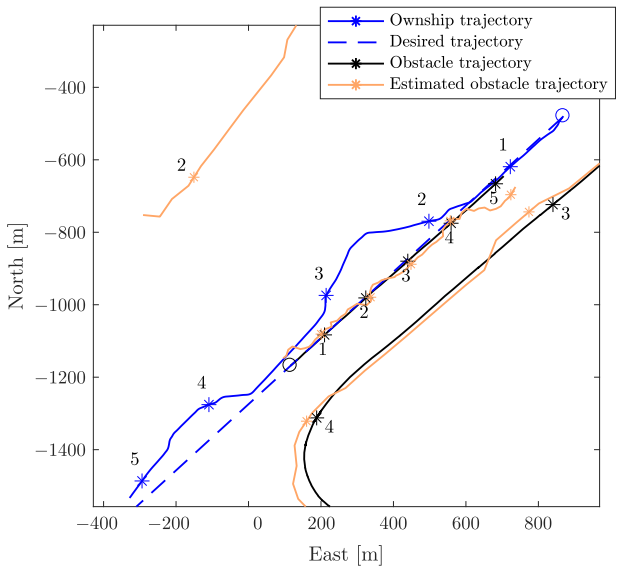

FIGURE 21 Experiment 1.3: Head-on scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 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 toward northeast and did not have AIS, so there is no ground-truth trajectory for this vessel [Color figure can be viewed at wileyonlinelibrary.com]

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

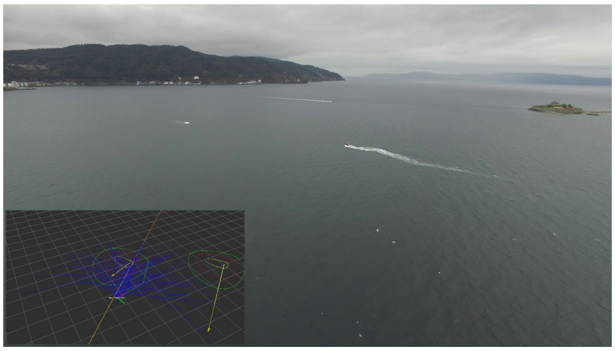

FIGURE 22 Drone picture during Experiment 1.3, approximately at the second time mark. The ownship is located in the middle of the picture, with the OSD1 to the left. The vessel in the background is a high-speed leisure craft. Yellow arrows in the visualization represent the estimated obstacle speed and 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 [Color figure can be viewed at wileyonlinelibrary.com]

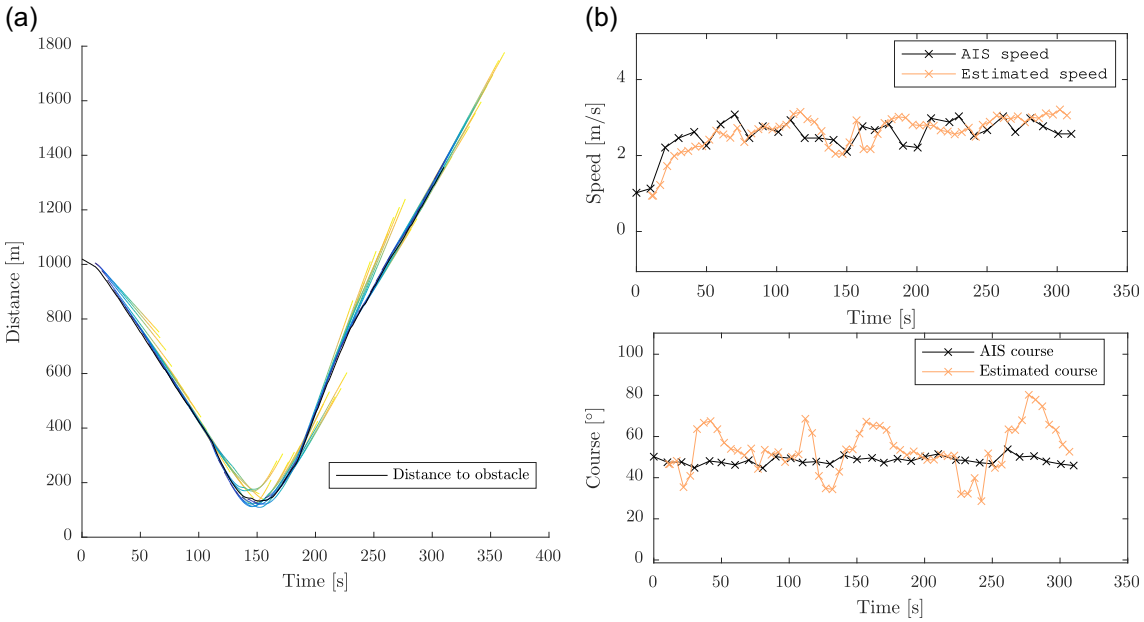

FIGURE 23 Distance to the OSD1 (a), and the estimated speed and course (b) during Experiment 1.3. (a) The black line shows the actual distance between the vessels, while the colored lines show the predicted future distance at each BC-MPC iteration. Blue represents the start of the predictions while yellow represents the end. (b) Speed and course estimates. The black crosses represent received AIS messages, while the orange crosses represent BC-MPC iterations [Color figure can be viewed at wileyonlinelibrary.com]

vessel. Hence, the algorithm chooses to do a new starboard maneuver to let the passenger ferry pass. Eventually, the ferry turns toward 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 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 Figure 23b. Still, as shown in

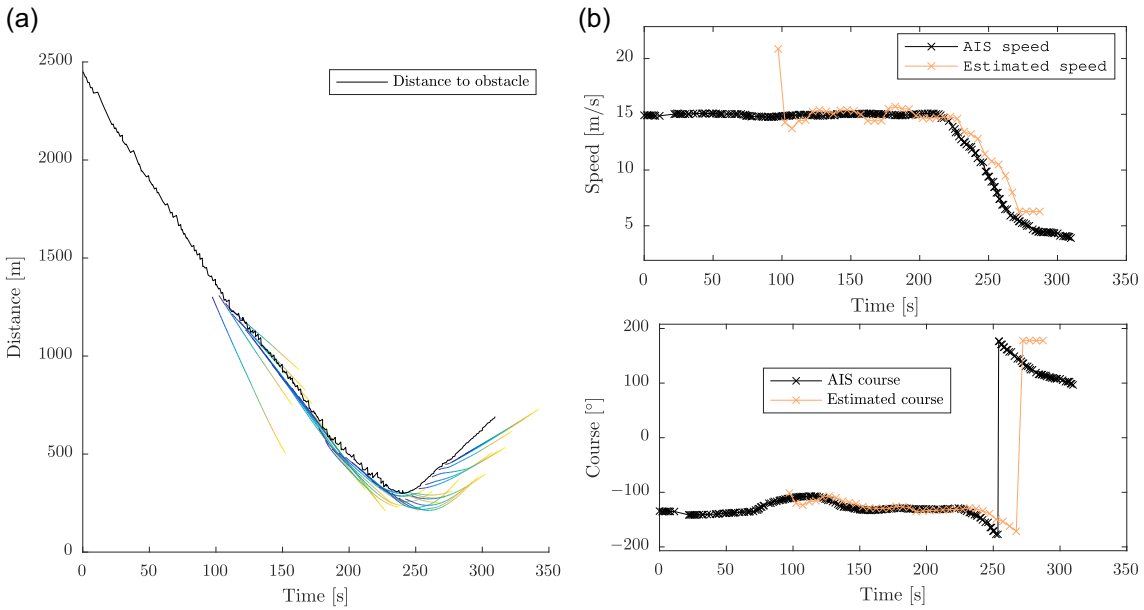

FIGURE 24 Distance to the Trondheimfjord II (a), and the estimated speed and course (b) during Experiment 1.3. (a) The black line shows the actual distance between the vessels, while the colored lines show the predicted future distance at each BC-MPC iteration. Blue represents the start of the predictions 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 [Color figure can be viewed at wileyonlinelibrary.com]

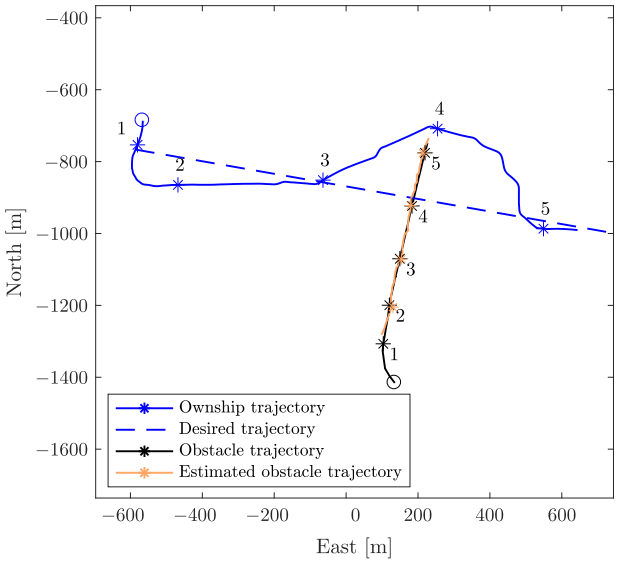

FIGURE 25 Experiment 2.1: Crossing from starboard scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 60 s [Color figure can be viewed at wileyonlinelibrary.com]

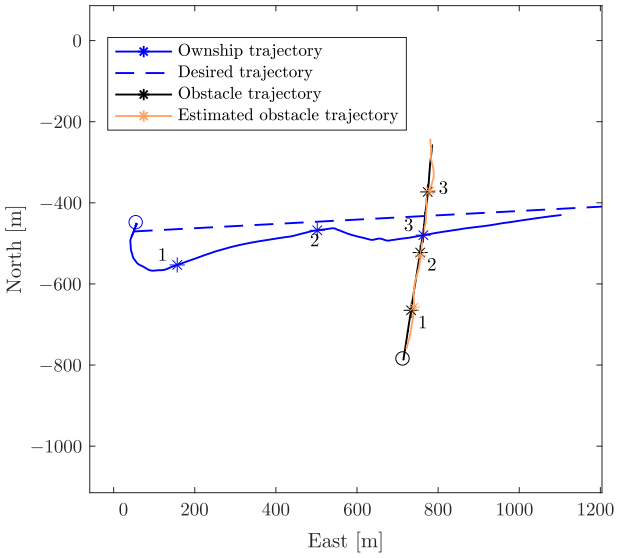

FIGURE 26 Experiment 2.2: Crossing from starboard scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 60 s [Color figure can be viewed at wileyonlinelibrary.com]

Figures 23a and 21, the BC-MPC algorithm manages to make quite smooth maneuvers, which again shows robustness with respect to obstacle estimate noise. Figure 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 Figure 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 Figure 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.

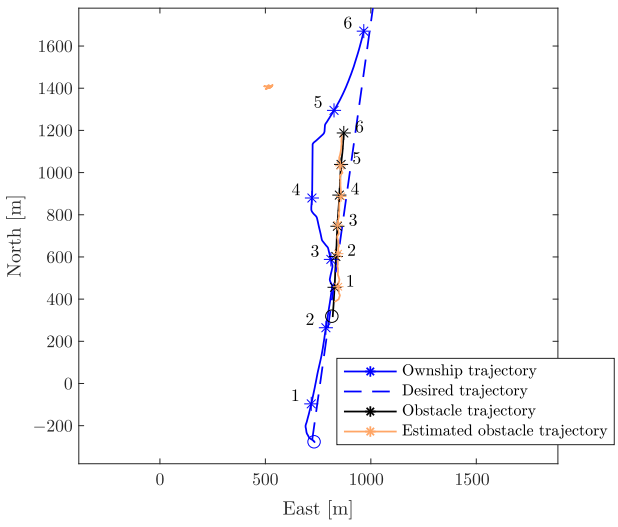

FIGURE 27 Experiment 3: Overtaking scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 60 s. The trajectory located at approximately (1,450, 500) originates from a navigational aid, which was detected approximately 130 s into the experiment [Color figure can be viewed at wileyonlinelibrary.com]

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.

Figure 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 toward 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.

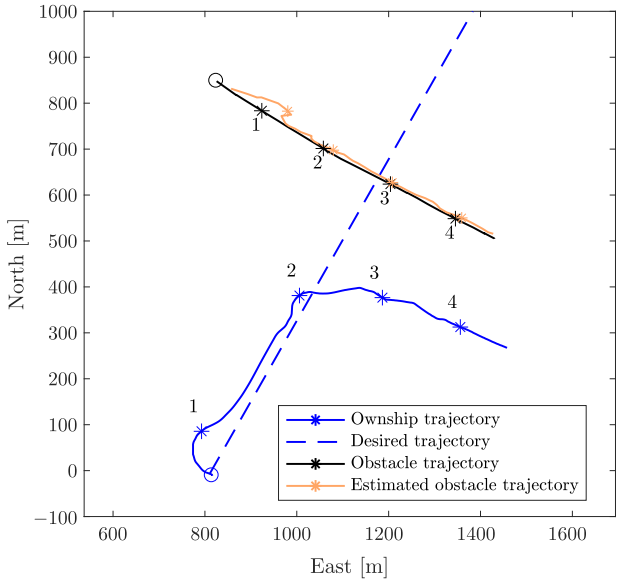

FIGURE 28 Experiment 4: Crossing from port scenario using the radar-based tracking system for providing obstacle estimates. The ownship and obstacle initial positions are marked with circles, and the estimated obstacle trajectory is shown with the thick orange line. The numbers represent time markers for each 60 s [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 3 Key points and numbers from the experiments

Experiment type and number	Obstacle sensor	Rule 13–15 compliance	Minimum distance to obstacle (m)
Head on			
1.1	AIS	No	197.8
1.2	Radar	Yes	100.8
1.3	Radar	Yes	132.5
Crossing from starboard			
2.1	Radar	Yes[a]	214.0
2.2	Radar	Yes	106.2
Overtaking			
3.1	Radar	Yes	127.3
Crossing from port			
4.1	Radar	N/A	231.7

[a]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.

Figure 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 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 seven 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 nonconventional maneuvers. In Experiment 2.2, we passed behind the obstacle, complying with the desirable behavior of COLREGs.

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, for example, 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 chooses nonconventional 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 toward obstacle estimate noise. This is an important property when using tracking systems based on exteroceptive sensors such as, for example, 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 multiobstacle 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. To challenge the algorithm, we present it with four multiobstacle scenarios:

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.

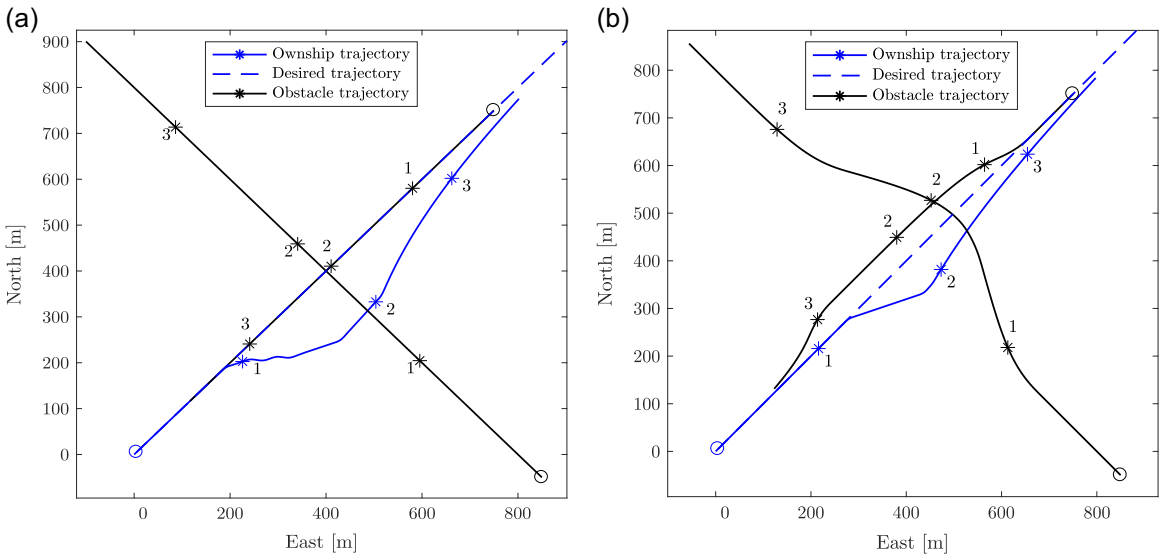

FIGURE 29 Simulation 1: Head on with simultaneous crossing from starboard with nonmaneuvering (a) and maneuvering (b) obstacles in accordance with COLREGs. The ownship and obstacle initial positions are marked with circles, and the numbers represent time markers for each 60 s [Color figure can be viewed at wileyonlinelibrary.com]

5.2 | Head on and crossing from starboard: Simulation 1

In this scenario, shown in Figure 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 toward starboard, and pass behind the head-on obstacle, which should maneuver to starboard in accordance with the head-on situation with the ownship. In Figure 29a, the obstacles do not maneuver, and the BC-MPC algorithm choose a maneuver to starboard to avoid the head-on obstacle and pass behind the crossing obstacle. In Figure 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 Figure 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 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 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 Figure 30a, the obstacles do not maneuver, and the ownship applies a clear and large maneuver to starboard to avoid the head-on obstacle, and avoid collision with the crossing obstacle. When the obstacles maneuver, shown in Figure 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 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 Figure 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 Figure 31a, the obstacles do not maneuver, and the ownship makes a starboard maneuver to avoid the head-on obstacle and

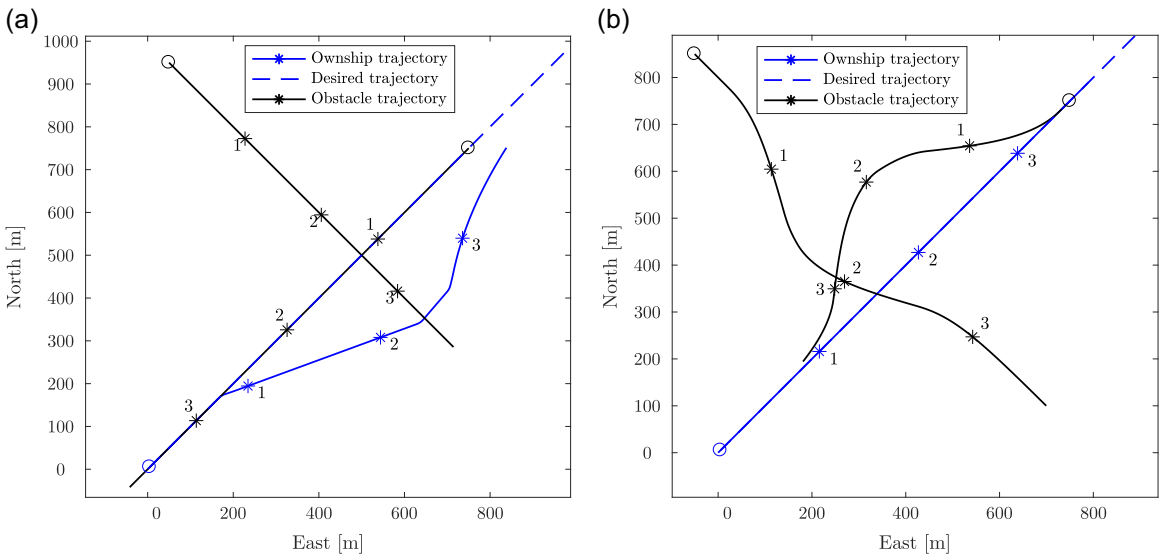

FIGURE 30 Simulation 2: Head on with simultaneous crossing from port with nonmaneuvering (a) and maneuvering (b) obstacles in accordance with COLREGs. The ownship and obstacle initial positions are marked with circles, and the numbers represent time markers for each 60 s [Color figure can be viewed at wileyonlinelibrary.com]

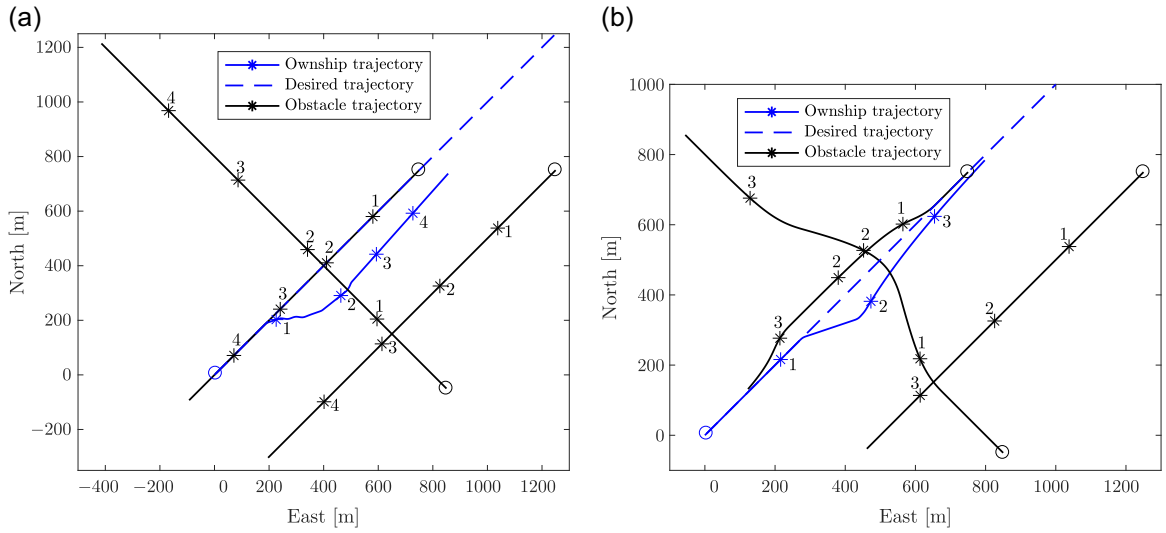

FIGURE 31 Simulation 3: Head on with simultaneous crossing from starboard with nonmaneuvering (a) and maneuvering (b) obstacles in accordance with COLREGs. In addition, a third obstacle approaches with an opposing course on a parallel path. The ownship and obstacle initial positions are marked with circles, and the numbers represent time markers for each 60 s [Color figure can be viewed at wileyonlinelibrary.com]

pass behind the crossing obstacle. Following this, the ownship makes a port maneuver 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 toward the desired trajectory in Figure 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 Figure 31b.

5.5 | Simultaneous crossing from starboard and port: Simulation 4

In this scenario, shown in Figure 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 Figure 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.

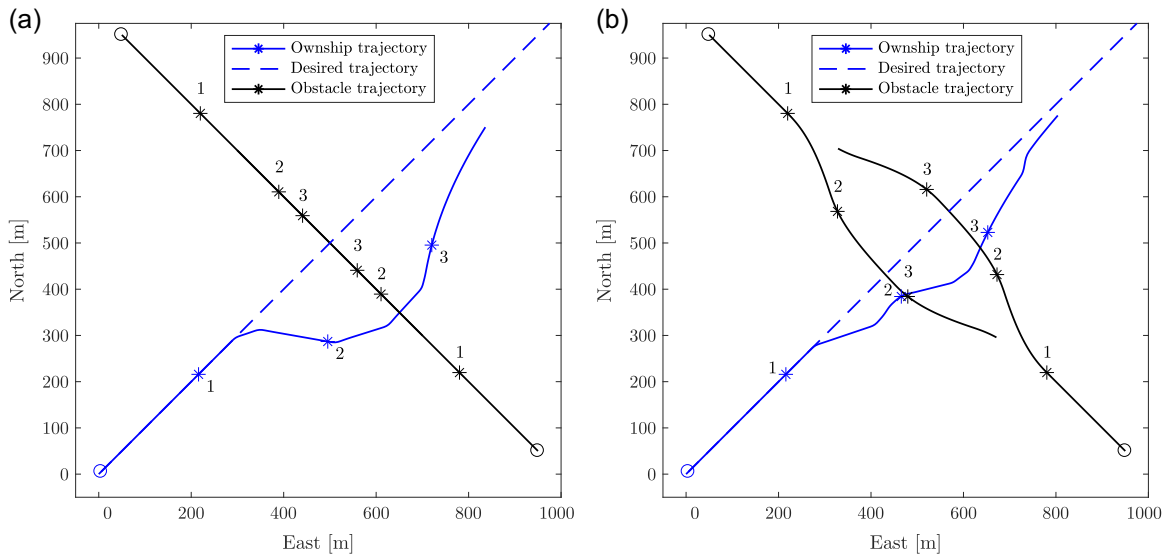

FIGURE 32 Simulation 4: Simultaneous crossing from starboard and port with nonmaneuvering (a) and maneuvering (b) obstacles in accordance with COLREGs. The ownship and obstacle initial positions are marked with circles, and the numbers represent time markers for each 60 s [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 4 Key points and numbers from the simulations

		Minimum distance to i obstacle nonmaneuvering/maneuvering		
Simulation type and number	Rule 13–15 compliance	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[a]
4: CS + CP	Yes	N/A	124.2 m/112.2 m	167.6 m/192.3 m

[a]The extra obstacle approaching with an opposing course on a parallel path.

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 Figure 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, 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. 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 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 do not 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 BC-MPC algorithm for ASV 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 nonmaneuvering 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 avoid 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 Rule 14 and 15 COLREGs when possible, and extra clearance is required if choosing nonconventional 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). Furthermore, we have also combined the algorithm with a long-term COLAV algorithm in a hybrid architecture in Eriksen, Bitar, Breivik, and Lekkas (2019), demonstrating COLAV compliant with COLREGs Rules 8 and 13–17 in simulations. 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, for example, include the effects of increasing noise levels on obstacle estimates, and how robust the algorithm is with respect to changes in the tuning parameters.

ACKNOWLEDGMENTS

This study 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.

ORCID

Bjørn-Olav H. Eriksen http://orcid.org/0000-0002-0786-2549
Morten Breivik http://orcid.org/0000-0002-0457-1850
Erik F. Wilthil http://orcid.org/0000-0001-8491-5432
Andreas L. Flåten http://orcid.org/0000-0001-9893-0107
Edmund F. Brekke http://orcid.org/0000-0001-8735-1687

REFERENCES

Abdelaal, M., & Hahn, A. (2016). NMPC-based trajectory tracking and collision avoidance of unmanned surface vessels with rule-based COLREGs confinement. *Proceedings of the 2016 IEEE Conference on Systems, Process and Control (ICSPC)*, Melaka, Malaysia, 23–28.

Benjamin, M. R., Leonard, J. J., Curcio, J. A., & 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., & Leonard, J. J. (2010). Nested autonomy for unmanned marine vehicles with MOOS-IvP. *Journal of Field Robotics*, *27*(6), 834–875.

Blaich, M., Rosenfelder, M., Schuster, M., Bittel, O., & Reuter, J. (2012). Fast grid based collision avoidance for vessels using A* search algorithm. *Proceedings of the 17th IEEE International Conference on Methods Models in Automation Robotics (MMAR)*, Miedzyzdrojie, Poland, 385–390.

Breivik, M., & Fossen, T. (2004). Path following for marine surface vessels. *Proceedings of OCEANS*, Kobe, Japan, 2282–2289.

Caldwell, C. V., Dunlap, D. D., & Collins, E. G. (2010). Motion planning for an autonomous underwater vehicle via sampling based model predictive control. Proceedings of the IEEE OCEANS conference, Seattle, WA, 1–6.

Chauvin, C. (2011). Human factors and maritime safety. *Journal of Navigation*, *64*(4), 625–632.

Cockcroft, A. N., & 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., & Brekke, E. F. (2018). The neighbor course distribution method with Gaussian mixture models for AIS-based vessel trajectory prediction. *Proceedings of the 21st IEEE International Conference on Information Fusion (FUSION)*, Cambridge, UK, 580–587.

Elkins, L., Sellers, D., & 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., Bitar, G., Breivik, M., & Lekkas, A. M. (2019). *Hybrid collision avoidance for ASVs compliant with COLREGs rules 8 and 13-17*. Submitted to Frontiers in Robotics and AI. Retrieved from https://arxiv.org/abs/1907.00198

Eriksen, B.-O. H., & Breivik, M. (2017a). *Modeling, identification and control of high-speed ASVs: Theory and experiments* (pp. 407–431). Cham: Springer International Publishing.

Eriksen, B.-O. H., & Breivik, M. (2017b). MPC-based mid-level collision avoidance for ASVs using nonlinear programming. *Proceedings of the 1st IEEE Conference on Control Technology and Applications (CCTA)*, Mauna Lani, HI, 766–772.

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

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

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

Eriksen, B.-O. H., Wilthil, E. F., Flåten, A. L., Brekke, E. F., & Breivik, M. (2018). Radar-based maritime collision avoidance using dynamic window. *Proceedings of the 2018 IEEE Aerospace Conference*, Big Sky, MT, 1–9.

Faltinsen, O. M. (2005). *Hydrodynamics of high-speed marine vehicles*. New York, NY: Cambridge University Press.

Fiorini, P., & Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research*, *17*(7), 760–772.

Flåten, A. L., & Brekke, E. F. (2017). Rao-blackwellized particle filter for turn rate estimation. Proceedings of the IEEE Aerospace conference, Big Sky, MT, 1–7.

Fossen, T. I. (2011). *Handbook of marine craft hydrodynamics and motion control*. Chichester, West Sussex: John Wiley & Sons Ltd.

Fox, D., Burgard, W., & Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, *4*(1), 23–33.

Gray, A., Ali, M., Gao, Y., Hedrick, J. K., & 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., & Johansen, T. A. (2018). MPC-based collision avoidance strategy for existing marine vessel guidance systems. *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia , 7618–7623.

Harati-Mokhtari, A., Wall, A., Brooks, P., & Wang, J. (2007). Automatic identification system (AIS): Data reliability and human error implications. *Journal of Navigation*, *60*(3), 373–389.

Hart, P., Nilsson, N., & 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*. Retrieved from http://www.imo.org/en/OurWork/Safety/Navigation/Pages/AIS.aspx

Johansen, T. A., Perez, T., & 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., & Bertram, T. (2015). A model predictive approach to emergency maneuvers in critical traffic situations. Proceedings of the IEEE ITSC conference, Las Palmas, Spain, 369–374.

Kongsberg Maritime. (2018). *Autonomous ship project, key facts about YARA Birkeland*. Retrieved from https://www.km.kongsberg.com/ks/web/nokbg0240.nsf/AllWeb/4B8113B707A50A4FC125811D00407045

Kuwata, Y., & 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., & Huntsberger, T. L. (2014). Safe maritime autonomous navigation with COLREGS, using velocity obstacles. *IEEE Journal of Oceanic Engineering*, *39*(1), 110–119.

La Valle, S. M. (1998). *Rapidly-exploring random trees: A new tool for path planning* (Technical Report No. 98–11). Iowa: Computer Science Department, Iowa State University.

Levander, O. (2017). Autonomous ships on the high seas. *IEEE Spectrum*, *54*(2), 26–31.

Ögren, P., & 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 (NTNU), Trondheim, Norway.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., & Ng, A. (2009). ROS: An open-source robot operating system. Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan.

Schuster, M., Blaich, M., & Reuter, J. (2014). Collision avoidance for vessels using a low-cost radar sensor. *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, 9673–9678.

Szlapczynski, R., & Szlapczynska, J. (2017). Review of ship safety domains: Models and applications. *Ocean Engineering*, *145*, 277–289.

Tan, C. S., Sutton, R., & 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., & Gupta, S. K. (2013). Dynamics-aware target following for an autonomous surface vehicle operating under COLREGs in civilian traffic. *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Tokyo, Japan, 3871–3878.

Wilthil, E. F., Flåten, A. L., & Brekke, E. F. (2017). *A target tracking system for ASV collision avoidance based on the PDAF* (pp. 269–288). Cham: Springer International Publishing.

How to cite this article: Eriksen B.-O. H., Breivik M., Wilthil E. F., Flåten A. L., Brekke E. F. The branching-course model predictive control algorithm for maritime collision avoidance. *J Field Robotics*. 2019;36:1222–1249.
https://doi.org/10.1002/rob.21900