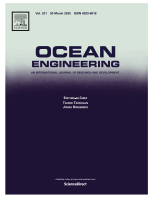
Ocean Engineering 353 (2026) 124802

Contents lists available at ScienceDirect

Ocean Engineering

journal homepage: www.elsevier.com/locate/oceaneng



Distributed MPC for autonomous ships on inland waterways with collaborative collision avoidance

Hoang Anh Tran [a,*], Tor Arne Johansen [a], Rudy R. Negenborn [b]

[a] Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), Trondheim, Norway
[b] Department of Maritime and Transport Technology, Delft University of Technology, Delft, The Netherlands

ARTICLE INFO

Keywords:
Collaborative collision avoidance
ADMM
Distributed model predictive control
Inland autonomous ships
Inland waterway traffic regulations

ABSTRACT

This paper presents a distributed solution for the problem of collaborative collision avoidance for autonomous inland waterway ships. A two-layer collision avoidance framework that considers inland waterway traffic regulations is proposed to increase navigational safety for autonomous ships. Our approach allows for modifying traffic rules without changing the collision avoidance algorithm, which is based on a novel formulation of model predictive control (MPC) for collision avoidance of ships. This MPC formulation is designed for inland waterway traffic and can handle complex scenarios. The alternating direction method of multipliers (ADMM) is used as a scheme for exchanging and negotiating intentions among ships. Simulation results demonstrate that the proposed algorithm enables ships to avoid collisions with a sufficient margin while adhering to traffic rules. Furthermore, the proposed algorithm can safely deviate from traffic rules when necessary to increase traffic efficiency in complex scenarios.

1. Introduction

1.1. Background and motivation

Developing inland autonomous surface ships is receiving increasing attention over the past years. The key to allowing the operation of an autonomous ship in inland waterway traffic (IWT) is the guarantee of navigation safety. A collision avoidance system (CAS) should ensure navigation safety of an autonomous ship. Different approaches for the CAS of inland autonomous ships have been proposed, including MPC (Mahipala and Johansen, 2023); Velocity obstacle (Zhang et al., 2022); Potential field method (Gan et al., 2022; Yan et al., 2020); and Scenario-based MPC (SB-MPC) (Tran et al., 2023a). Additional information about different solutions for developing CASs for inland autonomous ships can be found in Tran et al. (2023b).

When it comes to CASs, one of the challenges is predicting the intention and future trajectories of neighboring ships. Conventionally, a CAS predicts future trajectories of neighboring ships based on the information from onboard sensors and the AIS system. The current development of digital communication technology allows route exchange and intention sharing among ships (STM, 2015; Guiking, 2022). An intention exchange system combined with a distributed computing framework opens opportunities for collaborative CAS (C-CAS) of autonomous ships.

Over the past decade, proposals have been made to synthesize collaborative collision avoidance controllers for autonomous ships (Akdağ et al., 2022a; Zheng et al., 2017; Du et al., 2022). Frameworks for the C-CAS problem can be categorized as centralized or distributed. On one hand, a centralized framework uses one central computing unit to calculate a solution for C-CAS for all ships and broadcasts the solution over a network (Chen et al., 2018b; Tam and Bucknall, 2013; Kurowski et al., 2019). With this approach, the collision avoidance solution is usually globally optimal and does not require computational resources on each ship. However, this approach lacks scalability and robustness (Akdağ et al., 2022b). On the other hand, in a distributed framework, all ships within an area will calculate their own collision avoidance solutions and negotiate among one another through communication to adjust their own solutions (Chen et al., 2018a; Kim et al., 2017; Tang et al., 2022). Although the distributed C-CAS usually offers a locally optimal solution, robustness and scalability properties can lead to a distributed approach being a more promising solution for the C-CAS problem.

Because of the recent improvements in solvers for optimization problems, MPC has seen increasing applications in autonomous vehicles (Eriksen et al., 2020; Schwarting et al., 2017; Kneissl et al., 2018). MPC has shown the ability to significantly increase autonomous vehicles' safety even in complex traffic scenarios (Yu et al., 2021). In order to apply MPC to a distributed C-CAS problem, the Alternating

* Corresponding author.
E-mail addresses: hoang.a.tran@ntnu.no (H.A. Tran), tor.arne.johansen@ntnu.no (T.A. Johansen), r.r.negenborn@tudelft.nl (R.R. Negenborn).

https://doi.org/10.1016/j.oceaneng.2026.124802
Received 3 July 2025; Received in revised form 11 February 2026; Accepted 21 February 2026
Available online 2 March 2026

Direction Method of Multipliers (ADMM) is one of the well-known solutions used. Several studies have exploited the ADMM scheme to address the C-CAS problem for inland autonomous ships. A distributed MPC scheme approach based on ADMM was proposed by Zheng et al. (2017) to deal with the C-CAS problem in intersection crossing situations. The proposed communication scheme there, however, is not fully distributed since it requires one ship to take the coordinator role. A fully distributed nonlinear MPC-based ADMM, which requires no coordinator, is presented in Ferranti et al. (2018). Another fully distributed MPC scheme is proposed by Chen et al. (2018a) to deal with the C-CAS problem for vessel train formations. Unlike other ADMM schemes, Chen et al. (2018a) uses a serial iterative ADMM scheme instead of a parallel ADMM scheme. The iterative scheme allows ships to exploit up-to-date information from neighboring ships and is more suitable for transport networks (Negenborn and Maestre, 2014). In Chen et al. (2018a), how the update order for each ship within the serial iterative ADMM scheme affects the behavior of ships is also discussed, although a specific order is not suggested.

Although significant effort has been devoted to the problem of CAS for inland autonomous ships, none of the existing approaches explicitly consider the IWT regulations. In the existing literature, algorithms either consider a limited subset of the *Convention on the International Regulations for Preventing Collision at Sea* (COLREGS) or do not comply with any traffic regulations. Besides, methods considering COLREGS cannot be used directly, as different rules regulate inland waterways. To be allowed to operate in inland waterway traffic, ship's behaviors should comply with the local traffic regulations. Furthermore, it is recommended that, in case one ship needs to take action to avoid a collision, the altering of course or speed should be large enough to be readily apparent by neighboring ships (BPR, 2017; CCNR, 2023). One of the main obstacles when applying inland waterway regulations to collision avoidance algorithms is that the IWT regulations can change depending on the region of the waterways. For instance, the Netherlands' waterway is regulated by *Binnenvaartpolitiereglement* (BPR, 2017), while *Police regulations for the navigation of the Rhine* regulate the Rhine River (CCNR, 2023). A ship voyage could start in one region and end in another, meaning that the IWT regulation may change during the voyage. This poses a challenge for designing the CAS since changing the traffic rules could result in changing the behavior of the CAS algorithm.

1.2. Proposed two-layer framework for collision avoidance

In this paper, we introduce a C-CAS framework to deal with the inland waterway traffic rules compliance. The C-CAS framework includes two layers (see Fig. 1): the first layer is the traffic assessment & priority determination protocol (TAPD), and the second layer is the C-CAS algorithm. In the first layer, the traffic situation will be evaluated, and based on traffic rules a priority list is created through the communication protocol. The priority list is then used to determine the order of executing the C-CAS algorithm for each ship.

Following the inland traffic rules, in encountering situations between two ships, e.g., head-on, overtaking, and crossing, one ship is allowed to stand-on, and the other must give-way. A cost function can introduce the stand-on and give-way behavior using a binary variable as in Johansen et al. (2016), Tengesdal et al. (2022). However, this method is only effective for discrete input optimization algorithms such as SB-MPC. One way to apply this approach in MPC-based algorithms is to approximate the binary variable with a smooth function (Eriksen et al., 2020). Nonetheless, this approximation could increase the complexity of the MPC problem. Moreover, the traffic rules could be changed depending on the region in which a ship is sailing. If the traffic rules are integrated inside the CAS algorithm, then the CAS algorithm also has to be reformulated whenever the traffic rules change.

The advantages of our C-CAS framework are twofold. Firstly, the framework limits the complexity of the C-CAS algorithm by shifting the traffic rules out of the optimal control problems. Secondly, we can change the traffic rules by adjusting the upper layer, i.e., TAPD protocol, without changing the C-CAS algorithm.

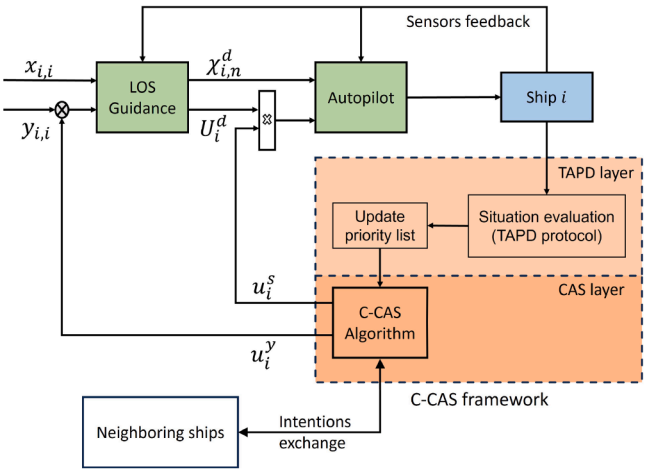


Fig. 1. Control scheme with the proposed C-CAS framework: U^d_i *and* $\chi^d_{i,n}$ *are nominal surge speed and course angle command (in the inertial frame* $\{n\}$*), respectively, used as input to an autopilot that contains course and speed control loops. The control signals from CAS are cross-track offset,* u^y_i*, and speed modification,* u^s_i*.*

1.3. Contributions

This paper proposes a novel C-CAS framework and a distributed algorithm that explicitly considers IWT regulations. The algorithm is based on distributed MPC (DMPC) using the ADMM framework. Each vessel individually computes a collision avoidance solution and exchanges intentions with neighboring ships through a hierarchical protocol. The main contributions of this paper are as follows:

1. A new approach to the problem of traffic rules compliance for collision avoidance: By introducing a two-layer framework, we separate the task of ensuring traffic rule compliance and avoiding collisions at different layers. This separation allows for modifying traffic rules (e.g., based on the region) without reformulating the CAS algorithm. Readers may find this approach similar to one presented in Eriksen et al. (2020). However, the method in Eriksen et al. (2020) integrates the COLREGS cost into the objective function, which prevents changes to traffic regulations without significantly altering the CAS algorithm.
2. A DMPC approach addresses the CAS problem for inland autonomous ships: This DMPC approach is an improvement of the algorithm presented in Tran et al. (2023a). A risk function is proposed to evaluate risks of collision between ships. By minimizing the risk function in the DMPC problem, we increase the navigation safety of ships. Besides, we use the serial iterative ADMM scheme to achieve a solution for the collaborative collision avoidance problem. Unlike the algorithm presented in Chen et al. (2018a), which is only suitable for bi-directional waterway scenarios, our algorithm can also solve intersection crossing scenarios.
3. The behaviors of ships using the proposed algorithm comply with traffic rules for the case of two-ship encounters. Most existing regulations only take this situation explicitly into account. In addition to this, the proposed algorithm reduces the collision risk in case of more than two ships.
4. The proposed algorithm is verified in simulation experiments with various representative traffic scenarios.

1.4. Outline

The remainder of this paper is organized as follows. Section 2 provides preliminaries on the coordinate systems, notation, and nonlinear

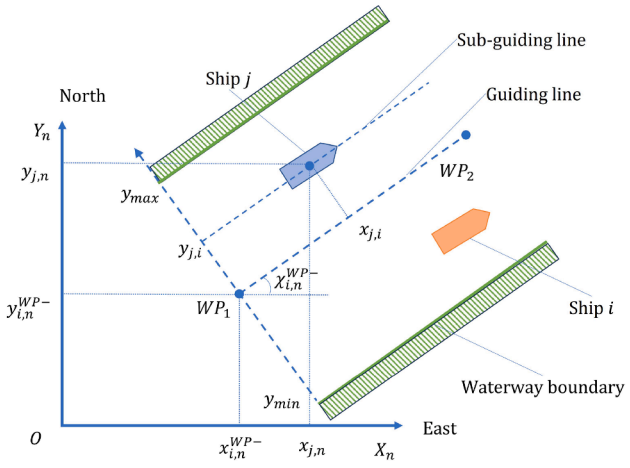


Fig. 2. Path coordinate and inertial coordinate.

ADMM. Section 3 presents the communication protocol adopted in the first layer. Section 4 describes the DMPC approach for the C-CAS problem adopted in the second layer. The results of simulation experiments are presented in Section 5, and Section 6 concludes and provides directions for future research.

2. Preliminaries

2.1. Notation

We denote the set of M ships as $\mathcal{M}$, and use the term "ship i" to refer to the ship with index i. The state of ship j in the inertial frame $\{n\}$ is $\eta_j = [x_{j,n}, y_{j,n}, \chi_{j,n}]^\top$, where $x_{j,n}, y_{j,n}$ are the position of the ship in X_n, Y_n axis and $\chi_{j,n}$ is the course angle, i.e., the angle from the X_n axis to the velocity vector of the ship (see Fig. 2). The velocity vector of ship j in the inertial frame $\{n\}$ is denoted as $v_{j,n}$. We use the path coordinate frame $\{w_i\}$ to illustrate the position of ship j along the defined waypoints of ship i as in Zheng et al. (2016). The position of ship j with respect to the path coordinate frame $\{w_i\}$ is defined as:

$$p_{j,i} = \begin{bmatrix} x_{j,i} \\ y_{j,i} \\ \chi_{j,i} \end{bmatrix} = R_i \begin{bmatrix} x_{j,n} - x_{j,n}^{WP^-} \\ y_{j,n} - y_{i,n}^{WP^-} \\ \chi_{j,n} - \chi_{i,n}^{WP^-} \end{bmatrix} = R_i(\eta_j - \eta_i^{WP^-}),$$

$$R_i = \begin{bmatrix} \cos\left(\chi_{i,n}^{WP^-}\right) & \sin\left(\chi_{i,n}^{WP^-}\right) & 0 \\ -\sin\left(\chi_{i,n}^{WP^-}\right) & \cos\left(\chi_{i,n}^{WP^-}\right) & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

where $\eta_i^{WP^-} = [x_{i,n}^{WP^-}, y_{i,n}^{WP^-}, \chi_{i,n}^{WP^-}]^\top$ contains the parameters of the previous active waypoint in the inertial frame $\{n\}$ (see Fig. 2). We use the notation $= [x_{i,i}, y_{i,i}, \chi_{i,i}]^\top$ to denote the state of ship i in the path coordinate frame $\{w_i\}$ of ship i. We denote the line segment that connects waypoints as the guiding line, and refer to a sub-guiding line as a line that is parallel with the guiding line. Moreover, the extended-real line is $\bar{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$, and the Kronecker product is $\otimes$.

2.2. Assumptions

We assume that the information from every ship is exchanged perfectly over the network:

Assumption 1. We consider the network of ships as a graph $\mathcal{G} = (v, e)$. Then, v is the set of ships; e is the set that represents the communication links between ships. We assume that at each time step k, the graph $\mathcal{G}$ is an undirected connected graph.

Assumption 2. We assume ships exchange information through a synchronous communication protocol, i.e., there is sufficient bandwidth for communication, and there is no delay or packet loss.

Assumption 3. We assume all ships in $\mathcal{M}$ exchange their position information continuously. Through communication and sensor information from their own, all ships have the same perception of the traffic situation.

It is worth noting that the Assumption 3 does not require the ship to have precise information about other neighboring ships, which is impractical. In here, we use the term "same perception of traffic situation" in the sense that if ship A knows that ship B must give way to A, then B also acknowledges that.

2.3. Nonlinear alternating direction method of multipliers

In Section 4 we introduce a distributed MPC algorithm to address the problem of collaborative collision avoidance. In this approach, we solve a distributed optimization problem to find a solution for ships to avoid collisions. This distributed optimization is solved cooperatively by each ship's controller that is solving its local optimization problem combined with information exchange between ships. We use the ADMM algorithm as a scheme for solving cooperative optimization problems and information exchange. More specifically, we use the Nonlinear Alternating Method of Multipliers (NADMM) that is proposed in Themelis and Patrinos (2020) as the basis for our algorithm. Compared with the original ADMM, this approach does not require the local optimization problem to be convex to ensure the algorithm's local convergence. Instead, the NADMM exploits the so-called *Douglas–Rachford splitting* to guarantee the tight convergence for nonconvex optimization problems.

Consider the following optimization problem:

$$\min_{(w,v)\in\mathbb{R}^m\times\mathbb{R}^n} \quad f(w) + g(v) \tag{1a}$$

$$\text{subject to} \quad Aw + Bv = b, \tag{1b}$$

where $f : \mathbb{R}^m \to \bar{\mathbb{R}}$, $g : \mathbb{R}^n \to \bar{\mathbb{R}}$, $A \in \mathbb{R}^{q\times m}$, $B \in \mathbb{R}^{q\times n}$, and $b \in \mathbb{R}^q$. According to Themelis and Patrinos (2020), NADMM iteratively solves problem (1) with the following steps:

$$\begin{cases} z^{+1/2} & = z - \beta(1-\lambda)(Aw + Bv - b), \\ w^+ & \in \arg\min \mathcal{L}_\beta(\cdot, v, z^{+1/2}), \\ z^+ & = z^{+1/2} + \beta(Aw + Bv - b), \\ v^+ & \in \arg\min \mathcal{L}_\beta(w, \cdot, z^+), \end{cases} \tag{2}$$

in which $\beta > 0$, and $\lambda \in (0, 2)$ are penalty and relaxation parameters, respectively. Moreover, $\mathcal{L}_\beta(w, v, z) := f(w) + g(v) + \langle z, Aw + Bv - b\rangle + \frac{\beta}{2}||Aw + Bv - b||^2$ is the augmented Lagrangian, where $z \in \mathbb{R}^q$ is the Lagrange multiplier. Besides, (w^+, v^+, z^+) denotes the updated state of (w, v, z).

3. Traffic assessment & priority determination protocol

In the first layer of the framework, traffic situation assessment is carried out. In the following, the TAPD protocol is presented in Section 3.1. Then, Section 3.2 discusses situations where the TAPD protocol could fail and how to resolve these situations.

3.1. Protocol development

From Chen et al. (2018a), we can observe that when two ships encounter one another, the ship that makes the first decision demonstrates the behavior of the give-way ship and vice versa. The explanation for this behavior is as follows. The ship that makes the first decision usually assumes that the other ship keeps the same course and speed. Assuming that the CAS can make a necessary action to avoid collision, this ship will act as a give-way ship. The other ship, with the information of

Algorithm 1 Priority assignment algorithm on ship i.

1: **for all** $j \in \mathcal{M}$ **do**
2: **if** j is on the STARBOARD side of the waterway **then**
3: $\rho_{j,i} := 1$
4: **else**
5: $\rho_{j,i} := 0$
6: **end if**
7: **end for**
8: **for all** $j \in \mathcal{M}, j \neq i$ **do**
9: **if** $T_i^{IC} < T_{IC}$ **and** $T_j^{IC} < T_{IC}$ **then**
10: $situation_i(j) = \text{CROSSING}$
11: **if** $\rho_{j,i} < 1$ **then**
12: **if** i comes from STARBOARD **then**
13: $\rho_{j,i} := \rho_{j,i} + \varpi$
14: **else if** i comes from PORT **then**
15: $\rho_{j,i} := \rho_{j,i} - \varpi$
16: **end if**
17: **end if**
18: **else if** j HEADON with i **then**
19: $situation_i(j) = \text{HEADON}$
20: **end if**
21: **if** $\rho_{ii} > \rho_{ij}$ **then**
22: $p\_list_i(j) := 1$
23: **else**
24: $p\_list_i(j) := 0$
25: **end if**
26: **end for**

CAS's actions from the former ship, can keep its course and portrays the behavior of a stand-on ship.

Influenced by this observation, we propose a protocol to establish the order of making decisions based on priority. Accordingly, the ship with give-way obligation will get priority to make a decision first, before the ship with stand-on gets permission. The TAPD protocol is as follows:

1. Depending on the situation with other ships and to comply with the traffic rules, every ship i ($i \in \mathcal{M}$) assigns a relative priority value for all surrounding ships j, including itself. We denote the relative priority value between ships i and j, assigned by ship i as $\rho_{j,i}$. It is not necessarily required that $\rho_{i,i} = \rho_{i,j}$, because the priority values reflect the relation between two ships.
2. All ships compare their priority values pair-wise to identify their priority in the network. A ship with a lower priority value has to give-way to a ship with higher one.
3. A ship i is considered to have the highest priority if no other ship has a higher priority value than ship i. There may be more than one highest-priority ship.
4. A ship i is considered to have the lowest priority if no other ship has a lower priority value than ship i. There may be more than one lowest-priority ship.
5. The C-CAS algorithm starts with the lowest-priority ships and ends with the highest-priority ships. In other words, a ship can make a CAS decision when all lower priority ships have made their decisions.

Remark 1. This protocol is established in a distributed manner, which requires no coordinator. Each ship identifies nearby neighboring ships with lower priority and waits until those lower priority ships have updated their decisions. Additionally, a ship broadcasts a signal within the network to notify neighboring ships when it has finished its C-CAS algorithm. Therefore, a coordinator to store the global priority list is not needed.

Our approach considers the rules adopted in the Netherlands' inland waterways (BPR, 2017) to determine the give-way or stand-on priorities. The following rules are considered:

- Head-on situation: If two vessels are approaching each other on opposite courses in such a way that there is a risk of collision, the vessel not following the starboard side of the fairway shall give-way to the vessel following the starboard side of the fairway. If neither vessel follows the starboard side of the fairway, each shall give-way to vessels on the starboard side so that they pass each other port to port.
- Crossing situation: If the courses of two ships cross each other in such a way that there is a risk of collision, the vessel not following the starboard side of the fairway shall give-way to the vessel following the starboard side of the fairway. In case none of the ships follows the starboard side of the fairway, the ship approaching from the port side gives way to the vessel approaching from starboard.
- Overtaking situation: A vessel overtaking another vessel should keep out of the way of the overtaken vessel.

The details of the priority assignment algorithm on ship i are presented in Algorithm 1. Firstly, ship i assigns to the ships the priority value 1 if the ship sails on the starboard side of the waterway (steps 1 – 6). If the time to approach the intersection of ship i and ship j, are both less than a predefined value T_{IC}, then the CROSSING situation is established (step 9). The time to approach the intersection of ship i, T_i^{IC}, is defined as the time (from present) until ship i reaches a predefined point at the center of the intersection. We assume that all ships know this center point beforehand. In CROSSING situation, we increase or decrease the priority value of ship j by ϖ ($\varpi > 0$) if ship i comes, respectively, from STARBOARD or PORT side of ship j (steps 11–16). It is important that $\varpi < 1$, so it does not affect the STARBOARD priority status of ship j and i that was defined in steps 1 – 6. The HEADON status is defined as:

$$\text{HEADON} = \left(\frac{v_{i,n} \cdot v_{j,n}}{|v_{i,n}|.|v_{j,n}|} < -\cos(22.5°) \right) \wedge (\text{TCPA} \leq T_{ho}) \wedge (\text{ship } j, i \text{ are in the same waterway}), \quad (3)$$

with TCPA is the time to closest point of approach between two ships, and T_{ho} being a predefined positive scalar. Here, a ship j is in the same waterway with ship i if $y_{\min} \leq y_{j,i} \leq y_{\max}$, where $y_{\min}$ and $y_{\max}$ define the boundaries width of the waterway in frame $\{w_i\}$ (see Fig. 2). The output of Algorithm 1 is the priority list p_list_i. If $p\_list_i(j) = 1$, ship i gives way to ship j; otherwise, ship i does not have to give way to ship j.

It is worth noting that the priority value assigned by Algorithm 1 is based on traffic rules. Therefore, it could be modified in accordance with the applicable traffic rules. The use of the Netherlands' regulations in this paper is only an example of how the traffic regulations are used. Algorithm 1 can be based on another set of regulations depending on the intended usage. Additionally, Algorithm 1 must satisfy the condition of consistency. That is, the priority of ship i over ship j that is calculated on ship i must be consistent with that calculated on ship j. This condition means that if Algorithm 1 on ship i gives that if ship i gives way to ship j, then Algorithm 1 on ship j must be ship j stand-on over ship i. In other words, the condition is equivalent to "if $\rho_{j,i} > \rho_{i,i}$ then $\rho_{j,j} > \rho_{j,i}$". As long as ships share the same perception of the surrounding environment, and there is no ambiguity in traffic regulations regarding the priority between ships, this condition holds. In rare cases, if this condition is not satisfied, it could lead to a deadlock situation, which is addressed next, in Section 3.2.

3.2. Resolving deadlock situations

From the step 5 of the TAPD protocol, a C-CAS's iteration starts from ships with the lowest priority. However, there are rare situations in which a ship with the lowest priority does not exist. For example, in Fig. 3, each ship gives way to the ship that comes from the starboard side; none of the four ships will make the first decision. We call this a deadlock situation. This situation can be detected with a coordinator that stores and compares the priority points of all ships. However,

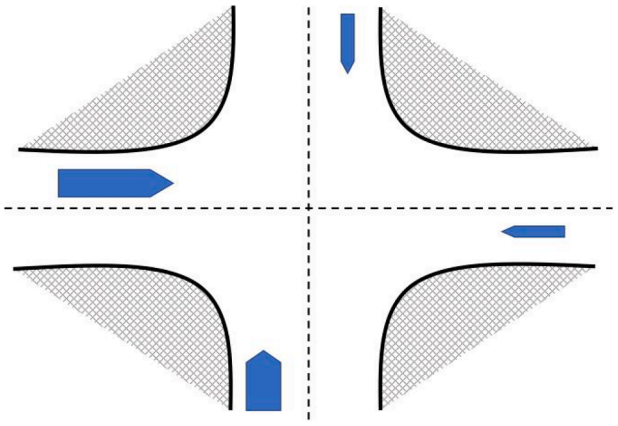


Fig. 3. Example of deadlock situation: Each ship wait for the ship comes from their starboard side.

our protocol is developed in a distributed manner without a coordinator. Therefore, we detect and resolve the deadlock situation through the following steps:

1. A timer is started when a ship waits for other lower-priority ship decisions. The deadlock is detected if, after a specific time, T_{DL}, none of the ships within the network updates its decision.
2. When a deadlock occurs, each ship exchanges the information of its weight and compares it with neighboring ships. At this time, the ship with the lightest weight, which has not made a decision, makes the first decision. In this case, the lightest-weight ship is the lowest-priority ship according to rule 4 of the TAPD protocol.

In this process, we use the weight of the ship as the secondary criterion to determine the priority between ships, since it is stated in BPR (2017) that small ships must give way to bigger ships. In practice, other criteria can be used as secondary criteria, e.g., length, beam, speed, or draft of ships.

4. Collaborative collision avoidance algorithm

In the second layer of our framework a distributed MPC-based collaborative collision avoidance algorithm for autonomous ships in inland waterway is embedded. We use the control scheme presented in Fig. 1, including line-of-sight (LOS) guidance (Fossen, 2011), CAS, and autopilot. In this scheme, LOS guidance keeps the ship on track with the predefined waypoints, while CAS helps the ship to avoid collisions. Unlike the MPC-based CASs of Chen et al. (2018a), Ferranti et al. (2022), where CASs take responsibility for guiding a ship from waypoint to waypoint, in our approach this task is handled by LOS guidance. In other words, the only task of the CAS of ship i is modifying the trajectory to guarantee a collision-free path for that system. This task is achieved by adding a cross-track offset (u^y) and speed modification (u^s) to the LOS system so that the OS will avoid an obstacle by sailing parallel with the guiding line. Further details regarding the design of the LOS system can be found in Fossen (2011).

4.1. Kinematic model of ship i

First let us define the kinematic model kinematic model of ship i with respect to the path coordinate frame $\{w_i\}$ based on the setpoint filter model (Lutz and Gilles, 2010) as follows:

$$
\begin{aligned}
x_{i,i}(k) &= x_{i,i}(k-1) + \Delta T u_i^s(k) U_i^d \cos(\chi_{i,i}(k)),\\
y_{i,i}(k) &= y_{i,i}(k-1) + \Delta T u_i^s(k) U_i^d \sin(\chi_{i,i}(k)),\\
\chi_{i,i}(k) &= \chi_{i,i}(k-1) + \frac{\Delta T}{T_1}\cdot\left[\chi_i^{max}\tanh(K_e(u_i^y(k) - y_{i,i}(k-1))) - \chi_{i,i}(k-1)\right],
\end{aligned}
\tag{4}
$$

where U_i^d, χ_i^{max} are, respectively, the nominal surge speed of ship i and the maximum steering angle that ship i can achieve in a sampling period ΔT, and T_1, K_e are positive constants depending on the hydrodynamics of the vessel and the controller tuning. Furthermore, $u_i(k) = \left[u_i^y(k), u_i^s(k))\right]^\top$ denotes the control action of ship i, where u_i^y, u_i^s are the cross-track offset and speed modification, respectively. The cross-track offset, u_i^y, dictates the position of the ship along the y-axis, while speed modification, u_i^s, adjusts the speed of the ship. The kinematic model (4) of ship i can then be rewritten in compact form as:

$$
(k+1) = f_i((k), u_i(k)). \tag{5}
$$

4.2. Risk evaluation

In order to minimize the risk of collision for ship i with neighboring ships, we introduce the collision risk function of ship i with respect to ship j, $R_{ij}(\cdot)$. The value of the risk function rises sharply when the distance between two ships is closer than a predefined value and it is approximately zero otherwise. This approach is similar to what is used in the SB-MPC algorithm (Johansen et al., 2016) but with a modification to remove the singular point that can cause numerical issues in a gradient-based MPC. The predicted risk of collision of ship i with respect to ship j at the time step $t_0 + k$ from the present time t_0 is defined as:

$$
R_{ij}(t_0+k) = \frac{K_{ca}}{\sqrt{1+K_d k}}\exp\left(-\frac{\left(\delta x_{ij}(k)\right)^2}{\alpha_{xj}} - \frac{\left(\delta y_{ij}(k)\right)^2}{\alpha_{yj}}\right), \tag{6}
$$

where $\delta x_{ij}(k) = x_{i,i}(t_0+k) - x_{j,i}(t_0+k)$, $\delta y_{ij}(k) = y_{i,i}(t_0+k) - y_{j,i}(t_0+k)$ are the predicted distances between ship i and j, respectively, and K_{ca} is predefined constant based on safety criteria that depend on the traffic situation. Moreover, α_{xj} and α_{yj} are parameters that are associated with the size, shape, and current speed of ship j. Accordingly, the greater the size or the speed of the ship, the larger the value of α_{xj} and α_{yj}. Besides, the prediction of risk further away from the present time tends to be less accurate. Therefore, we implement a discount factor $\frac{1}{\sqrt{1+K_d k}}$, with $K_d \geq 0$, to reduce the weight of the collision risk in the our MPC formulation as k increases. It is worth mentioning that, in most cases, we do not have $R_{ij} = R_{ji}$, because $\alpha_{xj} \neq \alpha_{xi}$ or $\alpha_{yj} \neq \alpha_{yi}$.

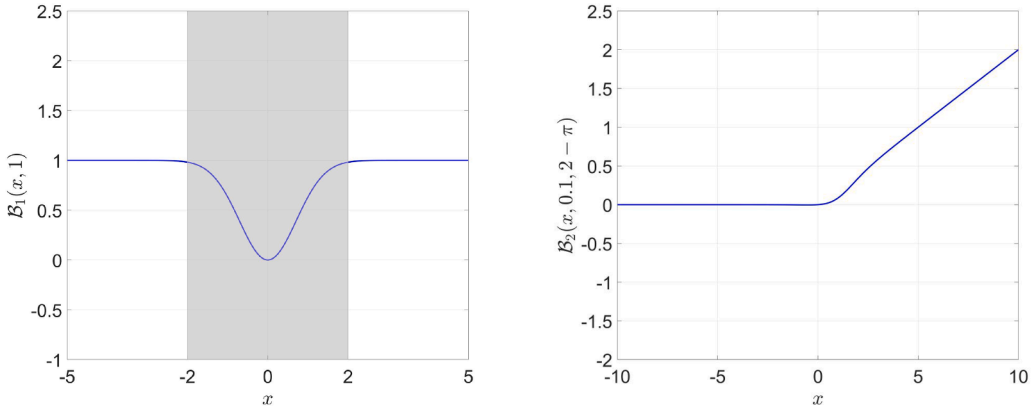
In the simulations presented in Section 5, we choose α_{xj} and α_{yj} greater than, or equal to, the occupied area of ship j, i.e., the length and beam of the ship. In practice, one can choose larger values for α_{xj} and α_{yj} to increase the safety domain around ship i. Increasing α_{xj} and α_{yj} can be helpful when the lower-level controller cannot precisely follow the command from the C-CAS algorithm due to external disturbances, or when ship j is moving at high speed. However, a large value of α_{xj} and α_{yj} may lead to unnecessary course or speed changes and less efficient use of the waterway's capacity.

4.3. Problem formulation

The main task of the CAS is to avoid collisions with consistent behaviors that comply with traffic rules. Thus, we propose the following cost function of ship i for the MPC formulation:

$$
\mathcal{J}_i(\tilde{p}_{i,i}, \tilde{u}_i) = \mathcal{J}_i^{ca}(\tilde{p}_{i,i}) + \mathcal{J}_i^{e}(\tilde{u}_i) + \mathcal{J}_i^{b}(\tilde{u}_i), \tag{7}
$$

where $\tilde{p}_{i,i}(t)$ and $\tilde{u}_i(t)$ are vectors containing the system state and input over the control horizon, i.e., $\tilde{u}_i(t) = \left[u_i^\top(t), u_i^\top(t+1), ..., u_i^\top(t+N-1)\right]^\top$, $\tilde{p}_{i,i}(t) = \left[^\top(t), ^\top(t+1), ..., ^\top(t+N)\right]^\top$. Moreover, $\mathcal{J}_i^{ca}(\tilde{p}_{i,i})$ is the sum of collision risks with respect to all neighboring ships over the horizon, i.e., $\mathcal{J}_i^{ca}(\tilde{p}_{i,i}) = \sum_{k=1}^{N+1}\left(\sum_{j\in\mathcal{M}\setminus\{i\}} R_{ij}(t+k)\right)$. It should be mentioned that $\mathcal{J}_i^{ca}(\tilde{p}_{i,i})$ also depends on the state of the neighboring ships. However, in the cost function of ship i, the state of ship j is not a decision variable and is considered as a given parameter instead. $\mathcal{J}_i^{e}(\tilde{u}_i)$ represents the cost of collision avoidance control actions, necessary for reflecting that



(a) $\mathcal{B}_1(x,r)$ with $r=1$ (b) $\mathcal{B}_2(x,r_1,r_2)$ with $r_1=0.1$, $r_2=2-\pi$.

Fig. 4. Behavior functions $\mathcal{B}_1(x,r)$ and $\mathcal{B}_2(x,r_1,r_2)$.

ship i should only change the course or speed when it can significantly decrease the collision risk. $\mathcal{J}_i^e(\tilde{u}_i)$ is defined as follows:

$$\mathcal{J}_i^e(\tilde{u}_i)=\sum_{k=1}^{N+1}\Big[K_y\big(u_i^y(t+k)-u_i^y(t+k-1)\big)^2+K_s\big(1-u_i^s(t+k)\big)^2\Big], \tag{8}$$

with K_y, $K_s>0$ being control parameters. On the one hand, smaller values of K_y and K_s make the ship overreact with collision risk, i.e., making an unnecessary action. On the other hand, larger values reduce the magnitude of course/speed change of the ship when facing a risk of collision. We can prioritize one collision avoidance action over another by changing the ratio $\frac{K_y}{K_s}$. For example, ship i will prioritize to change course rather than changing speed if $\frac{K_y}{K_s}$ is reduced and vice versa.

In order to make the behavior of ship i adequately represent traffic rules and be more visible to neighboring ships, i.e., the change in course or speed should be of a sufficiently large magnitude instead of a sequence of small changes; we define $\mathcal{J}_i^b(\tilde{u}_i)$ as follows:

$$\begin{aligned}\mathcal{J}_i^b(\tilde{u}_i)=\sum_{k=1}^{N+1}\Big[&\mu_1\mathcal{B}_1\big(1-u_i^s(t+k),\gamma_s\big)+\mu_2\mathcal{B}_1\big(u_i^y(t+k)-u_i^y(t+k-1),\gamma_y\big)\\&+\mathcal{B}_2\big(u_i^y(t+k-1)-u_i^y(t+k),\mu_3\big)\Big].\end{aligned} \tag{9}$$

The first and second terms in (9) represent that if the collision avoidance decision is made, it must be large enough to be observable by other neighboring ships. $\mathcal{B}_1(x,r)$ is defined as $\mathcal{B}_1(x,r)=1-\exp\left(\frac{-x^2}{r}\right)$, with $r>0$ is a parameter. As illustrated in Fig. 4(a), if the solution lies within the grey area, it can be easily be attracted toward the minimum point at zero. However, if the solution lies outside the grey area, the solution now lies on a local minimum point (as the gradient at this area is approximately zero). Hence, $\mathcal{B}_1(x,r)$ penalizes only the solution with small magnitude, but not those with large enough magnitude. The third term penalizes the port side steering behavior of ship i. The behavior function $\mathcal{B}_2(x,r_1,r_2)$ is defined as $\mathcal{B}_2(x,r_1,r_2)=r_1x[\tanh(x+r_2)+1]$, where r_1,r_2 are constant parameters. As shown in Fig. 4(b), the value of the function will increase as the ship steers toward the port side, i.e., $u_i^y(t+k-1)>u_i^y(t+k)$. Moreover, because $\mathcal{B}_2(x,r_1,r_2)$ increases slowly, ship i can still steer port if it can significantly reduce risk.

Remark 2. The behavior functions ensure the predictability of the ships' actions. They do not determine the ship's action (give-way or stand-on); instead, they make those actions, when required, more predictable and observable. The behavior functions follow standard practice in ship navigation and comply with traffic regulations.

We formulate the distributed MPC collision avoidance problem of ship i, $i\in\mathcal{M}$, with cost function (7) as follows:

$$\min_{\tilde{p}_{i,i},\tilde{u}_i}\quad \mathcal{J}_i(\tilde{p}_{i,i},\tilde{u}_i) \tag{10a}$$

$$\text{s.t.:}\quad (t+k+1)=f_i((t+k),u_i(t+k)), \tag{10b}$$

$$(t)=p_{i,i}^{init}, \tag{10c}$$

$$u_i(t+k)\in U_i, \tag{10d}$$

$$\tilde{p}_{i,i}=\tilde{R}_i(\xi_i-\bar{\eta}_i^{WP^-}), \tag{10e}$$

where $p_{i,i}^{init}$ is the current position of ship i, U_i defines the boundary set for control inputs $u_i(t+k)$; ξ_i is the global variable that is sent to other ships. It represents the predicted position of ship i with respect to $\{n\}$ over the prediction horizon, and is defined as $\xi_i=\tilde{R}_i^{-1}\tilde{p}_{i,i}+\bar{\eta}_i^{WP^-}$. Moreover, $\bar{\eta}_i^{WP^-}=\mathbf{1}_{N+1}\otimes\eta_i^{WP^-}$. Besides, $\tilde{R}_i$ is a block diagonal matrix with each block being the rotation matrix R_i:

$$\tilde{R}_i=\begin{bmatrix}R_i & & 0\\ & \ddots & \\ 0 & & R_i\end{bmatrix}\in\mathbb{R}^{3(N+1)\times 3(N+1)}.$$

Let us introduce $\bar{p}_{i,i}$ as the global variable that stores the position of ship i with respect to $\{w_i\}$, i.e., $\bar{p}_{i,i}=\tilde{R}_i(\xi_i-\bar{\eta}_i^{WP^-})$. Problem (10) is now rewritten in compact form as follows:

$$\min_{\tilde{u}_i,\tilde{p}_{i,i}}\quad \mathcal{J}_i(\tilde{p}_{i,i},\tilde{u}_i) \tag{11a}$$

$$\text{s.t.:}\quad [\tilde{u}_i^\top,\tilde{p}_{i,i}^\top]^\top\in\mathbf{G}_i \tag{11b}$$

$$\tilde{p}_{i,i}=\bar{p}_{i,i}, \tag{11c}$$

where (11c) is equivalent to (10e), and

$$\mathbf{G}_i:=\Big\{[\tilde{u}_i^\top,\tilde{p}_{i,i}^\top]^\top\,|\,\text{(10b), (10c), (10d) are satisfied}\Big\} \tag{12}$$

is the feasible region for ship i. Then, we have the NADMM update for the controller of ship i at iteration index s:

$$z_i^{s+1/2}=z_i^s-\beta(1-\lambda)\Big(\tilde{p}_{i,i}^s-\bar{p}_{i,i}^s\Big) \tag{13a}$$

$$\begin{aligned}\begin{bmatrix}\tilde{u}_i^{s+1}\\ \tilde{p}_{i,i}^{s+1}\end{bmatrix}=\text{argmin}_{[\tilde{u}_i,\tilde{p}_{i,i}]^\top\in\mathbf{G}_i}\Big\{&\mathcal{J}_i(\tilde{p}_{i,i},\tilde{u}_i)\\&+\Big\langle z_i^{s+1/2},\Big(\tilde{p}_{i,i}-\bar{p}_{i,i}^s\Big)\Big\rangle+\frac{\beta}{2}\|\tilde{p}_{i,i}-\bar{p}_{i,i}^s\|^2\Big\}\end{aligned} \tag{13b}$$

$$z_i^{s+1}=z_i^{s+1/2}+\beta\Big(\tilde{p}_{i,i}^{s+1}-\bar{p}_{i,i}^s\Big), \tag{13c}$$

$$\bar{p}_{i,i}^{s+1}=\tilde{p}_{i,i}^{s+1}+\frac{1}{\beta}z_i^{s+1}. \tag{13d}$$

It should be noted that when performing the NADMM update, each agent only update their decision variable and does not change others' decisions. Therefore, only parts of w and v are updated (in (13b) and (13d), respectively) each time an agent performs (13).

The details of the collaborative collision avoidance algorithm are given in Algorithm 2. The algorithm is executed after the communication protocol has established the priority list. According to the priority list ship i has to wait other ships with lower priority. At each iteration s, if all lower priority ships have made their decision and the number of iterations has not reached the maximum (step 4), then ship i can execute the algorithm (step 12). Before performing an update, ship i needs to know the future position of ship j, i.e., $\tilde{p}_{j,i}$, to calculate the risk function $R_{ij}(\cdot)$ as in (6). The future position of ship j is obtained via communication if it is available (step 7) or from a motion prediction algorithm if it is not available. Different prediction algorithms can be used to predict future motion of neighboring ships based on information from sensors and AIS system (see Tran et al., 2023a). Here, we adopt a constant velocity model to predict the motion of neighboring ships. In step 9, the prediction algorithm is called. The decision of ship i is then transformed into the inertial frame (step 13) and broadcast to neighboring ships (step 14).

Algorithm 2 Collaborative collision avoidance.

Input: For $i, j \in \mathcal{M}, i \neq j$, initialize $\mathbf{p}_{ii}^{1}, z_{ij}^{1}$ based on trajectory prediction.

1: $iter(i) := 0 \ \forall i \in \mathcal{M}$.
2: **for** s = 1,2,... **do**
3: **for all** $i \in \mathcal{M}$ **do**
4: **if** $wait(i) ==$ **false and** $ADMM_{DONE}(i) ==$ **false then**
5: **for all** $j \neq i$ **do**
6: **if** ξ_j is available **then**
7: $\tilde{p}_{j,i} := \tilde{R}_i(\xi_j - \bar{\eta}_i^{WP^-})$
8: **else**
9: $\tilde{p}_{j,i} := trajectory\_prediction((t))$
10: **end if**
11: **end for**
12: update according to (13)
13: $\xi_i := \tilde{R}_i^{-1}\tilde{p}_{i,i} + \bar{\eta}_i^{WP^-}$
14: Transmit data ξ_i to all ship $j \in \mathcal{M}_i$.
15: $iter(i) := iter(i) + 1$.
16: **if** $iter(i) == iter_{max}$ **then**
17: $ADMM_{DONE}(i) =$ **true**
18: $iter(i) := 0$.
19: **end if**
20: **else**
21: $\tilde{u}_i^{s+1} := \tilde{u}_i^s$
22: $\tilde{p}_{i,i}^{s+1} := \tilde{p}_{i,i}^s$
23: **end if**
24: **end for**
25: **if** $ADMM_{DONE}(i) ==$ **true** $\forall i \in \mathcal{M}$ **then**
26: **break**.
27: **end if**
28: **end for**

4.4. Convergence of Algorithm 2

We present in Theorem 1 the convergence analysis for Algorithm 2. This proof of convergence guarantees that the solution provided by Algorithm 2 helps ships to collaboratively reduce collision risks.

Theorem 1. *Assume Problem (11) has feasible solutions for all $i \in \mathcal{M}$. Then the solution provided by Algorithm 2 with $\lambda \in (0, 2)$, $\beta > 2L$, and $L > 0$ converges asymptotically to a (locally) optimal solution.*

Proof. See Appendix A. □

Algorithm 2 works when a solution exists to avoid collision. If such a solution does not exist, e.g., the waterway is too narrow to do any course change, human interference is required.

Table 1
Kinematics model's parameters.

Parameter	Unit	Simulation
$y_{\max}$	m	60
$\chi^{\max}$	rad	$\pi/6$
T_1/T_j	–	28.458
ΔT	s	20

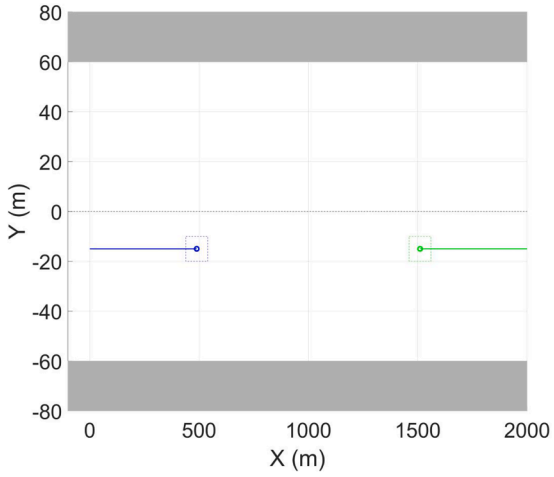


Fig. 5. Head on situation in inland waterway between two ships. Ship 1 and 2 are illustrated in blue and green dots, respectively. The rectangle around each ship is the safety area. The waterway is illustrated in white, and the grey area is where a ship cannot sail. The dash line at $Y = 0m$ is the central line divide port and starboard side of a waterway.

5. Simulation experiments

This section illustrates the performance of the proposed framework through several simulation experiments. Two typical scenarios are being used to evaluate the proposed framework: head-on and intersection crossing. In head-on scenarios, by evaluating different control parameters, we show how these parameters affect ships' behavior. In intersection crossing scenarios, we show how the complexity of the traffic situation could affect the behavior of ships.

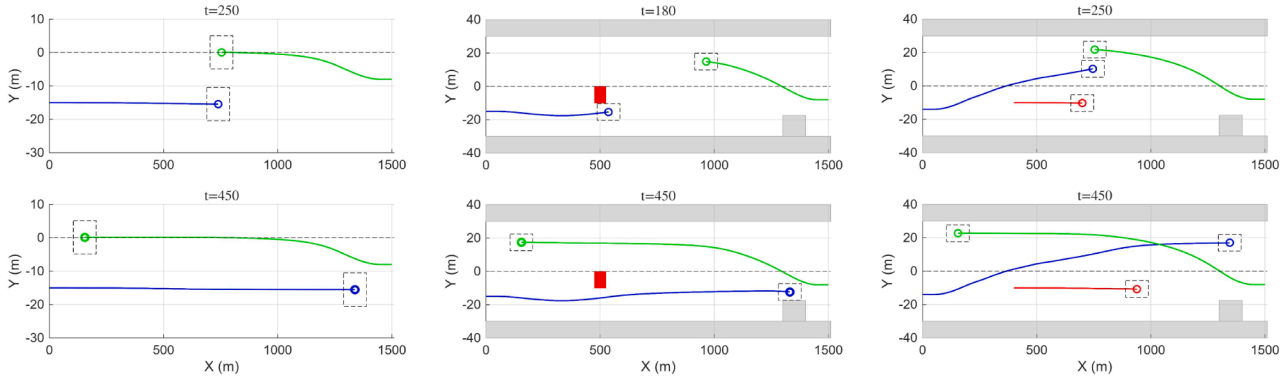
The local MPC problem (13b) is solved using the Casadi toolbox (Andersson et al., 2019) with the *ipopt* solver (Wächter and Biegler, 2004), i.e., an open source interior point optimizer software package for large-scale nonlinear optimization. The simulations are implemented in MATLAB 2025b running on a PC with an AMD Ryzen 9-8940HX and 32GB of RAM.

The parameters of the kinematics model (4) is presented in Table 1. Here y_{max} is the bound of the control signal u_i^y, i.e., $|u\_i^y| \leq y\_\max$, and the bound of u_i^s is $0 \leq u_i^y \leq 1$.

5.1. Head-on scenarios

In this scenario, two ships sail in the same waterway but in opposite directions (see Fig. 5). This scenario requires each ship to adopt a course change. According to the traffic rules mentioned in Section 3.1, a ship sailing on the starboard side of the waterway has the right to stand-on. Consequently, the expected resulting paths should be that ship 2 gives way by steering toward the starboard while ship 1 keeps its way. Additionally, this scenario considers the waterway to be wide enough for ships to perform a course change. Otherwise, if the waterway is too narrow and a feasible solution does not exist, a human decision is needed.

The simulations focus mainly on the collaboration of ships to avoid collisions with each other. Therefore, we omit the presence of static obstacles. However, if there are static obstacles, the proposed ADMM



(a) Head-on scenarios without static obstacles.

(b) Head-on scenarios with static obstacles. Ships must avoid their safety area to enter the red and gray zones.

(c) Combine head-on and overtaking scenarios.

Fig. 6. Head-on situation in an inland waterway between two ships in different scenarios. Ships 1, 2, and 3 are illustrated in blue, green, and red dots, respectively.

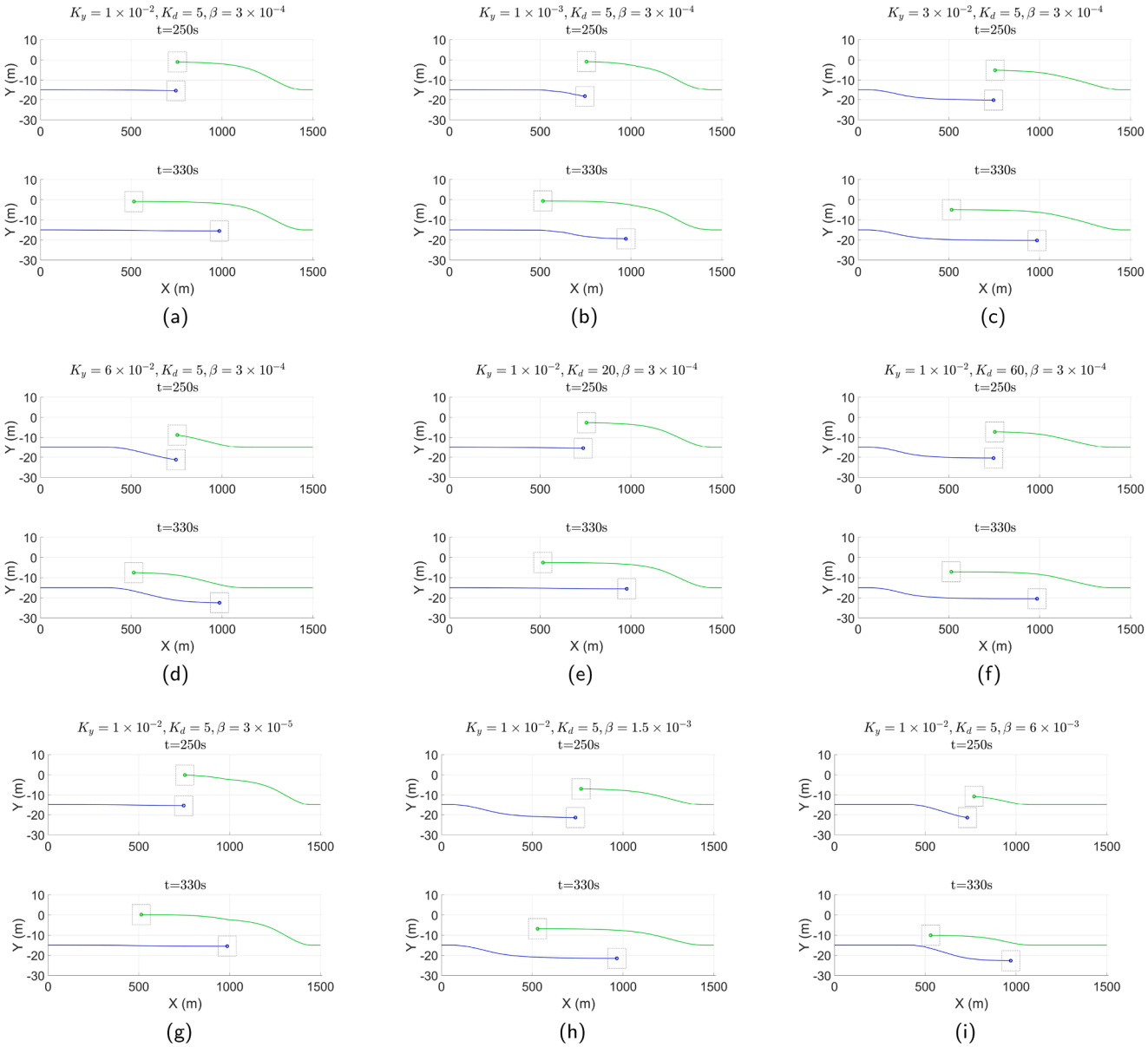


Fig. 7. Head-on situation with different sets of parameters. Ships 1 and 2 are illustrated in blue and green dots, respectively. The black dashed rectangles are the safety area of ships.

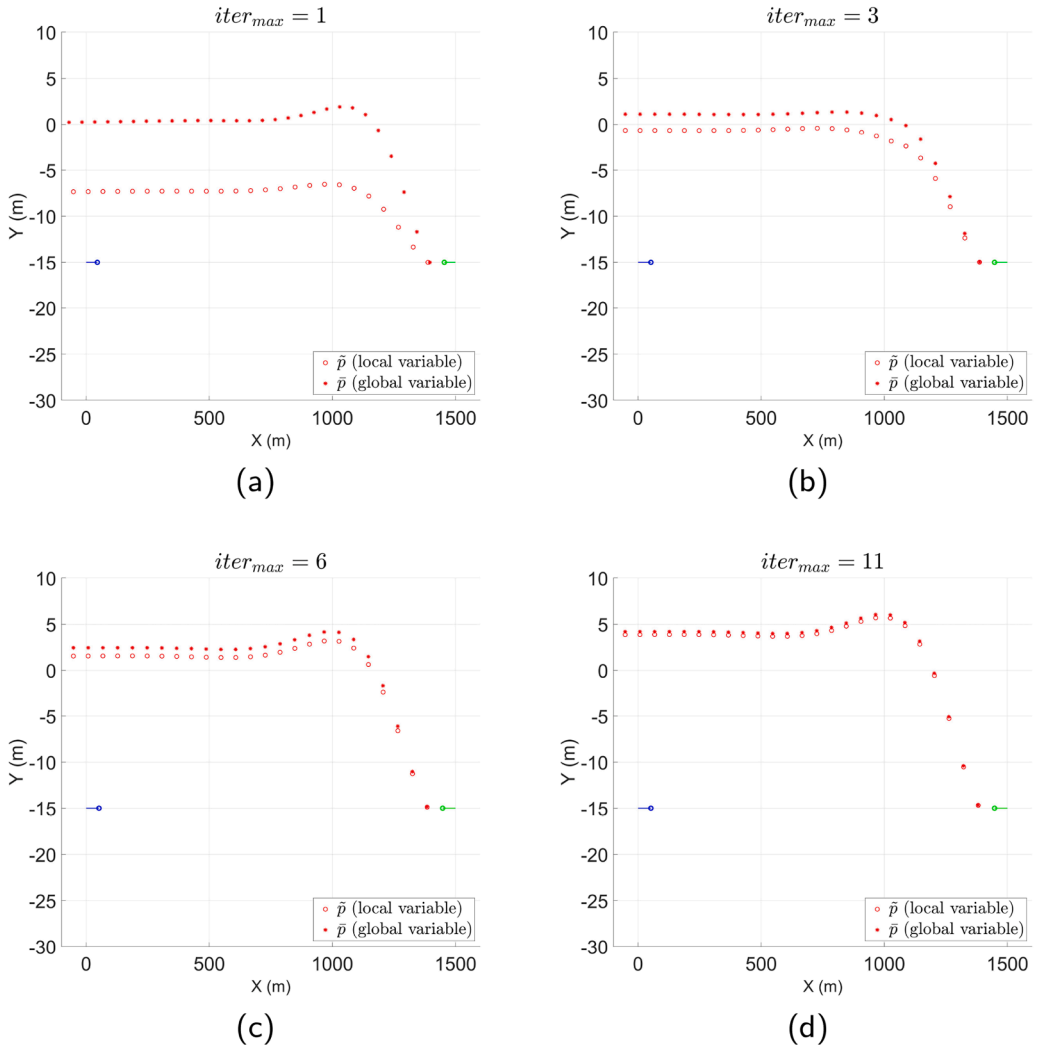


Fig. 8. Difference between $\bar{p}_{i,i}^{s+1}$ and $\tilde{p}_{i,i}^{s+1}$ with different $iter_{max}$. Ship 1 and 2 are illustrated in blue and green dots, respectively.

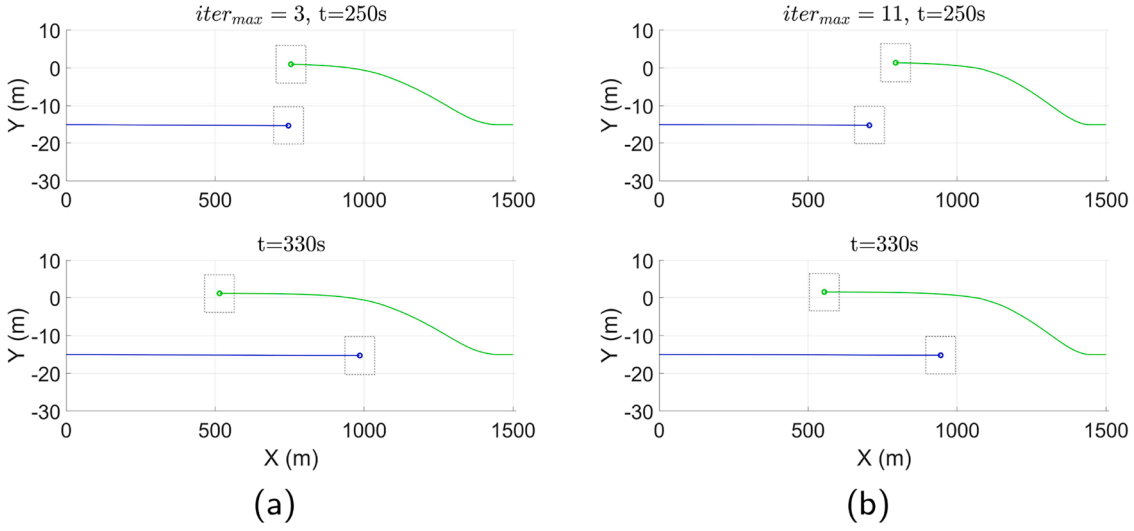


Fig. 9. The solution with different $iter_{max}$. Ships 1 and 2 are illustrated in blue and green dots, respectively. The black dashed rectangles are the safety area of ships.

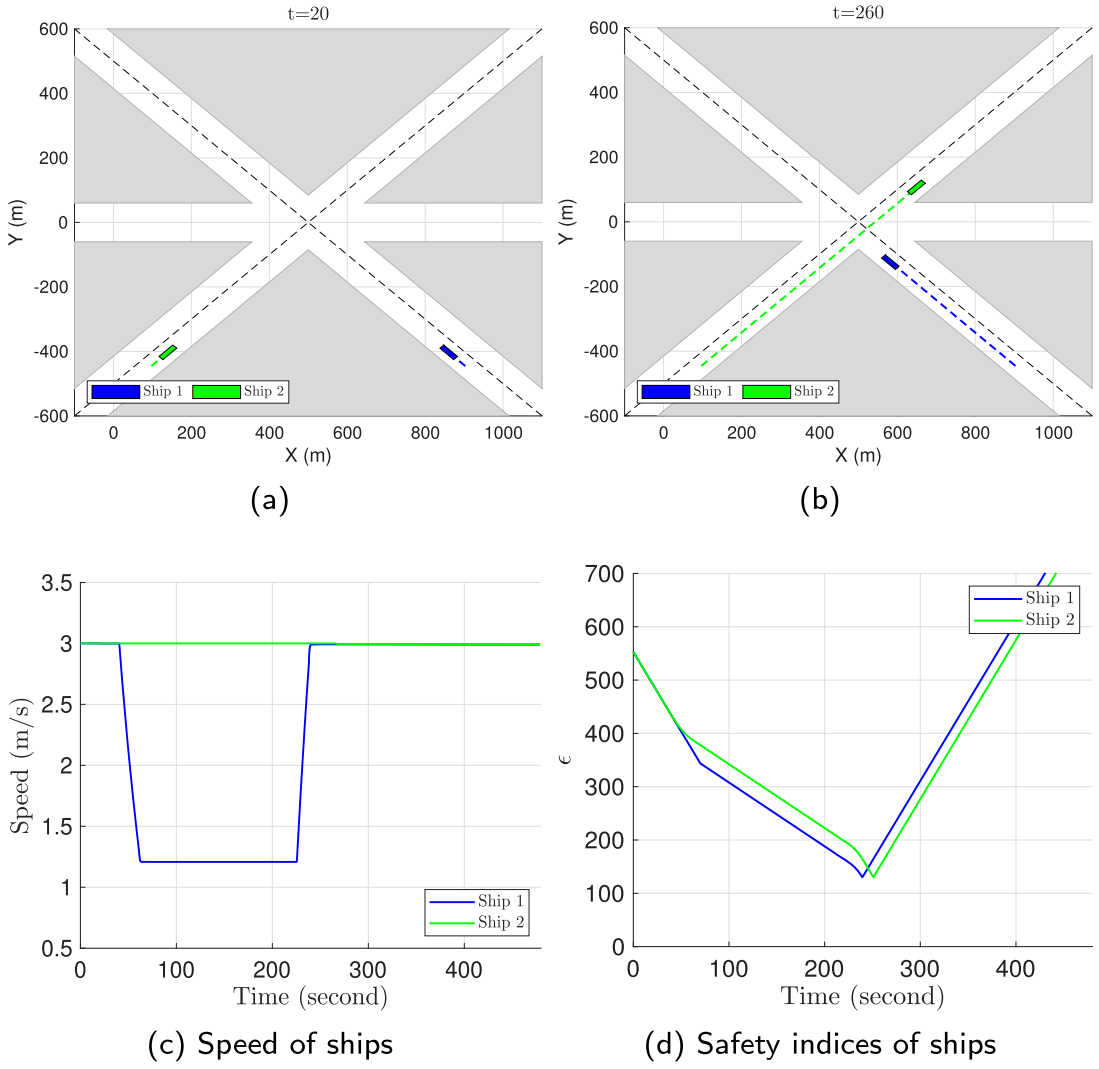


Fig. 10. Intersection crossing between 2 ships. Ship 2 has stand-on priority.

can still handle the situation well (see Fig. 6(b)). Moreover, our algorithm performed well even in combined head-on and overtaking scenarios with a static obstacle, as shown in Fig. 6(c).

5.1.1. Impact of control parameters on behavior of ships

We investigate the behavior of the proposed collaborative collision avoidance algorithm with different control parameters. We use the minimum distance in the x-axis, i.e., the distance in the x-axis over whole experiment, and the stand-on and give-way behavior of two ships as evaluation criteria. The investigated parameters are K_y, K_d, and β as these parameters directly affect the magnitude of the course change of the ship when facing collision risks. We avoid modifying $K_{ca}, \alpha_x, \alpha_y$ since they are directly related to the size or shape of neighboring ships, and $K_s = 2 \times 10^{-2}$. Through trial and error, we found the best parameters used as a reference in this scenario are $K_y = 10^{-2}$, $K_d = 5$, and $\beta = 3 \times 10^{-4}$. As shown in Fig. 7(a), ship 2 takes action early to avoid collision and keep a clear way so that ship 1 does not have to change course.

From Fig. 7(b), if we decrease K_y to 10^{-3}, the stand-on ship tend to overreact with respect to collision risk. Although ship 2 has made a starboard turn at a safe distance, ship 1 still made an unnecessary course change. On the other hand, increasing K_y reduces the minimum distance in the x-axis and delays the time of the first action (see Fig. 7(c) and (d)). These results also coincide with the analysis in Section 4.3. Furthermore, the reduction of the minimum distance in the x-axis in collision avoidance action also causes another unwanted behavior: the changing course of the stand-on ship.

While changing K_y could significantly affect a ship's behavior, changing K_d causes a lesser impact. As shown in Fig. 7(e), K_d increasing four times just slightly reduces the minimum distance in the x-axis. Similarly to K_y, a large value of K_d influences the stand-on ship to change course (see Fig. 7(f)).

Different from K_y and K_d, a change of β does not affect the minimum distance in the x-axis. However, the give-way and stand-on behavior are greatly affected. Reducing β causes a slightly inconsistent behavior of the give-way ship, while the overall performance is the same (see Fig. 7(g)). On the other hand, increasing β persuades the stand-on ship to take action (see Fig. 7(h)) or even swaps the role between stand-on and give-way ship (see Fig. 7(i)).

Depending on specific requirements of CAS, one can adjust the aforementioned parameter to achieve the ship's desired behaviors. However it should be kept in mind that changing the control parameter could, sometimes, result in violating the traffic rules. The behavior is still safe, since it is agreed upon among of both ships.

5.1.2. Impact of the number of iterations

As we see in Eq. (13d), the trajectory prediction that is broadcast to neighboring ships ($\bar{p}_{i,i}^{s+1}$), is not equal to the locally predicted trajectory

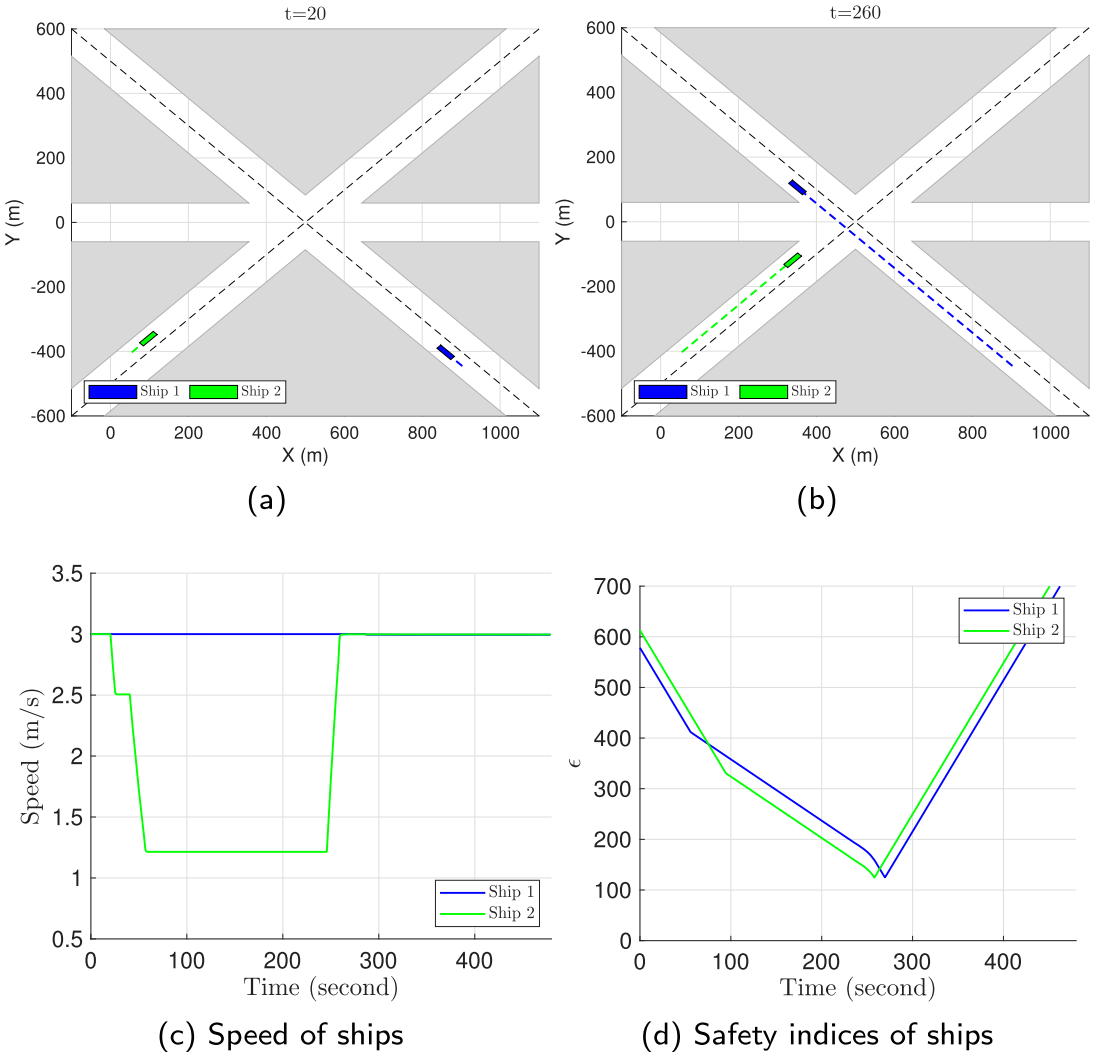


Fig. 11. Intersection crossing between 2 ships. Ship 1 has stand-on priority.

($\tilde{p}_{i,i}^{s+1}$), unless $z_i^{s+1} = 0$. The Lagrange multiplier, z_i^{s+1}, converges to zero as we increase the maximum number of iterations ($iter_{max}$). However, when we increase $iter_{max}$, the computation time also increases. Therefore, to use it in real-time systems, we should balance control performance and computation time. We evaluated the performance of the proposed algorithm with different $iter_{max}$. The results are shown in Fig. 8. It is clear that if we perform only one iteration ($iter_{max} = 1$), the difference between $\bar{p}_{i,i}^{s+1}$ and $\tilde{p}_{i,i}^{s+1}$ is significant (see Fig. 8(a)). The difference significantly decreases as $iter_{max} = 3$, and when $iter_{max} = 11$, the locally predicted trajectory coincides almost completely with the broadcast trajectory prediction. However, because there is no significant difference between the final solution in case $iter_{max} = 3$ compared with $iter_{max} = 11$ (see Fig. 9), we may choose $iter_{max} = 3$ to reduce the unnecessary computation time. It is worth mentioning that, in these simulations, we only consider the broadcast trajectories used within the ADMM scheme. Suppose this information is also used by other actors outside the ADMM scheme, e.g., shore control or manned ships. In that case, we could increase the number of iterations, e.g., $iter_{max} = 6$, to avoid misunderstanding of intention from actors outside the ADMM scheme.

5.2. Intersection crossing scenarios

In this scenario, two or more ships are sailing towards an intersection. The situation is set up in such a way that without a collision avoidance action, all ships shall cross the intersection at the same time. Due to the limited waterway width, a ship cannot change course to avoid collision as would be the case at open sea. Instead, the expected action (for the give-way ship) is to reduce speed. Accordingly, the pair of control parameters K_y and K_u is chosen as $K_y = 5$ and $K_s = 2 \times 10^{-2}$.

In intersection crossing scenarios, it is not easy to visualize the safety area of ships, especially when there are more than two ships. Therefore, we introduce the safety index ϵ_i to evaluate the safety performance of ship i, and is defined as follows:

$$\epsilon_i = \min_{j \in \mathcal{M} \setminus \{i\}} \left\{ \max\{|x_{i,i} - x_{j,i}| - d_i^x, |y_{i,i} - y_{j,i}| - d_i^y\} \right\}.$$

Here, $d_i^x = 51.5$ and $d_i^y = 8.6$ are, respectively, half the length and width of the safety area that are illustrated by the dash rectangle in Fig. 5. When $\epsilon_i > 0$, it means that there is no other ship in the safety area of ship i and there is no risk of collision at that time. When $\epsilon_i \leq 0$, it is likely that a collision will happen. Besides the safety index, we also present in Table 2 the computational time each ship needs to perform an update.

5.2.1. Intersection crossing of two ships

This is the situation in which two ships cross each other. One ship has to give-way to the other by reducing speed. Following the situation shown in Fig. 10(a), ship 2 is sailing on the starboard side of the waterway while ship 1 is sailing on the port side. Therefore, in this case, ship 1 is the give-way ship and makes the first collision avoidance decision. As a result, ship 1 reduces speed and waits for ship 2 to pass the

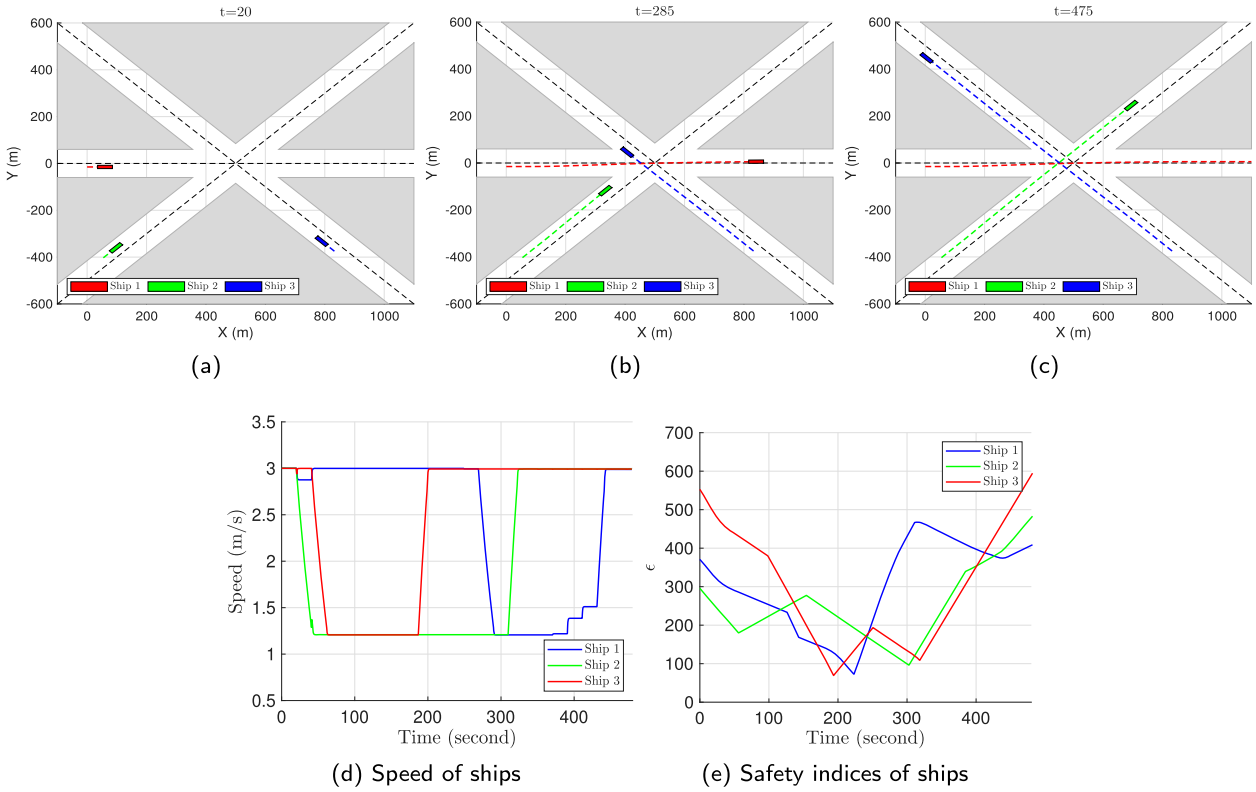


Fig. 12. Intersection crossing between 3 ships.

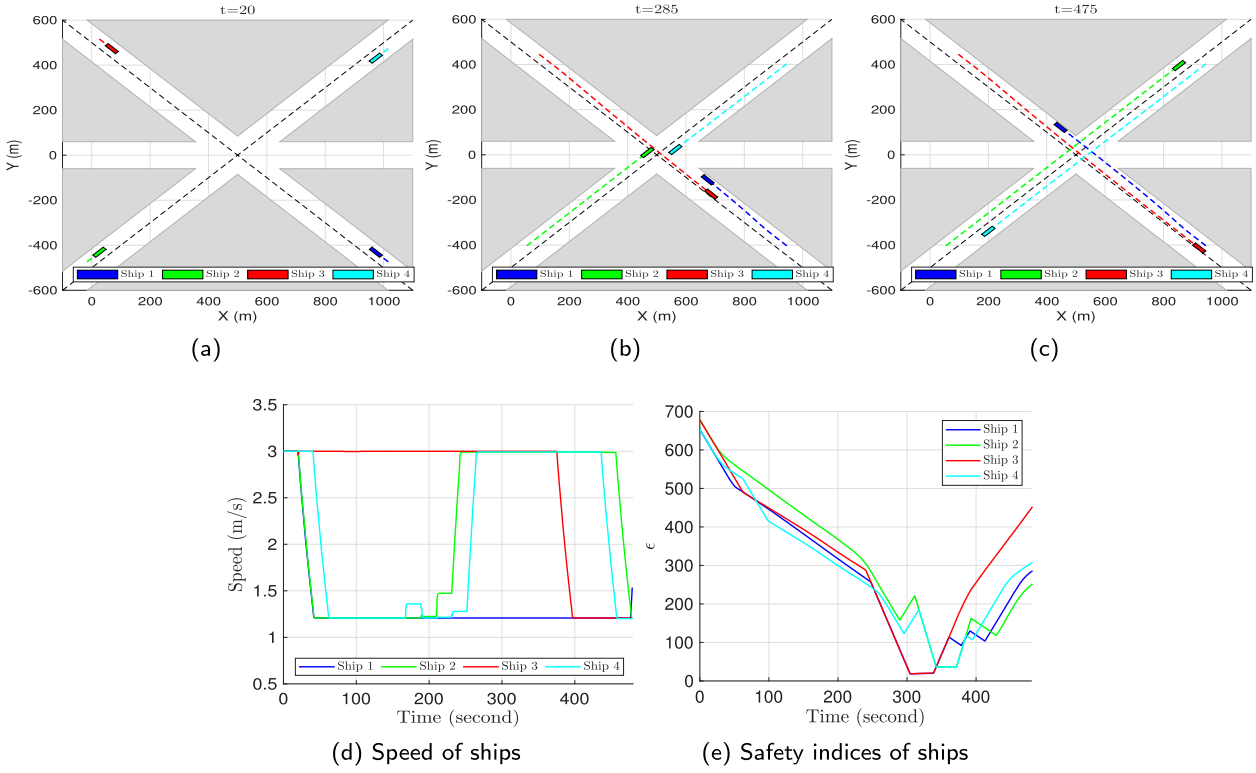
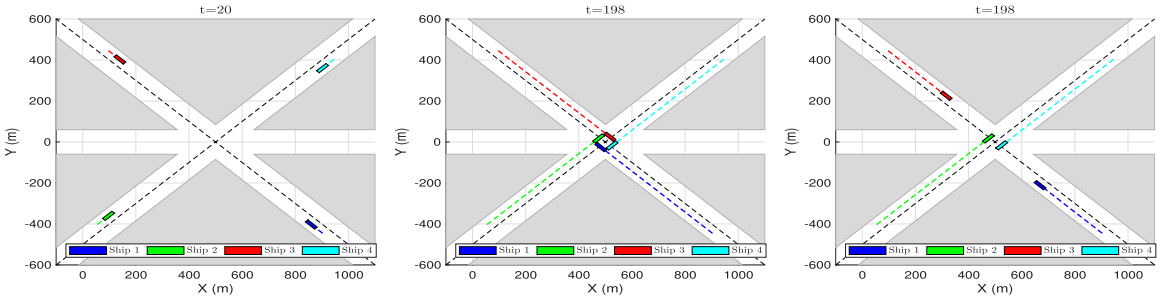


Fig. 13. Intersection crossing between 4 ships.



(a) Deadlock situation. (b) Result without deadlock detection. (c) Result with deadlock detection.

Fig. 14. Intersection crossing between 4 ships in a deadlock situation.

Table 2
Computational time (in seconds) needed to perform an update and the minimum safety index during simulation.

Scenario		Ship 1	Ship 2	Ship 3	Ship 4	Ship 5	Ship 6
2 ships, $N = 45$ (Fig. 10)	Max	0.633	0.615	–	–	–	–
	Mean	0.091	0.093	–	–	–	–
	min(ϵ)	128.787	129.206	–	–	–	–
2 ships, $N = 45$ (Fig. 11)	Max	0.245	0.119	–	–	–	–
	Mean	0.047	0.045	–	–	–	–
	min(ϵ)	125.15	124.57	–	–	–	–
3 ships, $N = 45$ (Fig. 12)	Max	0.986	0.457	0.228	–	–	–
	Mean	0.075	0.064	0.057	–	–	–
	min(ϵ)	72.569	96.036	69.281	–	–	–
4 ships, $N = 45$ (Fig. 13)	Max	0.448	0.454	0.187	0.106	–	–
	Mean	0.123	0.069	0.057	0.062	–	–
	min(ϵ)	17.012	40.977	17.013	40.976	–	–
4 ships, $N = 30$	Max	0.141	0.260	0.233	0.138	–	–
	Mean	0.043	0.091	0.100	0.051	–	–
	min(ϵ)	22.021	27.910	22.019	27.903	–	–
4 ships, $N = 60$	Max	2.275	0.895	0.885	1.158	–	–
	Mean	0.224	0.120	0.091	0.118	–	–
	min(ϵ)	17.349	40.842	17.342	40.841	–	–
4 ships, $N = 45$ (Fig. 14(c))	Max	0.277	0.135	0.117	0.610	–	–
	Mean	0.069	0.075	0.061	0.079	–	–
	min(ϵ)	41.648	41.806	41.647	41.806	–	–
5 ships, $N = 45$	Max	0.880	0.519	0.833	0.863	0.210	–
	Mean	0.273	0.180	0.273	0.222	0.063	–
	min(ϵ)	5.973	25.760	5.990	25.760	93.074	–
6 ships, $N = 45$	Max	0.953	0.448	0.565	0.897	1.373	1.153
	Mean	0.229	0.146	0.179	0.276	0.299	0.389
	min(ϵ)	18.118	19.466	18.124	23.324	50.161	50.157

intersection (see Fig. 10(b)). Fig. 11 shows a similar scenario, but this time, none of the two ships sails on the starboard side of the waterway. Therefore, ship 2 has to give way to ship 1 as ship 1 comes from the starboard side of ship 2 (see Fig. 11(b)).

5.2.2. *Intersection crossing of more than two ships*

Fig. 12 shows a situation with three ships crossing the intersection. According to the traffic rules, ship 1 can stand on, ship 3 gives way to ship 1, and ship 2 gives way to ships 1 and 3. The results show that the solution given by the proposed algorithm follows the traffic regulation. Firstly, as shown in Fig. 12(b), ships 2 and 3 reduce speed to give way to ship 1. When ship 1 crosses the intersection, ship 2 keeps a low speed to give way to ship 3. As shown by the safety indices in Fig. 12(e), all the maneuvers are done safely. In our experiments, we can increase K_y to reduce the unnecessary course change (as ship 1 in Fig. 12(b)). However, increasing K_y also reduces the ability to avoid collision with other ships coming from the opposite direction (head-on situation). Therefore, K_y, in this case, is chosen in such a way that the ship will prioritize reducing speed but can change course when reducing speed cannot resolve the collision avoidance problem.

When more than two ships cross the intersection simultaneously, the collision avoidance problem becomes more complex. In complex situations, ships may disregard the give-way (or stand-on) roles and pursue decisions based on mutual benefits. This is because, in our C-CAS framework, the traffic rules are not hard constraints that force ships to follow. The situation in Fig. 13 is an example of a complex situation in which ships are involved in the head-on and crossing situation simultaneously. As shown in Fig. 13(a), four ships are heading toward the intersection, in which ship 1 and 3 are also in a head-on situation. Ship 1 has the stand-on priority and is supposed to be given way by three other ships. Because K_y is large, following the experiment in Section 5.1, both ship 1 and 3 change course to avoid collision. Ship 3 is the first ship to cross the intersection, followed by ship 2 and 4, and then ship 1. Despite violating the traffic regulation, all the actions of ships are executed in a safe manner, as shown by the safety indices in Fig. 13(e).

In the last experiment, we evaluate the ability to resolve the deadlock situation as mentioned in Section 3.2. Results are given in Fig. 14. We assume that ship 1 is the smallest ship and make the first decision. Without the detection of a deadlock situation, the an accident could happen as shown in Fig. 14(b). However, with the proposed method in Section 3.2, four ships safely cross the intersection (see Fig. 14(c)). Furthermore, although being mentioned in Section 3.1 that a deadlock situation could arise when Algorithm 1 fails to fulfill the condition of consistency, this situation has never been encountered in our simulation experiments.

5.3. Computational performance and scalability

In order to run in a real-time system, it is crucial that the computational time of an update is less than the time between two updates, i.e., the sampling time ΔT. Besides the simulation shown in Figs. 10–14, we performed additional simulations to evaluate the scalability of the C-CAS algorithm when changing the number of participants, $\mathcal{M}$, or the prediction horizon, N.

As shown in Table 2, the computational time required to perform an update is relatively small compared to the sampling time $\Delta T = 20s$. The average time to perform an update is less than 1 s, while the maximum time is less than 4 s. These results suggest that the number of participants can increase to a larger value. However, the specific limit of the number of participants is unclear and needs further investigation. It is worth noting that there is an increase in computational time when the number of ships increases from 2 or 3 ships to 6 ships. This computational time can be reduced by employing a dedicated solver to solve the local optimization problem or by using more efficient software or hardware.

Regarding the prediction horizon N, from Table 2, the computational time increases when N increases and vice versa. It is worth noting that for $N = 60$, computational time increases, but there is little to no improvement in safety performance ($\min(\epsilon)$ of the ships) compared to $N = 45$.

5.4. Choice of K_y and K_s based on traffic scenarios

In Section 5.2.1 and Section 5.2.2, we used the same control parameters for both scenarios, except for two parameters, K_y and K_s. The reason for the difference between K_y and K_s in the two scenarios is that we expect different ship behavior in each scenario. We can use the same parameters (K_y and K_s) for two scenarios, but it will result in the unfavorable behaviors of ships. For example, if we choose $K_y = 2 \times 10^{-2}$ in intersection crossing scenarios, ships tend to make more unnecessary course changes before reducing speed. As discussed in Section 5.2.1, an increase in K_y would decrease the magnitude of the course change. Furthermore, if K_y is large enough, as in Section 5.2.2, the ship would prioritize speed change over course change. In contrast, K_s should be chosen small enough so that the ship can change speed when required. In practice, a ship can automatically switch between two sets of parameters based on the situation determination in steps 10 and 19 of Algorithm 1.

6. Conclusions and future research

This paper presented a two-layer framework for distributed collaborative collision avoidance of autonomous ships in inland waterways. We used the two-layer C-CAS framework to help ships to better comply with traffic regulations. By implementing a decision-making order based on priority, we separated the task of traffic rule compliance from collision avoidance control. The two-layer framework opens the possibility for ships to follow different traffic rules without modifying the control algorithm. Furthermore, we introduced an ADMM-based DMPC algorithm that is designed for inland waterway traffic. The simulation experiments illustrate the performance of our proposed algorithm in various typical traffic scenarios.

In the future, we will focus on improving the limitations of the proposed algorithm, the directions for future studies, including:

- *Increase the resilience of the C-CAS algorithm:* This paper did not consider the case when one ship fails to make a decision or cannot broadcast its decision. Additionally, a method for automatically adjusting the control parameters K_y and K_s based on traffic conditions will be considered.
- *Asynchronous communication scheme:* An asynchronous scheme where ships can make decisions simultaneously is better suited for practical applications.
- *Experiment with different traffic rules/scenarios:* This study only showcases the head-on and intersection crossing scenarios, while the real-world operation requires ships to follow a more complex set of traffic regulations. Further testing with additional traffic regulations would improve the applicability of the C-CAS algorithm. Moreover, in the real world, waterways may not be as straight as depicted in this paper. The Electronic Navigational Charts can be used to develop optimization constraints in such cases (Blindheim and Johansen, 2022). However, a proper solution to deal with irregular waterway boundaries will need further investigation.
- *Comprehensive performance verification:* A framework to evaluate the performance of a C-CAS algorithm must be developed to better compare different C-CAS algorithms.

CRediT authorship contribution statement

Hoang Anh Tran: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization; **Tor Arne Johansen:** Writing – review & editing, Writing - original draft, Validation, Supervision, Project administration, Methodology, Formal analysis, Conceptualization; **Rudy R. Negenborn:** Writing – review & editing, Writing – original draft, Validation, Supervision, Methodology, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 955.768 (MSCA-ETN AUTOBarge), and the Researchlab Autonomous Shipping at Delft University of Technology. This publication reflects only the authors' view, exempting the European Union from any liability. Project website: http://etn-autobarge.eu/.

Appendix A. Proof of Theorem 1

Let us introduce the following notation and definitions that are used in this proof. We denote **range**(M) as the range (column space) of matrix M. The domain of an extended-real-valued function $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ is $\mathbf{dom} f := \{x \in \mathbb{R}^n \mid f(x) < \infty\}$. We also uses the notion of lower semicontinuous function (lsc), image function, and Lipchitz continuous gradient defined as follows:

Definition 1 (lower semicontinuous function). A function $f : X \to \overline{\mathbb{R}}$ is called lower semicontinuous (lsc) at point $x_0 \in X$ if $\liminf_{x \to x_0} f(x) = f(x_0)$. Furthermore, f is called lsc if $f(x)$ is lsc at every point $x_0 \in \mathbf{dom} f$

Definition 2 (image function). Given $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ and $M \in \mathbb{R}^{m \times n}$. Then the image function $(Mf) : \mathbb{R}^m \to [-\infty, +\infty]$ is defined as $(Mf)(\epsilon) := \inf_{x \in \mathbb{R}^n} \{f(x) \mid Mx = \epsilon\}$.

Definition 3 (Lipchitz continuous gradient). A differentiable function h is said to have Lipschitz continuous gradient with constant $L_h > 0$ (or L_h – smooth) on $\mathbf{dom}(h)$ if

$$||\nabla h(x_1) - \nabla h(x_2)|| \le L_h ||x_1 - x_2||, \quad \forall x_1, x_2 \in \mathbf{dom}(h). \tag{A.1}$$

Next, we write the optimization problem of M ships in $\mathcal{M}$, where each individual problem follows (11), in the form of problem (1) with:

$$\begin{aligned} w &= [\tilde{u}_1^\top, \tilde{p}_{1,1}^\top, ..., \tilde{u}_M^\top, \tilde{p}_{M,M}^\top]^\top, \\ v &= [\bar{p}_{1,1}^\top, ..., \bar{p}_{M,M}^\top]^\top, \\ f(w) &= \sum_{i \in \mathcal{M}} \mathcal{J}_i(\tilde{p}_{i,i}, \tilde{u}_i) + \mathcal{G}_i(\tilde{p}_{i,i}, \tilde{u}_i), \\ g(v) &= 0, \\ A &= I_M \otimes \begin{bmatrix} 0_{3(N+1) \times 2N} & I_{3(N+1)} \end{bmatrix}, \\ B &= -I_{3M(N+1)}, \ b = 0, \end{aligned} \tag{A.2}$$

where $\mathcal{G}_i(\tilde{u}_i, \tilde{p}_{i,i})$ is an indicator function that is defined as $\mathcal{G}_i(\tilde{u}_i, \tilde{p}_{i,i}) = 0$ if $[\tilde{u}_i^\top, \tilde{p}_{i,i}^\top]^\top \in \mathbf{G}_i$ and $+\infty$ otherwise. Problem (11) is a special case of problem (1), in which $g(v) = 0$ and $B = -I$. In this case, the NADMM update (2) becomes:

$$\begin{cases} z^{+1/2} &= z - \beta(1-\lambda)(Aw - v - b), \\ w^+ &\in \arg\min \mathcal{L}_\beta(\cdot, v, z^{+1/2}), \\ z^+ &= z^{+1/2} + \beta(Aw - v - b), \\ v^+ &= Aw^+ - b + \frac{1}{\beta} z^+. \end{cases} \tag{A.3}$$

According to (Themelis and Patrinos, 2020, Thm. 5.6), Theorem 1 holds if Problem (11) satisfies (Themelis and Patrinos, 2020, Asm. II):

1. f and g are proper and lsc.
2. A is surjective.
3. (Af) is $L_{(Af)}$ – smooth.
4. (Bg) is lsc.

Since $g(v) = 0$ then it is trivial that g is proper, lsc, an (Bg) is lsc. We also have A is surjective because A is full row rank.

Now we prove that $f(w)$ is lsc. From Problem (11), and (A.2) we also have $f(w) = \sum_{i \in \mathcal{M}} \mathcal{J}_i(\tilde{p}_{i,i}, \tilde{u}_i) + \mathcal{G}_i(\tilde{p}_{i,i}, \tilde{u}_i)$. Taking into account that $\mathcal{G}_i(\tilde{p}_{i,i}, \tilde{u}_i)$ is the indicator function of a closed set then $\mathcal{G}_i(\tilde{p}_{i,i}, \tilde{u}_i)$ is lsc (Hiriart-Urruty and Lemaréchal, 1993, Prop. 1.2.2). Besides, $\mathcal{J}_i(\tilde{p}_{i,i}, \tilde{u}_i)$ is sum of continuous functions hence it is lsc. Therefore, $f(w)$ is lsc.

Following to (Themelis and Patrinos, 2020, Thm. 5.13), (Af) is $L_{(Af)}$ – smooth if there exist $L_{(Af)} > 0$ such that

$$\begin{aligned} -L_{(Af)} ||A(w_1 - w_2)||^2 &\le \langle \nabla f(w_1) - \nabla f(w_2), w_1 - w_2 \rangle \\ &\le L_{(Af)} ||A(w_1 - w_2)||^2 \end{aligned} \tag{A.4}$$

whenever $\nabla f(w_1), \nabla f(w_2) \in \mathbf{range}(A^\top)$.

From (A.2), we have $A = I_M \otimes \begin{bmatrix} 0_{3(N+1) \times 2N} & I_{3(N+1)} \end{bmatrix}$, in which case $\nabla f(w) \in \mathbf{range}(A^\top)$ if and only if $\tilde{u}_i = 0, \forall i \in \mathcal{M}$ and $\mathcal{G}_i = 0, \forall i \in \mathcal{M}$. Then $f(w) = \sum_{i \in \mathcal{M}} \mathcal{J}_i^{ca}(\tilde{p}_{i,i})$ is Lipschitz continuous gradient function because $R_{ij}(\cdot)$ is L_R – smooth. This implies that exist L_f such that

$$||\nabla f(w_1) - \nabla f(x_2)|| \le L_f ||w_1 - w_2||.$$

Using the Cauchy-Schwartz inequality we obtain

$$\begin{aligned} L_f ||w_1 - w_2||^2 &= L_f ||A(w_1 - w_2)||^2 \\ &\ge |\langle \nabla f(w_1) - \nabla f(w_2), w_1 - w_2 \rangle| \\ &\ge |\nabla f(w_1) - \nabla f(x_2)| . |w_1 - w_2|, \end{aligned}$$

for all $\nabla f(w) \in \mathbf{range}(A^\top)$. Then there exist $L_{(Af)}$ satisfy (A.4). □

References

Akdağ, M., Fossen, T.I., Johansen, T.A., 2022a. Collaborative collision avoidance for autonomous ships using informed scenario-based model predictive control. IFAC-PapersOnLine 55 (31), 249–256. https://doi.org/10.1016/j.ifacol.2022.10.439

Akdağ, M., Solnør, P., Johansen, T.A., 2022b. Collaborative collision avoidance for maritime autonomous surface ships: a review. Ocean Eng. 250, 110920.

Andersson, J. A.E., Gillis, J., Horn, G., Rawlings, J.B., Diehl, M., 2019. CasADi – A software framework for nonlinear optimization and optimal control. Math. Program. Comput. 11 (1), 1–36. https://doi.org/10.1007/s12532-018-0139-4

Blindheim, S., Johansen, T.A., 2022. Electronic navigational charts for visualization, simulation, and autonomous ship control. IEEE Access 10, 3716–3737.

BPR, 2017. Binnenvaartpolitiereglement. https://wetten.overheid.nl/BWBR0003628/2017-01-01.

CCNR, 2023. Central commission for the navigation of the Rhine - CCNR regulations. https://www.ccr-zkr.org/13020500-en.html.

Chen, L., Hopman, H., Negenborn, R.R., 2018a. Distributed model predictive control for vessel train formations of cooperative multi-vessel systems. Transp. Res. Part C-Emerging Technol. 92, 101–118.

Chen, L., Negenborn, R.R., Hopman, H., 2018b. Intersection crossing of cooperative multi-vessel systems. IFAC Papersonline 51 (9), 379–385. https://doi.org/10.1016/j.ifacol.2018.07.062

Du, Z., Negenborn, R.R., Reppa, V., 2022. Colregs-compliant collision avoidance for physically coupled multi-vessel systems with distributed mpc. Ocean Eng. 260. https://doi.org/10.1016/j.oceaneng.2022.111917

Eriksen, B.O.H., Bitar, G., Breivik, M., Lekkas, A.M., 2020. Hybrid collision avoidance for ASVs compliant with COLREGs Rules 8 and 13-17. Front. Rob. AI 7. https://doi.org/10.3389/frobt.2020.00011

Ferranti, L., Lyons, L., Negenborn, R.R., Keviczky, T., Alonso-Mora, J., 2022. Distributed nonlinear trajectory optimization for multi-robot motion planning. IEEE Trans. Control Syst. Technol. 31(2), 809–824.

Ferranti, L., Negenborn, R.R., Keviczky, T., Alonso-Mora, J., 2018. Coordination of multiple vessels via distributed nonlinear model predictive control. In: 2018 European Control Conference (ECC). IEEE, Limassol, Cyprus, pp. 2523–2528.

Fossen, T.I., 2011. Handbook of Marine Craft Hydrodynamics and Motion Control. John Wiley & Sons.

Gan, L., Yan, Z., Zhang, L., Liu, K., Zheng, Y., Zhou, C., Shu, Y., 2022. Ship path planning based on safety potential field in inland rivers. Ocean Eng. 260, 111928. https://doi.org/10.1016/j.oceaneng.2022.111928

Guiking, C., 2022. Digital intention sharing : simulation study on the benefits of intention sharing. https://open.rijkswaterstaat.nl/overige-publicaties/2022/digital-intention-sharing-simulation/.

Hiriart-Urruty, J.-B., Lemaréchal, C., 1993. Convex analysis and minimization algorithms I. Vol. 305 of *Grundlehren der mathematischen Wissenschaften*. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-02796-7

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 Trans. Intell. Syst. Technol. 17 (12), 3407–3422. https://doi.org/10.1109/Tits.2016.2551780

Kim, D., Hirayama, K., Okimoto, T., 2017. Distributed stochastic search algorithm for multi-ship encounter situations. J. Navig. 70 (4), 699–718. https://doi.org/10.1017/S037346331700008X

Kneissl, M., Molin, A., Esen, H., Hirche, S., 2018. A feasible MPC-based negotiation algorithm for automated intersection crossing. 2018 European Control Conference (Ecc) , 1282–1288.

Kurowski, M., Roy, S., Gehrt, J.J., Damerius, R., Buskens, C., Abel, D., Jeinsch, T., 2019. Multi-vehicle guidance, navigation and control towards autonomous ship maneuvering in confined waters. 2019 18th European Control Conference (Ecc) , 2559–2564.

Lutz, A., Gilles, E.-D., 2010. An automatic collision detection and avoidance module for inland navigation. IFAC Proc. Volumes 43 (20), 254–259.

Mahipala, D., Johansen, T.A., 2023. Model Predictive Control for Path Following and Collision-Avoidance of Autonomous Ships in Inland Waterways. In: 2023 31st Mediterranean Conference on Control and Automation (MED), pp. 896–903. https://doi.org/10.1109/MED59994.2023.10185796

Negenborn, R.R., Maestre, J.M., 2014. Distributed model predictive control an overview and roadmap of future research opportunities. IEEE Control Syst. Mag. 34 (4), 87–97. https://doi.org/10.1109/Mcs.2014.2320397

Schwarting, W., Alonso-Mora, J., Pauli, L., Karaman, S., Rus, D., 2017. Parallel autonomy in automated vehicles: safe motion generation with minimal intervention. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 1928–1935.

STM, 2015. Voyage exchange format and architecture, Monalisa 2 D1.3.2. STM Monalisa. https://stm-stmvalidation.s3.eu-west-1.amazonaws.com/uploads/20160420144429/ML2-D1.3.2-Voyage-Exchange-Format-RTZ.pdf.

Tam, C., Bucknall, R., 2013. Cooperative path planning algorithm for marine surface vessels. Ocean Eng. 57, 25–33. https://doi.org/10.1016/j.oceaneng.2012.09.003

Tang, C., Zhang, H., Wang, J., 2022. Flexible formation tracking control of multiple unmanned surface vessels for navigating through narrow channels with unknown curvatures. IEEE Trans. Ind. Electron. 70(3), 2927–2938.

Tengesdal, T., Johansen, T.A., Grande, T.D., Blindheim, S., 2022. Ship collision avoidance and anti grounding using parallelized cost evaluation in probabilistic scenario-based model predictive control. IEEE Access 10, 111650–111664.

Themelis, A., Patrinos, P., 2020. Douglas-Rachford splitting and ADMM for nonconvex optimization: tight convergence results. SIAM J. Optim. 30 (1), 149–181. https://doi.org/10.1137/18m1163993

Tran, H.A., Johansen, T.A., Negenborn, R.R., 2023a. A collision avoidance algorithm with intention prediction for inland waterways ships. IFAC-PapersOnLine 56 (2), 4337–4343. https://doi.org/10.1016/j.ifacol.2023.10.1805

Tran, H.A., Johansen, T.A., Negenborn, R.R., 2023b. Collision avoidance of autonomous ships in inland waterways - a survey and open research problems. J. Phys. Conf. Ser. 2618 (1), 012004. Publisher: IOP Publishing. https://doi.org/10.1088/1742-6596/2618/1/012004

Wächter, A., Biegler, L.T., 2004. On the Implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. 106 (1), 25–57.

Yan, X., Wang, S., Ma, F., Liu, Y., Wang, J., 2020. A novel path planning approach for smart cargo ships based on anisotropic fast marching. Expert Syst. Appl. 159, 113558.

Yu, S., Hirche, M., Huang, Y., Chen, H., Allgöwer, F., 2021. Model predictive control for autonomous ground vehicles: a review. Auton. Intell. Syst. 1 (1), 4. https://doi.org/10.1007/s43684-021-00005-z

Zhang, G., Wang, Y., Liu, J., Cai, W., Wang, H., 2022. Collision-avoidance decision system for inland ships based on velocity obstacle algorithms. J. Marine Sci. Eng. 10 (6). https://doi.org/10.3390/jmse10060814

Zheng, H., Negenborn, R.R., Lodewijks, G., 2016. Predictive path following with arrival time awareness for waterborne AGVs. Transp. Res. Part C-Emerging Technol. 70, 214–237. https://doi.org/10.1016/j.trc.2015.11.004

Zheng, H., Negenborn, R.R., Lodewijks, G., 2017. Fast ADMM for distributed model predictive control of cooperative waterborne AGVs. IEEE Trans. Control Syst. Technol. 25 (4), 1406–1413.