Smooth Scenario-Based Model Predictive Control for Autonomous Collision Avoidance in Inland Waterways

Dhanika Mahipala[1] and Tor Arne Johansen[2], *Senior Member, IEEE,*

***Abstract*—The Scenario-Based Model Predictive Control (SB-MPC) is an autonomous collision avoidance algorithm primarily designed for open and coastal waters. One of the challenges in adapting SB-MPC for autonomous inland waterway collision avoidance is the inability to use a derivative based optimization strategy due to non-smooth components in its cost function. Hence, we propose a novel algorithm, Smooth Scenario-Based Model Predictive Control (Smooth-SBMPC) specifically designed for highly constrained and complex navigational environments inherent to inland waterways. The effectiveness of Smooth-SBMPC is validated through a comprehensive simulation study, providing insights into its performance in complex navigational environments.**

***Index Terms*—Maritime collision avoidance, model predictive control, autonomous ships, Inland waterways, multi-layer optimization.**

I. Introduction

An algorithm designed for autonomous vessel navigation and collision avoidance in inland waterways should address the following aspects [1]:

- Should be able to avoid collisions with target vessels, riparian land (grounding hazard on either side of the water body e.g., riverbank) and static obstacles (reeks, wrecks, navigational aids, fishing zones and small islands etc.) without deviating unnecessarily from the original path.
- Should be compliant with local traffic rules or the Convention on the International Regulations for Preventing Collisions at Sea (COLREGS) [2] (A brief overview of the main rules related to collision avoidance can be found in APPENDIX I).
- Should be able to easily adapt secondary performance criteria.

There are numerous research results found in the literature that proposes different methods to address one or more of these aspects [3]–[5]. The Scenario-based Model Predictive Control (SB-MPC) method presented in [6], is such an algorithm that has proven to be successful over the years as a reactive collision-avoidance method based on prediction and receding horizon optimization in open and coastal waters. Since the initial introduction, many modifications have been suggested to improve different aspects of SB-MPC by various authors such as, [7]–[9] to name a few. The SB-MPC has a comprehensive cost function that includes cost considerations for COLREGS rules violation, maneuvering effort and collision risk. As proved in the cited articles above, SBMPC is a flexible algorithm that can be re-configured to account for issues that were not addressed in the initial implementation.

However, one of the main challenges in utilizing SB-MPC for autonomous inland waterway navigation is the resolution limitations that arise when deriving the optimal control inputs from a finite set of discretized solutions [1]. To that extent, previous literature on SB-MPC has not adequately tackled the issue. As a solution, a numerical optimization strategy can be employed to find the optimal control inputs instead of using exhaustive search method as proposed in [6]. In literature, similar optimization problems are handled by transcribing the Optimal Control Problem (OCP) to a Non-Linear Programming Problem (NLP) by discretization, and solving the NLP using a gradient optimization scheme [10], [11]. However, the cost function used in SB-MPC cannot be optimized using a gradient-based optimization algorithm due to non-smoothness in some of the cost components (such as the COLREGS cost) in the cost function. One solution to this problem is stochastic optimization [12], [13]. However, unlike gradient-based optimization strategies, stochastic-optimization methods do not guarantee local optimality. Another approach found in literature is the hybrid control system architecture [1], [14]–[16]. In these type of approaches, generally, several different algorithms are combined in a layered architecture, where each layer is responsible for a different task in the collision avoidance process. Hence, typical non-smooth costs such as COLREGS violation cost can be handled by an algorithm that uses a derivative-free optimization strategy. However, this implies that the control objective at different layers are different thus, creating the possibility of dissension between algorithms. Therefore, instead, we propose modifying/replacing the non-smooth cost components of the SB-MPC cost function in order to be able to use a gradient based optimization strategy.

An important step of gradient based optimization is the initialization step. A good initialization strategy is important to handle non-convex constraints and nonlinear dynamics in the OCP [17]. It also helps converge to the same locally optimal solution given that external factors remain constant, thus, improving the consistency in the overall performance of the algorithm. In addition, if the initialization trajectory is feasible, the OCP is guaranteed to find an optimal solution at least as good as the sub-optimal initialization trajectory. In the papers [18], [19], the authors present a similar concept where a two-step optimization strategy is proposed for automatic maneuvering of ships in cluttered environments. In this research, the authors use a state lattice [20] to generate a resolution optimal solution to a motion planning problem which is then used to warm-start the receding-horizon optimization based second step. However, this method could be less favorable

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

[1]Cybernetics and Simulation - ISOL, Kongsberg Maritime AS, Norway. (Email: dhanika.mahipala@km.kongsberg.com)

[2]Department of Engineering Cybernetics, Norwegian University of Science and Technology, Norway. (Email: tor.arne.johansen@ntnu.no)

having to pre-generate motion primitives to build the lattice structure. In addition, the cost function formulation is different and less concise to that of SB-MPC.

In addition, we require a method to find the navigable area for the vessel that is free of collisions with riparian land and other static obstacles. There are different approaches in the literature, where some propose to include the static obstacle avoidance as part of the objective function (for example using Artificial Potential Fields (APF) [21]) while others propose to use constraints [22]. We believe using constraints is a more direct and natural representation of the avoidance requirement. In authors' previous publication [1], the concept of a *cross-track error corridor* was proposed where, the edges of the riparian land were used as constraints of the MPC. The algorithm requires a path to generate the corridor where in the paper, the path connecting two adjacent way points was used. The obstacles that intersect with this path are considered as other static obstacles despite being part of the riparian land. The constraints for such obstacles were proposed to be derived by assuming circular regions around the obstacle and the ship (ship domain). Since this is not ideal in confined and complex environments, there is a need to propose a method to produce a better trajectory for the cross-track error corridor algorithm to generate the constraints.

Hence, this research aims to address three primary research objectives:

- Modifying/replacing non-smooth components of the SB-MPC cost function with smooth ones to be able to use a gradient-based optimization strategy. This will in turn solve the resolution limitation problem discussed.
- Propose an initialization method to warm-start the gradient-based optimization scheme.
- Develop a method to generate a trajectory that can be used to generate the cross-track error corridor at each time step.

To that extent, a novel autonomous navigation and collision avoidance algorithm suitable for inland waterways is proposed. From this point forward it will be called Smooth Scenario-Based Model Predictive Control (Smooth-SBMPC).

II. Overview

This section aims to provide an overview of the methodology of this research. The Smooth-SBMPC algorithm consists of a two stage optimization strategy. At the first stage i.e., initialization stage, the SB-MPC algorithm [6] is used to derive a collision free sub-optimal trajectory for the own ship. The non-smooth components of the cost function in the original SB-MPC will either be modified or replaced. At the next stage, a Model Predictive Controller (MPC) will be employed to generate the resolution sufficient solutions. The trajectory from the initialization stage will be used for,

- Deriving local spatial constraints (cross-track error corridor).
- Warm-starting the MPC.

The expected outputs from the MPC are modifications to course and speed commands generated by a guidance controller (similar to the original SB-MPC). Once, the modifications are applied, these commands can be directed to an autopilot (the design of the autopilot is beyond the scope of this research) to drive the steering and propulsion systems of the vessel accordingly.

III. SB-MPC Background

In this section, we present the SB-MPC [7] algorithm along with its associated cost components utilized in this research. The goal is to provide the reader with a comprehensive understanding of the algorithm, serving as the groundwork for the modifications discussed in the subsequent sections.

In SB-MPC, the input to the autopilot of the vessel is parameterized as a course command (χ_d) and a speed command (U_d). The main objective of the algorithm is to calculate course and speed modifications (χ_m and U_m) that would aid the vessel to avoid hazardous situations. These modifications are selected from discretized solution sets where a minimum in a typical implementation would look like the following:

- Course offset in degrees (χ_m): -90, -75, -60, -45, -30, -15, 0, 15, 30, 45, 60, 75, 90.
- Speed modification factors (U_m): 0, 0.5, 1

A combination of a course modification and a speed modification is defined as a **control behavior** $k \in \{1, 2, \ldots, N_s\}$ where, N_s stands for the total number of control behaviors. The two course and speed modification sets stated above leads to a total of $N_s = 13 \times 3 = 39$ control behaviors.

To find the optimal course and speed modifications (χ_m^* and U_m^*), each control behavior k is evaluated at discrete sample times over the prediction horizon T using the discretization interval T_s, as $D(\tau_0) = \{\tau_0, \tau_0 + T_s, \ldots, \tau_0 + T\}$, where τ_0 is the current time. The optimal control behavior k^* which minimizes the objective function defined in (1), gives the optimal χ_m^* and U_m^*.

$$k^*(\tau_0) = \arg\min_k \; H^k(\tau_0) \tag{1}$$

where,

$$\begin{aligned} &H^k(\tau_0) \\ &= \max_i \max_{\tau \in D(\tau_0)} \left(C_i^k(\tau) R_i^k(\tau) + \kappa_i \mu_i^k(\tau) + \lambda_i T_i^k(\tau)\right) \\ &\quad + \Gamma(U_m^k, \chi_m^k) \end{aligned} \tag{2}$$

The terms of the cost function in Equation (2) are defined as follows,

- The cost associated with collision with obstacle i at time τ in control behavior k, is $C_i^k(\tau)$, and the corresponding collision risk factor is $R_i^k(\tau)$.
- The COLREGS rules violation cost is $\kappa_i \mu_i^k(\tau)$, where $\mu_i^k(\tau)$ is a binary indicator of COLREGs rule violation and κ_i is a positive tuning parameter.
- The COLREGS-transitional cost which penalize abandonment of a COLREGS compliant maneuver is denoted by $\lambda_i T_i^k(\tau)$, where T_i^k is a binary indicator and λ_i is a positive tuning parameter.
- The cost of maneuvering effort associated with control behavior k is given by $\Gamma(U_m^k, \chi_m^k)$.

A. Collision Cost

The collision cost associated with the ith obstacle if using control behavior k at time τ can be calculated as,

$$R_i^k(\tau) = \begin{cases} \frac{1}{|\tau - \tau_0|^p} \left(\frac{d_i^{safe}}{d_{o,i}^k(\tau)}\right)^q, & \text{if } d_{o,i}^k(\tau) \leq d_i^{safe} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

and,

$$C_i^k(\tau) = K_i^{coll} \left|\vec{v}_o^k(\tau) - \vec{v}_i^k(\tau)\right|^2 \tag{4}$$

where, τ_0 is current time and $\tau > \tau_0$ is the time of prediction. The parameter $d^k_{o,i}(\tau)$ is the predicted distance between own ship and obstacle i at time τ in control behavior k. Typical choices for the exponents q (>1) and p (>1/2) are $q = 4$ and $p = 1$. The parameters $\vec{v}^k_o(\tau)$ and $\vec{v}^k_i(\tau)$ represent the predicted velocity of own ship and obstacle i at time τ in control behavior k. Moreover, d^{safe}_i should take into account COLREGS by ensuring sufficient safety distance to ships.

B. COLREGS Cost

The SB-MPC COLREGS cost has two factors, κ_i and $\mu^k_i(\tau)$, from which $\kappa_i > 0$ is a constant that can be used as a tuning parameter and $\mu^k_i(\tau)$ which gives a binary true output if the the own ship violates COLREGS rules 14 or 15 with the obstacle i at time τ in control behavior k. The variable $\mu^k_i(\tau)$ can be defined as,

$$\mu^k_i(\tau) = RULE14 \; or \; RULE15 \tag{5a}$$

$$RULE14 = CLOSE \;\&\; STARBOARD \;\&\; HEAD-ON \tag{5b}$$

$$RULE15 = CLOSE \;\&\; STARBOARD \;\&\; CROSSED \;\&\; NOT\,OVERTAKEN \tag{5c}$$

where,

- the obstacle i is said to be CLOSE to own ship at time τ in control behavior k if,

$$d^k_{o,i}(\tau) < d^{cl}_i \tag{6}$$

  where, d^{cl}_i is the smallest distance where the COLREGS responsibility for stay away is considered to apply.
- the own ship is said to be OVERTAKEN by the obstacle i at time τ in control behavior k if,

$$\vec{v}^k_o(\tau) \cdot \vec{v}^k_i(\tau) > \cos(67.5^\circ) \left|\vec{v}^k_o(\tau)\right| \left|\vec{v}^k_i(\tau)\right| \tag{7a}$$

$$|\vec{v}^k_i(\tau)| > |\vec{v}^k_o(\tau)| \tag{7b}$$

- the obstacle i is said to be STARBOARD of own ship at time τ in control behavior k if,

$$\text{bearing angle} > \text{heading angle of the own ship} \tag{8}$$

- the obstacle i is said to be HEAD-ON at time τ in control behavior k if,

$$\left|\vec{v}^k_o(\tau)\right| \text{ is not close to zero} \tag{9a}$$

$$\vec{v}^k_o(\tau) \cdot \vec{v}^k_i(\tau) < -\cos(22.5^\circ) \left|\vec{v}^k_o(\tau)\right| \left|\vec{v}^k_i(\tau)\right| \tag{9b}$$

$$\vec{v}^k_o(\tau) \cdot \vec{L}^k_i(\tau) > \cos(\phi_{ahead}) \left|\vec{v}^k_o(\tau)\right| \tag{9c}$$

  where $\vec{L}^k_i(\tau)$ is a unit vector in the Line-of-sight (LOS) direction from own ship to the obstacle with index i at time τ in control behavior k. The parameter ϕ_{ahead} can be selected appropriately. In this implementation we have set this to 67.5°.
- the obstacle i is said to be CROSSED at time τ in control behavior k if,

$$\vec{v}^k_o(\tau) \cdot \vec{v}^k_i(\tau) < \cos(67.5^\circ) \left|\vec{v}^k_o(\tau)\right| \left|\vec{v}^k_i(\tau)\right| \tag{10}$$

C. COLREGS-transitional cost

First introduced in [7], the COLREGS-transitional cost penalizes control behaviors that abort one COLREGS-compliant maneuver for another. It is formulated with the use of the binary indicator $T^k_i \in \{0, 1\}$ as,

$$T^k_i(\tau) = O^k_i(\tau) \vee Q^k_i(\tau) \vee X^k_i(\tau) \tag{11}$$

where, the binary indicators $O^k_i(\tau) = 1$, $Q^k_i(\tau) = 1$ and $X^k_i(\tau) = 1$ indicate the type of situation (the own ship is overtaking a vessel, the own ship is being overtaken and crossing situation, respectively) at time τ, and that the control behaviour k will at time τ cause the vessels to pass each toher on the side opposite to what is predicted with the current control behaviour.

D. Maneuvering Cost

The maneuvering cost can be calculated as,

$$\Gamma(U^k_m, \chi^k_m) = k_U(1 - U^k_m) + k_\chi {\chi^k_m}^2 + \Delta_U(U^k_m - U^k_{m,last}) + \Delta_\chi(\chi^k_m - \chi^k_{m,last}) \tag{12}$$

where, Δ_U and Δ_χ are penalty functions that are zero at the origin while k_U (> 0) and k_χ (> 0) are scalar tuning parameters. The parameters k_χ and Δ_χ are generally asymmetric and give a higher penalty on course commands to port than starboard, in compliance with COLREGS rules 14, 15 and 17.

IV. Ship Modeling

In this section we introduce the internal prediction models of the own ship and the target ships used for calculating the state at the next time step.

A. Own ship model

For this research we have decided to adapt the kinematic model used in [23] as,

$$\begin{aligned} x_{\tau+\delta\tau} &= x_\tau + U_\tau \cos(\chi_\tau)\delta\tau \\ y_{\tau+\delta\tau} &= y_\tau + U_\tau \sin(\chi_\tau)\delta\tau \\ \chi_{\tau+\delta\tau} &= \chi_\tau + \frac{\delta\tau}{T_\chi}(\chi_{input,\tau} - \chi_\tau) \\ U_{\tau+\delta\tau} &= U_\tau + \frac{\delta\tau}{T_U}(U_{input,\tau} - U_\tau) \end{aligned} \tag{13}$$

which describes the own ship state, $\mathbf{x}_\tau = [x_\tau, y_\tau, \chi_\tau, U_\tau]^T$ at time τ, that consists of East and North position in Cartesian local coordinates, course over ground (COG), and speed over ground (SOG). T_χ and T_U are course and speed time constants. The $\chi_{input,\tau}$ and $U_{input,\tau}$ i.e., $\mathbf{u}_\tau$ are the course and speed commands that serve as the inputs to the system at time τ. The parameter $\delta\tau$ is the discretization interval.

B. Target ship model

Other vessels in the vicinity of the own ship are identified as target ships. For these, we are using the Constant Velocity Model (CVM) as follows,

$$\begin{aligned} x^{target}_{\tau+\delta\tau} &= x^{target}_\tau + U^{target}_\tau \cos\left(\chi^{target}_\tau\right)\delta\tau \\ y^{target}_{\tau+\delta\tau} &= y^{target}_\tau + U^{target}_\tau \sin\left(\chi^{target}_\tau\right)\delta\tau \end{aligned} \tag{14}$$

where, $\mathbf{x}^{target}_\tau = [x^{target}_\tau, y^{target}_\tau, \chi^{target}_\tau, U^{target}_\tau]^T$ describes the target ship state at time τ and the parameter $\delta\tau$ is the discretization interval. Essentially, a straight-line trajectory is assumed for the target vessels.

V. Modified Smooth-SBMPC Cost

The objective of this section is to convert the non-smooth cost components of the SBMPC cost function (2) so that it can be optimized using a gradient optimization scheme. Different strategies have been utilized for each cost component, which will be explained separately in the following sub-sections. A key tool used in these strategies is the sigmoid function defined in (15).

$$\sigma(z) = a_3 + \frac{a_4}{1 + e^{-a_2(z-a_1)}} \tag{15}$$

where a_1, a_2, a_3 and a_4 are constants that can be utilized to adjust the shape of the sigmoid function according to the requirement.

A. Collision Cost

The collision cost component in (2), has a discontinuity in its collision risk factor (3). Therefore, the equation is modified by introducing the sigmoid function defined in (15) as,

$$\text{Modified collision cost} \Rightarrow \sigma_{coll}(d_{o,i}(\tau))C_i(\tau)R_i(\tau) \tag{16}$$

where,

$$R_i(\tau) = \frac{1}{|\tau - \tau_0|^p}\left(\frac{d_i^{safe}}{d_{o,i}(\tau)}\right)^q \tag{17}$$

and,

$$C_i(\tau) = K_i^{coll}\left|\vec{v}_o(\tau) - \vec{v}_i(\tau)\right|^2 \tag{18}$$

The scaling parameters of σ_{coll} can be denoted as $a_1 = d_i^{safe}, a_3 = 0, a_4 = 1$. The parameter a_2 can be used to adjust the slope of the curve. We note that the superscript k has been removed in the above equations to denote that it is not limited to a discretized set of control behaviors when evaluating the respective cost.

B. COLREGS Cost

The original COLREGS cost component in (5) is formulated using Boolean Logic, which results in a non-smooth cost function. Therefore, we propose replacing Boolean Logic with smooth Fuzzy Logic. The rest of the sub-section will explain how the said cost component is modified/converted using a Fuzzy Inference System (FIS). However, this section does not repeat the theory of Fuzzy Logic and its implementation as it is well documented in literature. Instead, the readers are referred to articles such as [24], [25]. For each fuzzy expert system explained here, standard Mamdani fuzzy inference [26] is used to execute the designs.

Before delving into the FIS that replaces the binary logic defined in (5) i.e., $FIS_{colregs}$, two parameters need to be defined. These are, the course angle difference ($\Delta COG_i(\tau)$), and the Relative Bearing Angle ($RB_i(\tau)$) of the obstacle i at time τ. These parameters can be represented as shown in Figure 1.

Using $\Delta COG_i(\tau)$, the conditions stated in Equations (7a), (9b) and (10) can be represented as shown in Figure 2. Therefore, $\Delta COG_i(\tau)$ can be taken as an input to the $FIS_{colregs}$ using the linguistic variable $DeltaCOG$. It consists three membership functions to represent *Overtaking* (OT), *Crossing* (CR) and *Head-on* (HO). As illustrated in Figure 2, the input to $DeltaCOG$ range between $[0, 180°]$. The crisp ranges of OT ($0°-67.5°$), CR ($67.5°-157.5°$) and HO ($157.5°-180°$) are extended by approximately $3°$, $5°$ and $5°$ respectively. The Table I summarizes the details of the linguistic variable

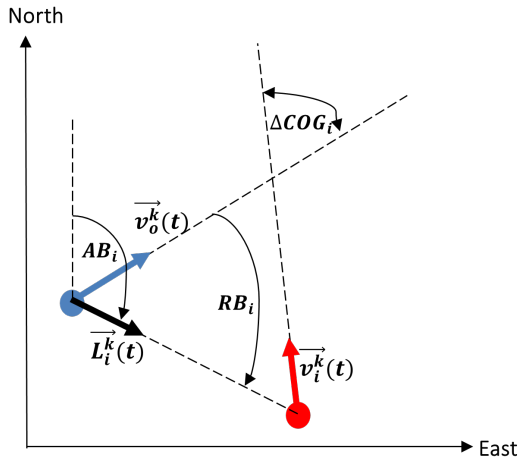


Fig. 1: The figure depict the parameters $\Delta COG_i(\tau)$ and $RB_i(\tau)$. The parameter AB_i is the actual bearing angle of the obstacle i (red) relative to the own ship (blue).

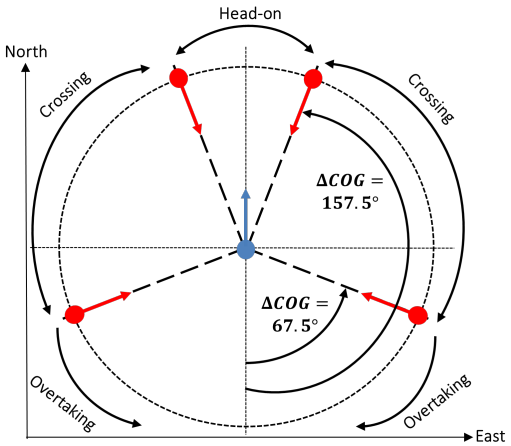


Fig. 2: Different collisions situation identified in Equations (7a),(9b) and (10) using $\Delta COG_i(\tau)$. The blue dot and the arrow represent the own ship and its course direction while red represent the target vessels.

$DeltaCOG$. In the table, the function type *sigmf* represent Sigmoidal Membership Function while *dsigmf* represents the difference between two Sigmoidal Functions, and the parameters a_1, a_2, a_3, a_4 are defined in Equation (15).

	Label	Type	Parameters P $[a_1, a_2, a_3, a_4]$ (unit)
$DeltaCOG$	OT	sigmf	$[67.5, -1.5317, 0, 1](°)$
	CR	dsigmf	$[67.5, 0.919, 0, 1](°)$ $[157.5, -0.919, 0, 1](°)$
	HO	sigmf	$[157.5, 0.919, 0, 1](°)$

TABLE I: Definition of the linguistic variable $DeltaCOG$.

Using $RB_i(\tau)$, the conditions stated in Equations (8) and (9c) can be represented as shown in Figure 3. Therefore, $RB_i(\tau)$ is also taken as an input to the $FIS_{colregs}$ using the linguistic variable RB. It consists of membership functions to represent *Starboard* (SB) and *Starboard Ahead* ($SBAH$). As illustrated in Figure 3, the input to RB range between $[-180°, 180°]$. The crisp ranges of SB ($0°-180°$) and $SBAH$ ($0° - 67.5°$) are extended by approximately $5°$. The Table II

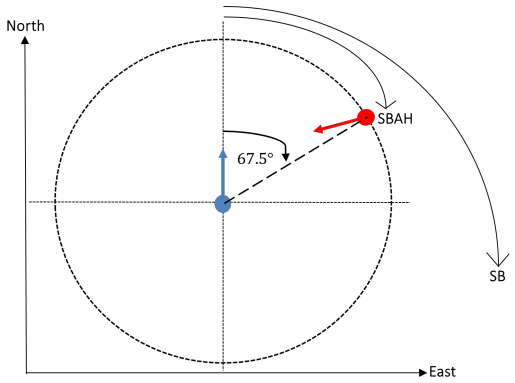


Fig. 3: The conditions identified in Equations (8) and (9c) using $RB_i(\tau)$. The blue dot and the arrow represent the own ship and its course direction while red represent a target vessel.

summarizes the details of the linguistic variable RB.

	Label	Type	Parameters P $[a_1, a_2, a_3, a_4]$ (unit)
RB	SB	sigmf	$[0, 0.919, 0, 1](^\circ)$
	$SBAH$	dsigmf	$[0, 0.919, 0, 1](^\circ)$ $[67.5, -0.919, 0, 1](^\circ)$

TABLE II: Definition of the linguistic variable RB

Finally, using $d_{o,i}(\tau)$ (the distance between own ship and obstacle i at time τ), the condition stated in Equation (6) can be evaluated by taking it as an input to the $FIS_{colregs}$ using the linguistic variable DCL. It contains one membership function *Close* (CL). The value for the parameter d_i^{cl} (introduced in Equation (6)) is taken as the crisp value and extended by a constant value chosen depending on the size of the own ship. A *sigmf* is used for the CL membership function, where it's parameters can be set according to the requirement.

The output of $FIS_{colregs}$ is represented by the linguistic variable mu which ranges between [0, 1]. The output contains two membership functions which have been summarized in Table III.

	Label	Type	Parameters P $[a_1, a_2, a_3, a_4]$
mu	mu_{true}	sigmf	$[0.95, 114.878, 0, 1]$
	mu_{false}	sigmf	$[0, 05. - 114.878, 0, 1]$

TABLE III: Definition of the linguistic variable mu

The graphical representation of the membership functions of inputs and outputs of $FIS_{colregs}$ are depicted in Figure 4.

Finally, the rules of the FIS needs to be defined. As the initial step, the different traffic situations in (5) i.e., CLOSE, STARBOARD, HEAD-ON, CROSSED and OVERTAKEN can be defined using the input membership functions of the FIS as shown in Table IV.

Traffic Situation	Linguistic Variable	Membership Function
CLOSE	DCL	CL
STARBOARD	RB	SB
HEAD-ON	DeltaCOG	HO
	RB	SBAH
CROSSED	DeltaCOG	CR
OVERTAKEN	DeltaCOG	OT

TABLE IV: Definition of traffic situations in (5) using the input membership functions of $FIS_{colregs}$

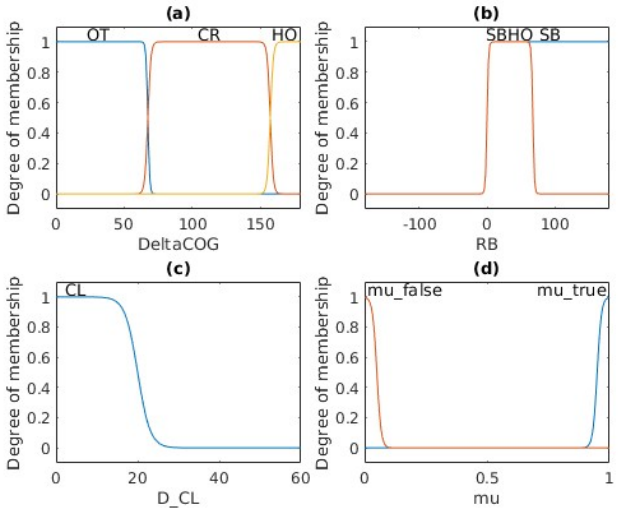


Fig. 4: Graphical representation of the membership functions of inputs and outputs of $FIS_{colregs}$. (a) $DeltaCOG$ (b) RB (c) DCL (d) mu.

In (5b), both STARBOARD and HEAD-ON traffic situations are included. Referring to Table IV, it is evident that the same linguistic variable RB appear for both traffic situations with different membership functions namely, SB and $SBAH$. However, when observing the graphical representation of RB depicted in Figure 3, it is clear that RB in SB is redundant when RB is in $SBAH$ since the input range of $SBAH$ overlap with that of SB. In simpler terms, when the target vessel is in the $SBAH$ range it is already on the Starboard side of the own ship. Therefore, the RULE14 defined in (5b) can be redefined with the input membership functions of the FIS as,

$$\text{R1: If } DeltaCOG \text{ is } HO \text{ and } RB \text{ is } SBAH \text{ and } DCL \text{ is } CL \text{ then } mu \text{ is } mu\_true \tag{19}$$

which can be considered the first rule of $FIS_{colregs}$.

Similarly, in (5c), both CROSSED and OVERTAKEN traffic situations are included. Referring to Table IV, it is evident that the same linguistic variable $DeltaCOG$ appear for both traffic situations with different membership functions namely, CR and OT. However, the condition in RULE15 is triggered when the target vessel is **not** overtaking the own ship. By refering to Figure 2, it becomes evident that due to the manner which the linguistic variable $DeltaCOG$ is defined, the condition NOT OVERTAKEN becomes redundant when $DeltaCOG$ is in CR. Thus, the RULE15 can be redefined with the input membership functions of the FIS as,

$$\text{R2: If } DeltaCOG \text{ is } CR \text{ and } RB \text{ is } SB \text{ and } DCL \text{ is } CL \text{ then } mu \text{ is } mu\_true \tag{20}$$

which can be considered the second rule of $FIS_{colregs}$. Additionally, due to the same reason, the speed condition for OVERTAKEN defined in (7b) can be disregarded as well.

Lastly, the third and the final rule of the FIS can be derived to drive the FIS output mu towards mu_{false} membership function range when the above two conditions are not met as,

$$\text{R3: If } RB \text{ is } not\ SB \text{ and } DCL \text{ is } not\ CL \text{ then } mu \text{ is } mu\_false \tag{21}$$

Therefore, the entire boolean logic explained in Equation (5) can be replaced by $FIS_{colregs}$. If the defuzzified crisp output of $FIS_{colregs}$ for obstacle i at time τ is represented by $\mu_i(\tau)$, the COLREGS cost component in (2) can be re-defined by multiplying $\mu_i(\tau)$ with the tuning parameter κ_i as,

$$\text{Modified COLREGS cost} \Rightarrow \kappa_i \cdot \mu_i(\tau) \tag{22}$$

There are additional advantages of using a FIS for COLREGS evaluation such as (a) accounting for uncertainties in measurements, and (b) ambiguity in COLREGS rules. The majority of the crisp values used when designing the linguistic variables are based on authors' experience and information from literature e.g. [27]. Therefore, they should not be taken as convention. These values can be modified depending on your system requirements and/or through surveys from experienced seafarers.

C. Maneuvering Cost

The maneuvering cost component defined in (12) is also modified by introducing the Sigmoid Function in (15). As mentioned in Section III-D, the parameters k_χ and Δ_χ needs to be asymmetric and penalise port side turns more compared to starboard turns. Therefore, addition of two Sigmoid Functions were used for each parameter as shown.

$$\text{Modified } k_\chi \Rightarrow \sigma^{port}_{k_\chi}(\chi_m) + \sigma^{starboard}_{k_\chi}(\chi_m) \tag{23}$$

where, the scaling parameters of $\sigma^{port}_{k_\chi}$ can be denoted as $a_1 = tunable, a_2 = tunable, a_3 = lowerlim_p, a_4 = upperlim_p$, while parameters of $\sigma^{starboard}_{k_\chi}$ as $a_1 = tunable, a_2 = tunable, a_3 = lowerlim_{sb}, a_4 = upperim_{sb}$. The 's' curve represented by $\sigma^{port}_{k_\chi}$ gives the penalty value when χ_m increases on the port side while $\sigma^{starboard}_{k_\chi}$ gives that on the starboard side. The parameters a_1 and a_2 of the two sigmoids can be use to adjust the slope of the curves. The parameters $lowerlim_p = lowerlim_{sb} = 0$ since the penalty should be zero when $\chi_m = 0$. The parameters satisfy $upperlim_p > upperlim_{sb}$, since port side turns should be penalized more than the starboard side turns. Similarly, for Δ_χ,

$$\begin{aligned}\text{Modified } \Delta_\chi \Rightarrow \sigma^{port}_{\Delta_\chi}(\chi_m - \chi_{m,last}) \\ + \sigma^{starboard}_{\Delta_\chi}(\chi_m - \chi_{m,last})\end{aligned} \tag{24}$$

where, the parameters a_1, a_2, a_3, a_4 of the two Sigmoid functions are chosen accordingly. Finally, the modified maneuvering cost can be denoted as,

$$\begin{aligned}\Gamma(U_m, \chi_m) = k_U(1 - U_m)^2 \\ + \text{Modified } k_\chi \cdot \chi_m^2 \\ + \Delta_U(U_m - U_{m,last})^2 \\ + \text{Modified } \Delta_\chi \cdot (\chi_m - \chi_{m,last})^2\end{aligned} \tag{25}$$

VI. Initialization Strategy

The initialization strategy utilizes the SBMPC algorithm stated in Section III with the modifications proposed for its cost components in Section V. The idea is to find a collision free state trajectory $\Psi(\tau_0) = \{\xi_{\tau_0}, \xi_{\tau_0+T_s^{init}}, \ldots, \xi_{\tau_0+T^{init}}\}$ where, T^{init} and T_s^{init} are prediction horizon and the discretization interval for the initialization stage. Following this definition, a single trajectory point at time $\tau \in D(\tau_0)$ can be denoted as $\xi_\tau = [x_\tau, y_\tau, \chi_\tau, U_\tau]^T$, which represent the state vector of the vessel. Algorithm 1 describes how the trajectory is calculated.

Algorithm 1: *calcInitTraj()*
Calculate the initialization trajectory

Data: current own ship state ξ_{τ_0}, current target ship state $\xi^{ts}_{\tau_0}$, $\chi_{m,last}$ and $U_{m,last}$ at time $\tau = \tau_0$.
Result: own ship trajectory $\Psi(\tau_0)$ predicted over the horizon T^{init} at discretization intervals T_s^{init}, starting from time $\tau = \tau_0$.

$\Psi(\tau_0) \leftarrow$ add current own ship state ξ_{τ_0};
$\chi^{init}_{m,last} = \chi_{m,last}$;
$U^{init}_{m,last} = U_{m,last}$;
$\xi_\tau = \xi_{\tau_0}$;
$\xi^{ts}_\tau = \xi^{ts}_{\tau_0}$;
$\tau = \tau_0$;
while $\tau \leq \tau_0 + T^{init}$ **do**
  $\chi^{init}_d, U^{init}_d \leftarrow$ using LOS guidance [28];
  $\chi^{init}_m, U^{init}_m \leftarrow$ $calcOptCtrls(\xi_\tau, \xi^{ts}_\tau, \chi^{init}_d, U^{init}_d, \chi^{init}_{m,last}, U^{init}_{m,last})$;
  $\chi^{init}_m = \chi^{init}_{m,last}$,$U^{init}_m = U^{init}_{m,last}$;
  $\chi^{init}_c = \chi^{init}_d + \chi^{init}_m$;
  $U^{init}_c = U^{init}_d \cdot U^{init}_m$;
  $\xi_\tau \leftarrow$ get the own ship state at $\tau + T^{init}_s$ using χ^{init}_c and U^{init}_c as inputs of Equation (13);
  $\xi^{ts}_\tau \leftarrow$ predict the target ship state at $\tau + T^{init}_s$ using Equation (14);
  $\Psi(\tau_0) \leftarrow$ add current own ship state ξ_τ;
  $\tau \leftarrow \tau + T^{init}_s$;
end

The function $calcOptCtrls()$ is used to calculate the optimal control inputs (χ^*_m and U^*_m) at the time step τ using the same exhaustive search method as the original SB-MPC. The Algorithm 2 further explains the function. The following finite sets of course (χ_m) and speed (U_m) modifications are considered in this process.

$$\begin{aligned}\chi^*_m \in \{-90, -75, -60, -45, -30, \\ -15, 0, 15, 30, 45, 60, 75, 90\}\end{aligned} \tag{26a}$$

$$U^*_m \in \{0.1, 0.5, 1\} \tag{26b}$$

In Algorithm 2, for every combination of χ^*_m and U^*_m stated in (26), the trajectories of the own ship and target ships ($\bar{\Psi}$ and $\bar{\Psi}^{ts}$) are derived. When calculating the trajectory of the own ship over the horizon T^{init}, it is worth noting that the corresponding course and speed modifications (χ^*_m, U^*_m) are assumed to remain constant. Thus, the inputs of the Equation (13) can be calculated as $\chi_{input,\tau} = \chi^{init}_d + \chi^*_m$ and $U_{input,\tau} = U^{init}_d \cdot U^*_m$. Next, cost from the SB-MPC algorithm and the length of the trajectory before the first collision point (with a static obstacle i.e., grounding hazard) in the trajectory is calculated (N_{TP}). Then, the N_{TP} value is converted to a cost component and added to $sbmpc\_cost$ to penalize χ^*_m, U^*_m combinations that result in shorter trajectories. This is achieved by subtracting the N_{TP} value from $N^{init}(= T^{init}/T_s^{init})$, which can be the maximum possible length of a trajectory at any given time, and multiplying the difference by a tuning parameter $k_{N_{TP}}$ to scale the value before adding it to the $total\_cost$. The purpose of this step is to penalize any control behaviors that would drive the vessel towards static obstacles.

To find N_{TP}, all points in the own ship trajectory ($\bar{\Psi}$) is looped-over while a counter variable is incremented (starting from zero) until a situation such as depicted in Figure 5 is reached, in which case the loop is terminated and the final value of the counter variable is taken as N_{TP}. In the Figure 5, the circular dashed-line marked around the ship is considered a hazardous region and any object that breaches the limits of this region is considered a collision with the vessel. In the literature, several studies propose different cost functions to address grounding hazards [8]. A shared characteristic among these methods lies in the parameterization, specifically the calculation of the distance to each static obstacle. However, the utilization of this common parameter poses implementation challenges. In contrast, our proposed method offers a more simplified strategy. Notably, our approach is characterized by its practicality and ease of parameter tuning. This advantage comes from the automatic consideration of ship domain and maneuvering capabilities when identifying collision situations, as illustrated in Figure 5.

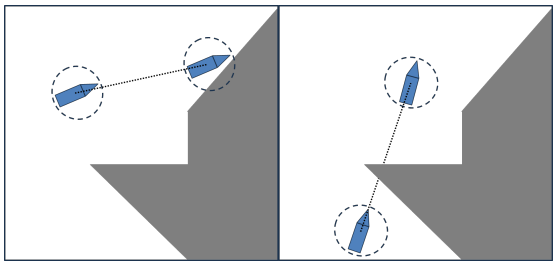


Fig. 5: The figure illustrates what is considered collision situations when calculating N_{TP} in Algorithm 2. The situation on the left is showcasing a collision instance where the limits of the hazardous region of the vessel is breached by the static obstacle. The situation on the right showcases another such instance where even though the limits of the hazardous region is not breached by the static obstacle, the path connecting the two trajectory points crosses a static obstacle region, thus considered another collision situation.

VII. MPC

This section will describe the implementation of the MPC controller for the vessel. Using (16), (22) and (25), the Discrete time Optimal Control Problem (DOCP) at time τ_0 can be defined as (27).

$$
\begin{aligned}
&\underset{\mathbf{x}_\tau, \mathbf{u}_{m,\tau}}{minimize}\ J_d = \\
&\sum_{i=1}^{N_{ts}} \sum_{\tau=\tau_0}^{\tau_0+T^{mpc}} (\sigma_{coll}(d_{o,i}(\tau)) \cdot C_i(\tau) \cdot R_i(\tau) \\
&+ \kappa_i \cdot \mu_i(\tau)) \\
&+ \sum_{\tau=\tau_0}^{\tau_0+T^{mpc}} \Gamma(\mathbf{u}_{m,\tau}) \qquad (27) \\
&s.t. \quad \mathbf{x}_{\tau+T_s^{mpc}} = \mathbf{F}_d(\mathbf{x}_\tau, \mathbf{u}_\tau), \\
&\quad \mathbf{x}(\tau_0) = \xi_{\tau_0},\ \mathbf{x}(\tau_0 + T^{mpc}) = \xi_{\tau_0+T^{mpc}}, \\
&\quad -90 \leq \chi_{m,\tau} \leq 90, \\
&\quad 0.1 \leq U_{m,\tau} \leq 1, \\
&\quad \mathbf{h}_{corr}(\mathbf{x}_\tau^{pos}) \leq 0, \qquad \forall \tau \in D(\tau_0)
\end{aligned}
$$

where $\mathbf{x}_\tau$, $\mathbf{u}_\tau$ and $\mathbf{F}_d$ are the states, control inputs and the discrete vessel dynamics of the own ship defined in (13). The control inputs $\mathbf{u}_\tau$ i.e., $\chi_{input,\tau}$ and $U_{input,\tau}$ are calculated by applying control modifications $\mathbf{u}_{m,\tau}$ i.e., $\chi_{m,\tau}$ and $U_{m,\tau}$ to the desired control commands derived using a LOS guidance controller. If the desired control commands are defined as $\chi_{d,\tau}$ and $U_{d,\tau}$, the control inputs $\chi_{input,\tau}$ and $U_{input,\tau}$ can be calculated as, $\chi_{input,\tau} = \chi_{d,\tau} + \chi_{m,\tau}$ and $U_{input,\tau} = U_{d,\tau} \cdot U_{m,\tau}$. Furthermore, the parameter N_{ts} denote the number of target vessels while the parameters T_s^{mpc} and T^{mpc} represent the sample time and the prediction horizon respectively. The sample times T_s^{init} and T_s^{mpc} should be equal while $T^{init} > T^{mpc}$. The initial and terminal conditions for the vessel states are applied using the state trajectory $\Psi(\tau_0)$ derived in Algorithm 1. The terminal condition is the vessel state at $\tau = \tau_0 + T^{MPC}$ i.e., $\xi_{\tau_0+T^{mpc}}$. This is set to ensure the vessel ends up at a collision free state found during the initialization and is updated at every time step. The value range of the decision variables $\chi_{m,\tau}$ and $U_{m,\tau}$ are set according to the maximum and minimum of the finite sets of course and speed modifications used for the initialization step as defined in (26a)-(26b). An additional constraint, $\mathbf{h}_{corr}(\mathbf{x}_\tau^{pos})$ is used to ensure that the vessel respect the cross-track error corridor boundaries explained in Section VII-A. The constraint is dependent on the variable $\mathbf{x}_\tau^{pos}$ that denotes the position coordinates $[x_\tau, y_\tau]$ of the vessel state at τ. The DOCP is solved repeatedly using the receding horizon approach while using $\Psi(\tau_0)$ to warm-start the solver. Furthermore, given the feasibility of the initialization stage, the DOCP is assured to discover an optimal solution that is at least as good as the sub-optimal initialization trajectory.

Algorithm 2: *calcOptCtrls()*
Calculate the optimum control inputs (χ_m^* and U_m^*) at the time step τ

Data: own ship state ξ_τ, target ship state ξ_τ^{ts}, χ_d^{init},U_d^{init},$\chi_{m,last}^{init}$ and $U_{m,last}^{init}$ at time step τ.
Result: The optimum control modifications χ_m^{init} and U_m^{init} for the vessel at time step τ.
foreach χ_m^* *in Equation (26a)* **do**
  foreach U_m^* *in Equation (26b)* **do**
    $\bar{\Psi} \leftarrow$ calculate the trajectory for own ship using χ_m^*, U_m^*, χ_d^{init}, U_d^{init} and Equation (13) over the horizon T^{init};
    $\bar{\Psi}^{ts} \leftarrow$ calculate the trajectory for own ship using Equation (14) over the horizon T^{init};
    $N_{TP} =$ calculate the number of trajectory points before the first collision point;
    $sbmpc\_cost =$ solve Equation (1) using $\bar{\Psi}, \bar{\Psi}^{ts}, \chi_{m,last}^{init}$ and $U_{m,last}^{init}$;
    $total\_cost = sbmpc\_cost + k_{N_{TP}}(N^{init} - N_{TP})$;
    $cost\_matrix \leftarrow [\chi_m^*, U_m^*, total\_cost]$;
  end
end
χ_m^{init},$U_m^{init} \leftarrow \chi_m^*$, U_m^* that gives the lowest $total\_cost$ value in the $cost\_matrix$;

A. Cross-track Error Corridor

The local spacial constraints $\mathbf{h}_{corr}(\mathbf{x}_\tau^{pos})$, are also calculated using the own ship trajectory derived from Algorithm 1. The constraints are generated using the trajectory generated during the initialization stage i.e., $\Psi(\tau_0)$, as the reference

trajectory to generate a cross-track error corridor; introduced in authors' previous work [1]. The idea is to define a corridor region as the navigable area for the vessel trajectory over a time horizon. As explained in Algorithm 2, the trajectory $\Psi(\tau_0)$ is free of collisions with static obstacles. Therefore, by using $\Psi(\tau_0)$ to generate the so called cross-track error corridor, we essentially ensure that the constraints $\mathbf{h}_{corr}(\mathbf{x}_\tau^{pos})$ eliminate grounding hazards. Hence, it is favourable to have a larger T^{init} as it would help take early action against collision hazards with a relatively lower computational effort. The algorithm to generate the cross-track error corridor is explained in Algorithm 3,

Algorithm 3: Calculating cross-track error corridor

Data: Static obstacle polygon data G, the own ship trajectory $\Psi(\tau_0)$ calculated in Algorithm 1, and maximum corridor width Δd.
Result: Two arrays of points, $g^{\Delta d+}$ and $g^{\Delta d-}$ that mark the error corridor on either side of the vessel

$N^{mpc} = (T^{mpc}/T_s^{mpc})$;
while $i < N^{mpc} + 1$ **do**
  $P_i \leftarrow i$th vessel coordinate point in $\Psi(\tau_0)$;
  $P_{i+1} \leftarrow (i+1)$th vessel coordinate point in $\Psi(\tau_0)$;
  $L_{perp} \leftarrow$ perpendicular line to the line connecting P_i and P_{i+1};
  $P_{\Delta d+} \leftarrow$ point on L_{perp} at a $+\Delta d$ distance along the L_{perp} from the point P_i;
  $P_{\Delta d-} \leftarrow$ point on L_{perp} at a $-\Delta d$ distance along the L_{perp} from the point P_i;
  $L_{\Delta d+} \leftarrow$ line connecting $P_{\Delta d+}$ and the point P_i;
  $L_{\Delta d-} \leftarrow$ line connecting $P_{\Delta d-}$ and the point P_i;
  if $L_{\Delta d+}$ *intersect with* G **then**
    $\mathbf{g}^{\Delta d+} \leftarrow$ coordinate of the intersection point
  else
    $\mathbf{g}^{\Delta d+} \leftarrow P_{\Delta d+}$
  end
  if $L_{\Delta d-}$ *intersect with* G **then**
    $\mathbf{g}^{\Delta d-} \leftarrow$ coordinate of the intersection point
  else
    $\mathbf{g}^{\Delta d-} \leftarrow P_{\Delta d-}$
  end
  $i \leftarrow i + 1$;
end

In Algorithm 3, the input G is obtained by defining the static obstacles as two dimensional polygons. Since the own ship is considered as a point object for the calculations in Algorithm 3, we offset that by adding a buffer of length equivalent to the radial distance of a circular region around the vessel (a breach in the limits of this circular region by a static obstacle is considered a collision). The input, maximum clearance on each side (Δd) is half of the total width of the corridor when static obstacles aren't present on either side of the vessel, and can be defined based on the vessel dimensions. When obstacles are present on either side of the vessel, the total clearance on each side is less than Δd. Finally, the outputs of the algorithm (the two point arrays $\mathbf{g}^{\Delta d+}$ and $\mathbf{g}^{\Delta d-}$), can be used to derive the $\mathbf{h}_{corr}(\mathbf{x}_\tau^{pos})$ constraints as,

$$\mathbf{h}_{corr}(\mathbf{x}_\tau^{pos}) = \begin{bmatrix} \mathbf{x}_\tau^{pos} - max(\mathbf{g}_j^{\Delta d+}, \mathbf{g}_j^{\Delta d-}) \\ -(\mathbf{x}_\tau^{pos} - min(\mathbf{g}_j^{\Delta d+}, \mathbf{g}_j^{\Delta d-})) \end{bmatrix} \tag{28}$$

where, $j = 1, 2, \ldots, N^{mpc} (= T^{mpc}/T_s^{mpc})$ for $\forall \tau \in D(\tau_0)$.

B. Fuzzy Logic Implementation

Building a smooth objective function in (27) for an NLP solver is not straightforward due to the Fuzzy Logic component $\mu_i(\tau)$. In particular, there are two main stages in fuzzy logic implementation that could result in non-smooth function components, namely, inference stage and the defuzzification stage. Let us first analyse the fuzzy inference stage. It is the process of interpreting the fuzzified inputs as outputs according to the user defined fuzzy inference rules, for example, (19)-(21) of $FIS_{colregs}$. In a typical implementation, the 'and' and 'or' statements in the rules are interpreted using standard 'min' and 'max' functions respectively. These can be defined as,

$$max(z_1, z_2) = \begin{cases} z_1, & \text{if } z_1 \geq z_2 \\ z_2, & \text{otherwise} \end{cases} \tag{29a}$$

$$min(z_1, z_2) = -max(-z_1, -z_2) \tag{29b}$$

where z_1 and z_2 are inputs.

However, equations in (29) are non-smooth functions. Therefore, it is important to use a smooth variant of the 'max' function when building the objective function. Different 'max' function approximations can be found in literature such as Mellowmax, p-Norm and Smooth Maximum Unit to name a few. From these we have chosen the Smooth Maximum Unit which is defined as,

$$\begin{aligned} max_\varepsilon(z_1, z_2) &= \frac{z_1 + z_2 + |z_1 - z_2|_\varepsilon}{2} \\ &= \frac{z_1 + z_2 + \sqrt{(z_1 - z_2)^2 + \varepsilon}}{2} \end{aligned} \tag{30}$$

where, $\varepsilon \geq 0$ is a small pertubation. As $\varepsilon \to 0$, $|\cdot|_\varepsilon \to |\cdot|$ and thus $max_\varepsilon \to max$. Therefore, following the definition in (29b), the smooth approximation of 'min' function can be defined as, $min_\varepsilon(z_1, z_2) = -max_\varepsilon(-z_1, -z_2)$.

After the inference stage, the resulting fuzzy set is converted into a crisp value during the defuzzification stage. The Center of Gravity (CoG) method is one of the most popular defuzzification methods [29] and is used in this research as it gave the best results. For comprehensibility, consider a Single Input Single Output (SISO) FIS as shown in Figure 6. The input has been limited to two Triangular membership functions namely, $\underline{A}$ and $\underline{B}$ for simplicity, and the output has been assigned two Sigmoidal membership functions $\underline{C}$ and $\underline{D}$ similar to the output mu in $FIS_{colregs}$. The rules are stated as follows,

- R1: If Z is $\underline{A}$ then Y is $\underline{C}$
- R2: If Z is $\underline{B}$ then Y is $\underline{D}$

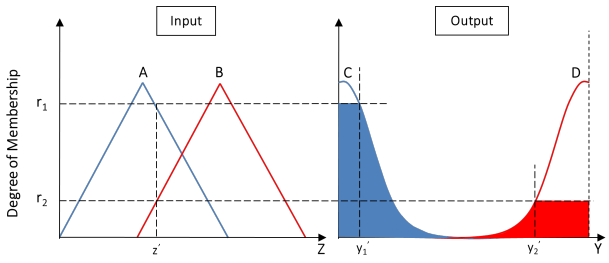


Fig. 6: A Single Input Single Ouput FIS example.

When following the CoG defuzzification method, calculation of y_1' and y_2' is required which is difficult and could lead to discontinuities in the objective function (27). Hence,

we propose scaling the output membership functions by multiplication of each membership function with the respective inference values r_1 and r_2 (in Figure 6) as $r_1 \cdot \sigma_C$ and $r_2 \cdot \sigma_D$ where, σ_C and σ_D represent membership functions $\underline{C}$ and $\underline{D}$. Next, these scaled membership functions can be directly used for CoG calculation using an integral approximation method such as rectangle approximation, eliminating the need of calculating $y_1^{'}$ and $y_2^{'}$. The scaling process of the membership function is further explained in Figure 7.

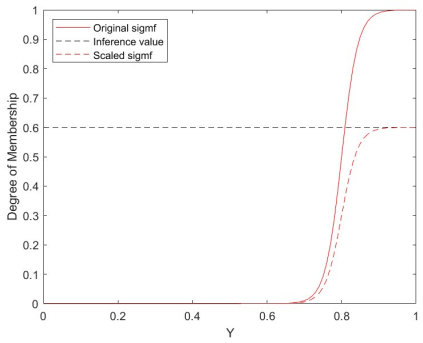


Fig. 7: The figure illustrates an example where the output membership function is scaled for defuzzification. The Original $sigmf$ is the actual membership function $\underline{D}$ of output Y. The inference value is 0.6 from the fuzzy rules r_1 in Figure 6. The Scaled $sigmf$ is the result of $r_2 \cdot \sigma_D$.

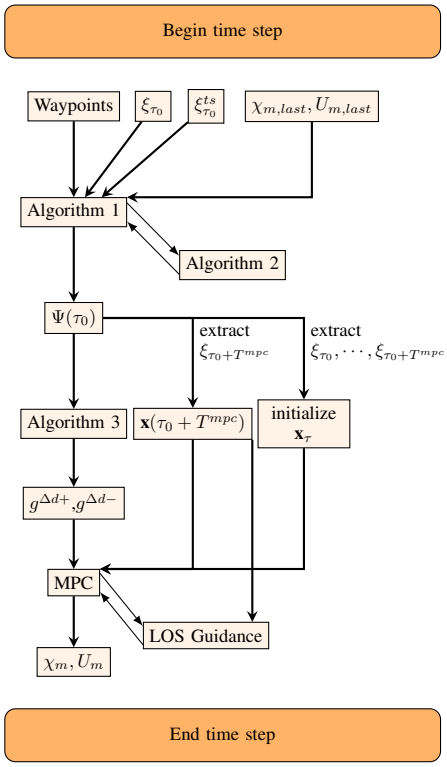


Fig. 8: The collision avoidance control system flow chart

VIII. CONTROL SYSTEM ARCHITECTURE

This section intends to summarize the the entire collision avoidance algorithm explained over the Sections VI and VII. The Figure 8 depicts the flowchart of the process where the optimum course and and speed modifications are determined at each time step.

Initially, the Algorithm 1 is run to find the a sub-optimal trajectory $\Psi(\tau_0)$, that is free of collisions with static obstacles. As explained in the Section VII, the prediction horizon of the initialization stage (T^{init}) is larger than that of the MPC stage (T^{MPC}), allowing the algorithm to look far ahead with comparatively lower computational effort. Once $\Psi(\tau_0)$ is obtained, it is used to formulate the DOCP in (27). Firstly, $\Psi(\tau_0)$ is used directly by Algorithm 3, to generate the cross-track error corridor which serves as constraints of the DOCP. Secondly, the trajectory position of own ship at $\tau_0 + T^{MPC}$ i.e., $\xi_{\tau_0+T^{mpc}}$ is used as the terminal point and the trajectory points from τ_0 to $\tau_0 + T^{MPC}$ i.e., $\xi_{\tau_0}, \cdots, \xi_{\tau_0+T^{mpc}}$ are used to initialize the optimization variable $\mathbf{x}_\tau$ to warm-start the MPC. The $\xi_{\tau_0+T^{mpc}}$ value is also used by the LOS Guidance controller to calculate the desired course and speed commands (χ_d and U_d). Finally, the MPC solves the DOCP to obtain the optimal control input modifications χ_m and U_m. After modifying χ_d and U_d values by χ_m and U_m, the resulting course and speed commands should be fed to an autopilot to control the vessel.

IX. SIMULATION RESULTS

The Smooth-SBMPC algorithm is evaluated in a simulation study. The algorithm is implemented on Matlab version 9.13.0 (R2022b) Update 3 running on a work station with Intel (R) Core(TM) i7-10750H CPU with 32 GB RAM and an NVIDIA RTX 2070 SUPER Mobile GPU. When generating simulation environments, two-dimensional polygons are used to represent riparian land and other static obstacles. Matlab built-in API are used for polygon manipulation and information extraction.

The study is designed as a comparative analysis. In Section I, it was pointed out that deriving optimal control inputs from a finite solution pool hinder the chances of adapting SB-MPC for inland waterway navigation and collision avoidance. Therefore, the Smooth-SBMPC algorithm is compared with the original SB-MPC [7], and the objective of the comparison is to investigate the benefit of higher accuracy in the control space. As mentioned in the same section, one of the solutions found in the literature for this problem is using a hybrid control system architecture. Hence, we also compare Smooth-SBMPC with the two-level hybrid control algorithm from our prior research [1] to assess the significance of maintaining consistent optimization constraints across all layers. During the study, first, a simulation scenario i.e., static obstacles, own ship way points, target ship trajectories etc. is defined and three separate simulations were conducted changing the collision avoidance algorithm of the own ship. Next, the obtained data were analyzed and compared. We have repeated this process for several simulation scenarios and have obtained similar results. Thus, we have decided to present only one of those simulation scenarios in this paper. Simulation 1-3 presented in the subsequent sections all contain the same simulation scenario while the own ship is equipped with a different collision avoidance algorithm in each simulation.

Four snapshots of each said simulations are depicted in the Figures 9, 11 and 13. Here, gray color areas mark static obstacle polygons while white regions denote the water body. The blue color is used to denote the own ship. The target vessels are denoted using red, magenta, cyan and green colors. The target vessels are considered as point objects while the own ship is given a circular ship domain marked by the

solid blue circle around the ship. The remaining two blue colored dashed circles around the own ship represent d^{safe} (see (3) for the definition) and d^{cl} (see (6) for the definition) parameters respectively. In addition, in Figures 10a, 12a and 14a, the Euclidean distance between each target vessel and the own ship is plotted. The colors of the lines in these figures correspond to the respective target vessel color in Figures 9,11 and 13. The course angle and speed commands of the own ship in each simulation is plotted in Figures 10b, 12b and 14b.

A. Simulation 01

In this section we will be studying the original SB-MPC algorithm [6]. The tuning parameter values of the SB-MPC algorithm are set to be consistent with the corresponding parameters in the Smooth-SBMPC algorithm. This consistency ensures a standardized basis for comparison between the two algorithms, facilitating a meaningful evaluation of their respective performances. The results of the simulation are shown in Figures 9 and 10.

B. Simulation 02

In this section we will be observing the the two-level hybrid collision avoidance algorithm presented in [1]. The algorithm consists of two layers where, the top layer is equiped with an MPC controller which acts as a guidance controller while respecting the assigned constraints for riparian land and other static obstacles. On the bottom layer the original SB-MPC is run with an additional cost component accounting for the grounding cost. The bottom layer SB-MPC is solely responsible for avoiding collisions with target vessels. Similar to Simulation 01, the tuning parameter values of the bottom layer SB-MPC algorithm are also set to be consistent with the corresponding parameters in the Smooth-SBMPC algorithm.

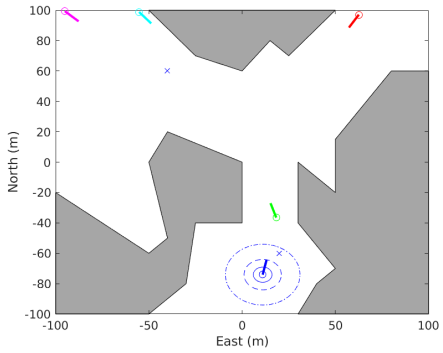
The results of the simulation are shown in Figures 11 and 12. The land section that intersect with the path connecting the two adjacent waypoints is considered a static obstacle. Thus, a circular region is generated (beige colored circular region marked in the Figure 11) as an approximation to derive the constraints.

C. Simulation 03

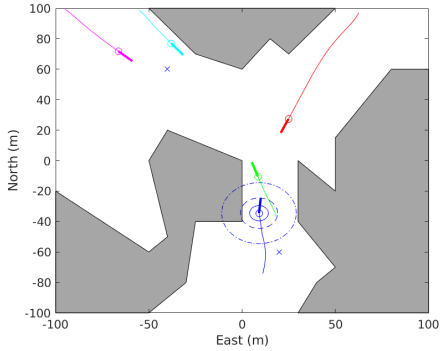
In this section we will be observing the performance of the Smooth-SBMPC algorithm presented in this paper. The results of the simulation are shown in Figures 13 and 14.

D. Comparisons

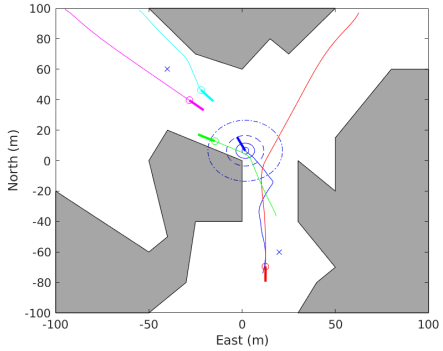
1) Evaluation Framework: Several papers can be found in literature with methods to validate different collision avoidance algorithms. Among them, based on the initial work by [30], the authors of [27] have developed an evaluation framework with regards to safety and compliance with the COLREGS Rules 8(a, b), 13, 14, 15, 16 and 17, which from here on referred to as **EvalTool**. The assessment is conducted on an encounter basis, where an encounter is defined as the period when two vessels assume a specific configuration of position, course, and speed that triggers the application of one of the COLREGs rules. For each encounter, the evaluation generates a single score ($S_{sit} \in [0, 1]$) per vessel, reflecting the vessel's adherence to safety measures and COLREGs compliance concerning the relevant rules. This score is computed using metrics that characterize the vessels' trajectory throughout the encounter. Scores or



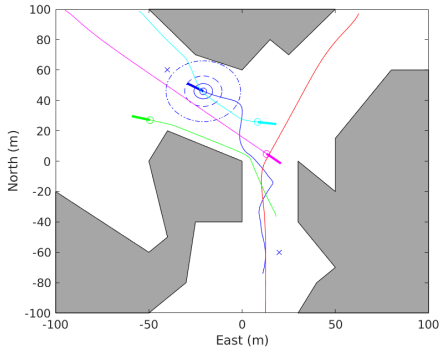
(a) $t = 4s$



(b) $t = 24s$

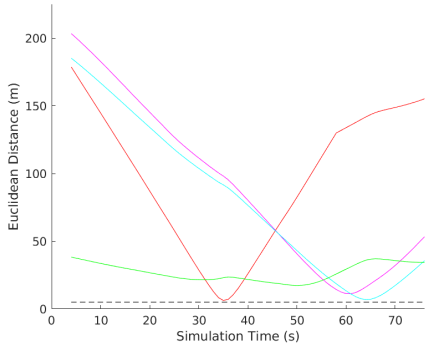


(c) $t = 49s$

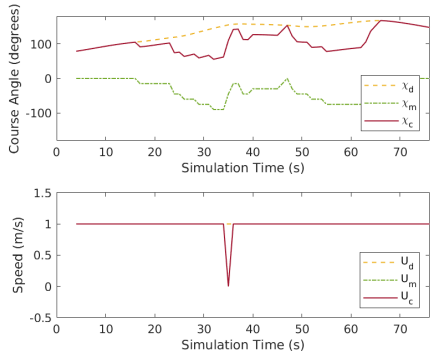


(d) $t = 76s$

Fig. 9: Trajectories of each vessel during Simulation 01



(a) Euclidean distance between each target ship and own ship



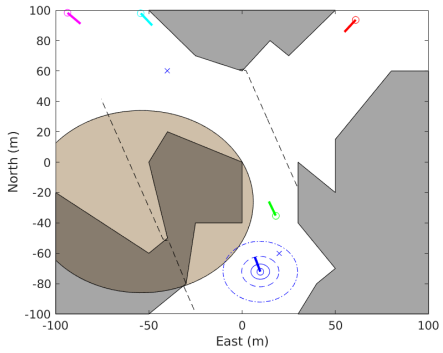
(b) Course and speed commands of the own ship

Fig. 10: Additional information of Simulation 01. In (a), each colored-line corresponds to the respective colored vessel in Figure 9.

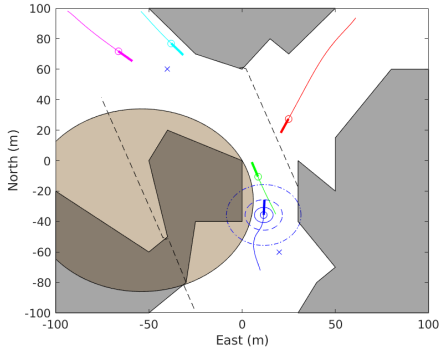
penalties are assigned to the behavioral traits represented by these metrics based on whether the behavior aligns with or deviates from the prescribed rules. These individual scores and penalties are then aggregated to establish the overall encounter score for the vessel. An overview of the scores and penalties employed in the EvalTool can be found in APPENDIX II. In addition, the readers are referred to the original paper [27] for detailed information. While the suggested framework exhibits considerable potential, it is crucial to recognize that it is not exhaustive in addressing all aspects. This is evident in its limitations, such as not considering the overall quality of the trajectory and being restricted to accounting for single-ship encounters.

Hence, we first use the EvalTool to obtain a primary validation for the three algorithms in terms of safety and COLREGs compliance. Each target ship encounter is compared over the three simulations for comparison. In the study by Hagen et al. [27], a default set of tuning parameter values for the EvalTool is introduced. It's important to note that these values are initially proposed for larger vessels executing collision avoidance maneuvers in open waters. However, our research focuses exclusively on smaller inland vessels operating in confined environments. Consequently, apart from the parameter values explicitly outlined in Table V, all other values have been adopted without alteration for the purposes of this simulation study. Furthermore, these parameter values have been kept equal for all simulations.

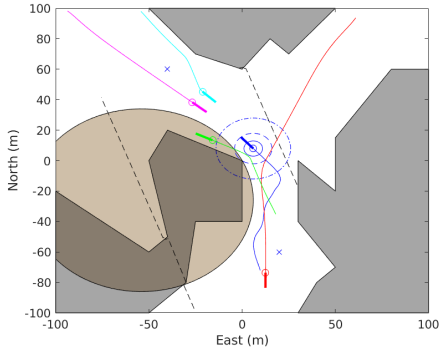
The evaluation results from EvalTool for each target vessel are presented in Tables VI, VII, VIII and IX. For every target



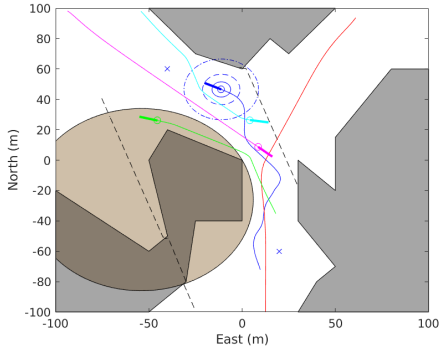
(a) $t = 5s$



(b) $t = 24s$

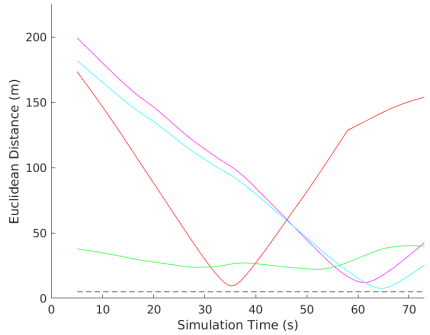


(c) $t = 50s$

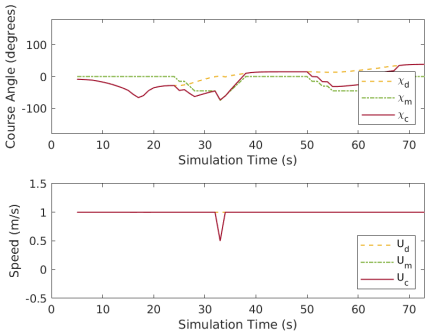


(d) $t = 73s$

Fig. 11: Trajectories of each vessel during Simulation 02



(a) Euclidean distance between each target ship and own ship



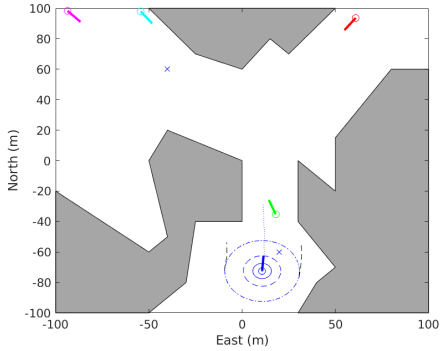
(b) Course and speed commands of the own ship

Fig. 12: Additional information of Simulation 02. In (a), each colored-line corresponds to the respective colored vessel in Figure 11.

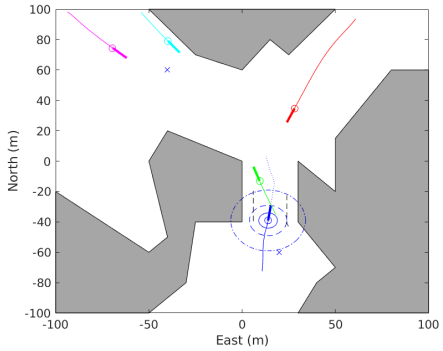
Parameter	Value	Unit
r_{Stage2}	80	m
r_{Stage3}	45	m
r_{Stage4}	16	m
r_{pref}	7	m
r_{min}	6	m
r_{nm}	5	m
r_{col}	4	m
ϵ_U	0.2	m/s
$\Delta_{\chi_{ap}}$	13	deg
$\Delta_{\chi_{md}}$	1	deg
ΔU_{ap}^{rel}	0.2	m/s
ΔU_{md}^{rel}	0.1	m/s

TABLE V: EvalTool adjustable parameter values changed from that of the default values presented in [27].

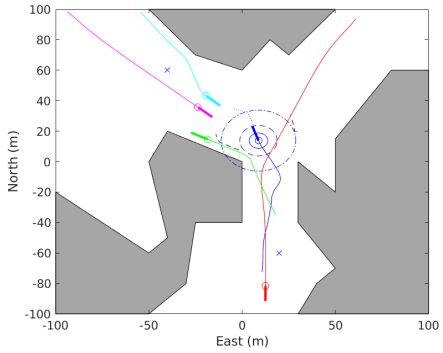
vessel, Simulation 3 has scored the highest score closely followed by Simulation 2 while Simulation 1 having the lowest scores. From the four target vessels, the own ship has fared poorly against the Magenta colored vessel in all three simulations. This observation can be illustrated using Figures 9c, 11c, and 13c, which depict corresponding instances in each simulation involving the Magenta-colored target vessel. Subsequently, the own ship adjusts its maneuvering not only to avoid collision with the Magenta-colored vessel but also with the Cyan-colored vessel. However, the EvalTool is designed only for evaluating single-ship encounters, thus, unable to compensate for maneuvers related to other target vessels that needs a simultaneous reaction. The EvalTool recognizes the encounter with the Magenta-colored vessel as a 'crossing stand-on' (based on COLREGs rule 15 and 17) situation. In such scenarios, the own ship is expected to maintain its



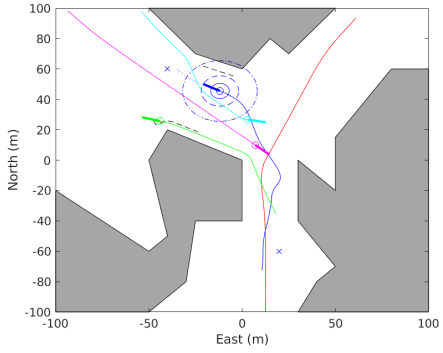
(a) $t = 5s$
johansen˙ship˙2016



(b) $t = 22s$

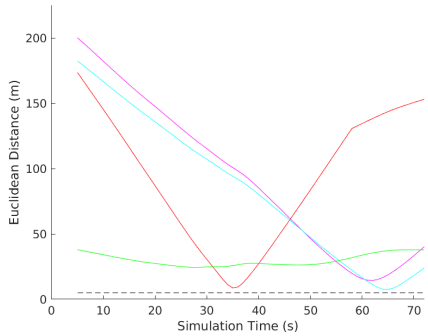


(c) $t = 52s$

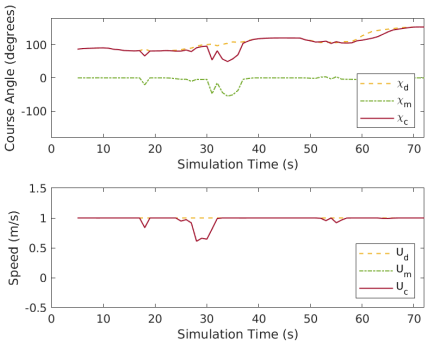


(d) $t = 72s$

Fig. 13: Trajectories of each vessel during Simulation 03



(a) Euclidean distance between each target ship and own ship



(b) Course and speed commands of the own ship

Fig. 14: Additional information of Simulation 03. In (a), each colored-line corresponds to the respective colored vessel in Figure 13.

course and speed until it becomes evident that the give-way vessel (the Magenta-colored vessel, in this case) fails to take appropriate action. Consequently, the own-ship's reaction to avoid collision with the Cyan-colored vessel is penalized when evaluating the Magenta-colored vessel. Hence the overall lower score from all three algorithms.

	Simulation 01	Simulation 02	Simulation 03
Collision situation	Crossing give-way	Head-on	Crossing give-way
Scores and Penalties	-S_{15}: 0.34564 –P_{ah15}: 0 –S_{16}: 0.34564 —S_{safety}: 0.96321 —-S_{Θ}: 0.75188 —-S_r: 0.85174 —P_{delay}: 0.64116 —$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_U}$: 0	-S_{14}: 0.955 –P_{nsb}: 0 –P_{sts}: 0.075006 –$P^{\neg ap}_{\Delta_\chi}$: 0 –$P_{delay}$: 0	-S_{15}: 1 –P_{ah15}: 0 –S_{16}: 1 —S_{safety}: 1 —-S_{Θ}: 0.89363 —-S_r: 1 —P_{delay}: 0 —$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_U}$: 0

TABLE VI: Results of the EvalTool for the Red colored target vessel in Simulation 1-3.

Another essential factor in assessing the safety of a collision avoidance algorithm is the grounding clearance, which refers to the distance between the vessel and the shore throughout its trajectory. While all three algorithms examined in the study managed to prevent grounding hazards during navigation,

	Simulation 01	Simulation 02	Simulation 03
Collision situation	Crossing stand-on	Crossing stand-on	Crossing stand-on
Scores and Penalties	–S_{15}: 0.09 –S_{safety}: 1 —S_{Θ}: 0.91321 —S_r: 1 –S_{17}: 0.09 —P_{pt}: 0 —P_2: 0.7 —-$P_{2,\Delta\chi}$: 1 —-$P_{2,\Delta U\uparrow}$: 0 —-$P_{2,\Delta U\downarrow}$: 0 —-$C_{2,gw}$: 0.2 —P_3: 0.7 —-$P_{3,\Delta\chi}$: 1 —-$P_{3,\Delta U\uparrow}$: 0 —-$P_{3,\Delta U\downarrow}$: 0 —-$C_{3,gw}$: 0.2	–S_{15}: 0.24 –S_{safety}: 1 —S_{Θ}: 0.88079 —S_r: 1 –S_{17}: 0.24 —P_{pt}: 0 —P_2: 0.2 —-$P_{2,\Delta\chi}$: 0 —-$P_{2,\Delta U\uparrow}$: 0 —-$P_{2,\Delta U\downarrow}$: 0 —-$C_{2,gw}$: 0.2 —P_3: 0.7 —-$P_{3,\Delta\chi}$: 1 —-$P_{3,\Delta U\uparrow}$: 0 —-$P_{3,\Delta U\downarrow}$: 0 —-$C_{3,gw}$: 0.2	-S_{15}: 0.30291 –S_{safety}: 1 —S_{Θ}: 0.9627 —S_r: 1 –S_{17}: 0.30291 —P_{pt}: 0 —P_2: 0.53154 —-$P_{2,\Delta\chi}$: 0.66308 —-$P_{2,\Delta U\uparrow}$: 0 —-$P_{2,\Delta U\downarrow}$: 0 —-$C_{2,gw}$: 0.2 —P_3: 0.35339 —-$P_{3,\Delta\chi}$: 0.30679 —-$P_{3,\Delta U\uparrow}$: 0 —-$P_{3,\Delta U\downarrow}$: 0 —-$C_{3,gw}$: 0.2

TABLE VII: Results of the EvalTool for the Magenta coloured target vessel in Simulation 1-3.

	Simulation 01	Simulation 02	Simulation 03
Collision situation	Head-on	Crossing give-way	Crossing give-way
Scores and Penalties	-S_{14}: 0.70719 –P_{nsb}: 0 –P_{sts}: 0.086949 –$P^{\neg ap}_{\Delta_\chi}$: 0 –$P_{delay}$: 0.25389	-S_{15}: 1 –P_{ah15}: 0 –S_{16}: 1 —S_{safety}: 1 —-S_{Θ}: 0.95141 —-S_r: 1 —P_{delay}: 0 —$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_U}$: 0	-S_{15}: 1 –P_{ah15}: 0 –S_{16}: 1 —S_{safety}: 1 —-S_{Θ}: 1 —-S_r: 1 —P_{delay}: 0 —$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_U}$: 0

TABLE VIII: Results of the EvalTool for the Cyan coloured target vessel in Simulation 1-3.

it is imperative to evaluate the effectiveness of this hazard avoidance. Such evaluation serves as evidence of the benefits derived from increased resolution in the control space. Hence, the shortest distance from the vessel to the shore was calculated for each simulation, resulting in values of 5.86 meters for simulation 1, 9.45 meters for simulation 2, and 9.46 meters for simulation 3. As expected, Smooth-SBMPC algorithm has the highest ground clearance out of the three algorithms.

2) Trajectories: While it is essential to evaluate safety and compliance with COLREGs rules for a comprehensive assessment of the three algorithms, the advantages of increased resolution in the control space become more apparent in the overall quality of trajectories. A side-by-side comparison of the trajectories followed by the own ship across the three simulations proves valuable in assessing the overall trajectory quality produced by each algorithm. Figure 15 clearly illustrates that Smooth-SBMPC yields a significantly smoother path compared to the other two algorithms.

Furthermore, the trajectory in Simulation 3 can be considered the expected behaviour if a manned-vessel encounter a similar collision scenario. This is evident in its ability to navigate between the two way-points without substantial alterations to course and speed. In contrast to the EvalTool, which

	Simulation 01	Simulation 02	Simulation 03
Collision situation	Overtaking give-way	Overtaking give-way	Overtaking give-way
Scores and Penalties	-S_{13}: 1 –P_{ah13}: 0 –S_{16}: 1 —S_{safety}: 1 —-S_{Θ}: 0.55148 —S_r: 1 —P_{delay}: 0 —$P^{\neg ap}$: 0 —-$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_U}$: 0	-S_{13}: 1 –P_{ah13}: 0 –S_{16}: 1 —S_{safety}: 1 —-S_{Θ}: 0.54698 —S_r: 1 —P_{delay}: 0 —$P^{\neg ap}$: 0 —-$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_U}$: 0	-S_{13}: 1 –P_{ah13}: 0 –S_{16}: 1 —S_{safety}: 1 —-S_{Θ}: 0.31925 —S_r: 1 —P_{delay}: 0 —$P^{\neg ap}$: 0 —-$P^{\neg ap}_{\Delta_\chi}$: 0 —-$P^{\neg ap}_{\Delta_U}$: 0

TABLE IX: Results of the EvalTool for the Green coloured target vessel in Simulation 1-3.

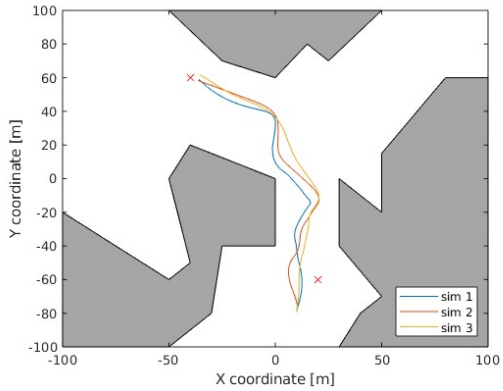


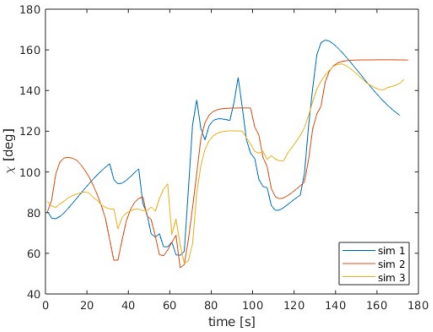
Fig. 15: Trajectories of the own ship from simulation 1-3 are plotted in the same graph.

is commonly employed for safety and COLREGS compliance evaluation, there is a limited body of research dedicated to numerically assessing the performance of collision avoidance algorithms in terms of quality and their resemblance to manned vessels in similar collision scenarios. However, certain papers primarily focused on safety and collision risk assessment propose straightforward equations for evaluating trajectory fitness [31], [32]. While these equations have limitations when used independently for our comparison, they share a common parameterization approach for the task. Namely, path length and voyage time. Hence, the path length and voyage time for each simulation have been extracted and are presented in Table X. As anticipated, the Smooth-SBMPC algorithm exhibits the shortest path length, indicative of the smoothness of its trajectory.

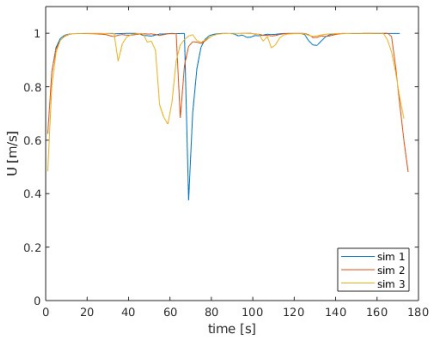
Simulation	Distance (m)	Time (s)
1	166.3	172
2	169.2	176
3	165.6	174

TABLE X: Total distance travelled by the own ship in each simulation and the respective time spent.

3) Course and speed changes: Another useful data when contemplating the advantage of increased control space is the course angle and speed changes throughout the trajectory. Figure 16a depict the course angle variations of the own ship over the simulation period for all three simulations, while



(a) Course Angle (χ)



(b) Speed (U)

Fig. 16: Course Angle (χ) and Speed (U) of the own ship for over the duration of each simulation for Simulation 1,2 and 3.

Figure 16b demonstrate the corresponding variations in speed.

The Figure 16 illustrates that in Simulation 3, successive alterations in course and speed tend to be more modest compared to the other two simulations. Quantitatively, the course values exhibit ranges: $59° - 165°$, $53° - 155°$, and $55° - 153°$ while speed values exhibit ranges: $0.37 - 0.99ms^{-1}$, $0.48 - 1ms^{-1}$ and $0.48 - 0.99ms^{-1}$ for simulations 1-3, respectively. Moreover, the average rate of absolute course change, representing the average magnitude by which the course angle of a vessel changes over a specific time frame, was determined to be 2.43, 2.04, and 1.53 degrees per second for simulations 1, 2, and 3, respectively. These observations serve as a testament to the smoothness achieved by the Smooth-SBMPC algorithm.

X. CONCLUSIONS

A novel collision avoidance control algorithm for ships, Smooth-SBMPC (Smooth Scenario-Based Model Predictive Control), has been developed to address the resolution limitation of solutions derived from the discretized set of solutions in the original SBMPC algorithm [6]. Smooth-SBMPC modifies and replaces non-smooth cost components in the fitness function used in the SBMPC, enabling the derivation of optimal solutions through derivative-based optimization.

The algorithm employs a two-stage optimization process. First, the original SBMPC generates a sub-optimal solution trajectory, which is then used to warm-start the MPC controller in the subsequent optimization stage. Additionally, this trajectory is utilized to derive local spatial constraints, effectively avoiding grounding hazards. The algorithm's efficacy has been verified through a comparative simulation study, where it

was compared with two other algorithms, including the original SBMPC. Results from the simulation study suggest that Smooth-SBMPC addresses the research objectives outlined in Section II. Future work may delve into improving the efficiency of the algorithm.

APPENDIX I

Below you can find the main rules of COLREGS [2] associated with collision avoidance, cited from [6],

- Rule 6—Safe speed: The following should be considered: Visibility, traffic density, stopping distance and turning.
- Rule 8—Actions to avoid collision. Actions shall be made in ample time. If there is sufficient sea-room, alteration of course alone may be most effective. Safe distance required. Reduce speed, stop or reverse if necessary. Action by the ship is required if there is risk of collision, also when the ship has right-of-way.
- Rule 13—Overtaking. Any vessel overtaking any other shall keep out of the way of the vessel being overtaken. A vessel shall be deemed to be overtaking when coming up with another vessel from a direction more than 22.5 degrees abaft her beam.
- Rule 14—Head-on situation. When two power-driven vessels are meeting on nearly reciprocal courses so as to involve risk for collision, then alter course to starboard so that each pass on the port side of each other.
- Rule 15—Crossing situation. When two power-driven vessels are crossing so as to involve risk of collision, the vessel which has the other on her own starboard side shall keep out of the way.
- Rule 16—Actions by give-way vessel. Take early and substantial action to keep well clear.
- Rule 17—Actions by stand-on vessel. Keep course and speed (be predictable) if possible. If it is necessary to take action, then the ship should try to avoid altering course to port for a vessel on her own port side.
- Rule 18—Responsibilities between vessels. Except for Rules 9, 10, and 13, a power-driven vessel shall keep out of the way of: a vessel not under command, a vessel restricted in her ability to maneuver, a vessel engaged in fishing, and a sailing vessel.
- Rule 19—Conduct of vessels in restricted visibility. Avoid alteration of course to port for a vessel forward of the beam, and avoid alteration of course towards a vessel abeam or abaft the beam, if possible.

APPENDIX II

In Table XI you can find the scores and penalties employed by the EvalTool cited from [27]. In the table, the term 'CPA' stands for Closest Point of Approach, i.e., the point when the range between vessels is at its smallest. Moreover, the term 'stage' is employed to differentiate specific ranges of distance values between the target ship and the own ship. The definition of each stage is as follows:

- Stage 1: Includes vessels that have been identified but are situated at a distance where no actions need to be taken.
- Stage 2: The nature of the encounter needs to be determined, and considerations for evasive maneuvers must be made.
- Stage 3: At this stage, the stand-on vessel has the option to take measures to prevent a collision.
- Stage 4: In this stage, the stand-on vessel is obligated to take measures to prevent a collision.

Parameter	Value
S_{13}	Score for overtaking situation.
S_{14}	Score for head-on situation.
S_{15}	Score for crossing situation.
S_{16}	Total score for give-way behavior.
S_{17}	Total score for stand-on behavior.
S_{safety}	Score for pose and range safety at CPA.
S_r	Score for range safety at CPA.
S_Θ	Score for pose safety at CPA.
S_α	Score for contact angle safety at CPA.
S_β	Score for relative bearing safety at CPA.
S_{s2}	Score for stand-on behavior during Stage 2.
S_{s3}	Score for stand-on behavior during Stage 3.
S_{pt}	Score for stand-on behavior during Stage 4.
$S_{\Delta U\uparrow}$	Penalty on speed increase by stand-on vessel.
$S_{\Delta U\downarrow}$	Penalty on speed decrease by stand-on vessel.
S_{delay}	Penalty delayed action by give-way vessel.
S_{nsb}	Penalty on making a non-starboard turn in head-on situations.
S_{sts}	Penalty on starboard-to-starboard passing in head-on situations.
S^{ap}_{Δ}	Score for apparentness of maneuver by give-way vessel.
$S^{ap}_{\Delta\chi}$	Score for apparentness of course change by give-way vessel.
$S^{ap}_{\Delta U}$	Score for apparentness of speed change by give-way vessel.
S_{ah13}	Penalty on the give-way vessel crossing ahead of the stand-on vessel in overtaking situations.
S_{ah15}	Penalty on the give-way vessel crossing ahead of the stand-on vessel in crossing situations.

TABLE XI: Scores and penalties employed in EvalTool

REFERENCES

[1] D. Mahipala and T. A. Johansen, "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)*, Jun. 2023, pp. 896–903, iSSN: 2473-3504.

[2] "Convention on the International Regulations for Preventing Collisions at Sea, 1972 (COLREGs)." [Online]. Available: https://www.imo.org/en/About/Conventions/Pages/COLREG.aspx

[3] A. Vagale, R. Oucheikh, R. T. Bye, O. L. Osen, and T. I. Fossen, "Path planning and collision avoidance for autonomous surface vehicles I: a review," *Journal of Marine Science and Technology*, vol. 26, no. 4, pp. 1292–1306, Dec. 2021. [Online]. Available: https://doi.org/10.1007/s00773-020-00787-6

[4] A. Vagale, R. T. Bye, R. Oucheikh, O. L. Osen, and T. I. Fossen, "Path planning and collision avoidance for autonomous surface vehicles II: a comparative study of algorithms," *Journal of Marine Science and Technology*, vol. 26, no. 4, pp. 1307–1323, Dec. 2021. [Online]. Available: https://doi.org/10.1007/s00773-020-00790-x

[5] H. A. Tran, T. A. Johansen, and R. Negenborn, "Collision avoidance of autonomous ships in inland waterways – A survey and open research problems," *Journal of Physics: Conference Series (JPCS)*, 2023, accepted: 2023-11-17T11:36:08Z Publisher: IOP. [Online]. Available: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/3103213

[6] T. A. Johansen, T. Perez, and A. Cristofaro, "Ship Collision Avoidance and COLREGS Compliance Using Simulation-Based Control Behavior Selection With Predictive Hazard Assessment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3407–3422, Dec. 2016.

[7] I. B. Hagen, D. K. M. Kufoalor, E. F. Brekke, and T. A. Johansen, "MPC-based Collision Avoidance Strategy for Existing Marine Vessel Guidance Systems," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 7618–7623, iSSN: 2577-087X.

[8] T. Tengesdal, T. A. Johansen, T. D. Grande, and S. Blindheim, "Ship Collision Avoidance and Anti Grounding Using Parallelized Cost Evaluation in Probabilistic Scenario-Based Model Predictive Control," *IEEE Access*, vol. 10, pp. 111 650–111 664, 2022.

[9] M. Akdağ, T. I. Fossen, and T. A. Johansen, "Collaborative Collision Avoidance for Autonomous Ships Using Informed Scenario-Based Model Predictive Control," *IFAC-PapersOnLine*, vol. 55, no. 31, pp. 249–256, Jan. 2022.

[10] B.-O. H. Eriksen and M. Breivik, *MPC-Based mid-level collision avoidance for ASVs using nonlinear programming*. Institute of Electrical and Electronics Engineers (IEEE), 2017, accepted: 2018-01-24T14:14:43Z Publication Title: 766-772.

[11] H. Zheng, R. R. Negenborn, and G. Lodewijks, "Trajectory tracking of autonomous vessels using model predictive control," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 8812–8818, Jan. 2014.

[12] M. J. Kochenderfer and T. A. Wheeler, *Algorithms for Optimization*. Cambridge, Massachusetts: The MIT Press, Mar. 2019.

[13] J. C. Spall, *Introduction to Stochastic Search and Optimization*, 1st ed. Hoboken, N.J: Wiley-Interscience, Mar. 2003.

[14] B.-O. H. Eriksen, G. Bitar, M. Breivik, and A. M. Lekkas, "Hybrid Collision Avoidance for ASVs Compliant With COLREGs Rules 8 and 13–17," *Frontiers in Robotics and AI*, vol. 7, 2020.

[15] E. Serigstad, B.-O. H. Eriksen, and M. Breivik, "Hybrid Collision Avoidance for Autonomous Surface Vehicles," *IFAC-PapersOnLine*, vol. 51, no. 29, pp. 1–7, Jan. 2018.

[16] G. Casalino, A. Turetta, and E. Simetti, "A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields," in *OCEANS 2009*, May 2009, pp. 1–8.

[17] K. Bergman, O. Ljungqvist, T. Glad, and D. Axehill, "An Optimization-Based Receding Horizon Trajectory Planning Algorithm," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15 550–15 557, Jan. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896320330810

[18] K. Bergman, O. Ljungqvist, J. Linder, and D. Axehill, "An Optimization-Based Motion Planner for Autonomous Maneuvering of Marine Vessels in Complex Environments," in *2020 59th IEEE Conference on Decision and Control (CDC)*, Dec. 2020, pp. 5283–5290, iSSN: 2576-2370.

[19] ——, "A COLREGs-Compliant Motion Planner for Autonomous Maneuvering of Marine Vessels in Complex Environments," Jan. 2021, arXiv:2012.12145 [math]. [Online]. Available: http://arxiv.org/abs/2012.12145

[20] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *Journal of Field Robotics*, vol. 26, no. 3, pp. 308–333, 2009, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.20285. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20285

[21] H. Lyu and Y. Yin, "Fast Path Planning for Autonomous Ships in Restricted Waters," *Applied Sciences*, vol. 8, no. 12, p. 2592, Dec. 2018, number: 12 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2076-3417/8/12/2592

[22] M. Abdelaal and A. Hahn, "Predictive Path Following and Collision Avoidance of Autonomous Vessels in Narrow Channels," *IFAC-PapersOnLine*, vol. 54, no. 16, pp. 245–251, Jan. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896321015020

[23] T. Tengesdal, T. A. Johansen, and E. F. Brekke, "Ship Collision Avoidance Utilizing the Cross-Entropy Method for Collision Risk Assessment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11 148–11 161, Aug. 2022, conference Name: IEEE Transactions on Intelligent Transportation Systems.

[24] L. Sangmin, K.-Y. Kwon, and J. Joh, "A fuzzy logic for autonomous navigation of marine vehicle satisfying COLREG guidelines," *International Journal of Control, Automation, and Systems*, vol. 2, Jun. 2004.

[25] A. Bakdi and E. Vanem, "Fullest COLREGs Evaluation Using Fuzzy Logic for Collaborative Decision-Making Analysis of Autonomous Ships in Complex Situations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 18 433–18 445, Oct. 2022, conference Name: IEEE Transactions on Intelligent Transportation Systems.

[26] E. Mamdani, "Application of fuzzy algorithms for control of simple dynamic plant," *Proceedings of the Institution of Electrical Engineers*, vol. 121, no. 12, p. 1585, 1974. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/piee.1974.0328

[27] I. B. Hagen, O. Vassbotn, M. Skogvold, T. A. Johansen, and E. F. Brekke, "Safety and COLREG evaluation for marine collision avoidance algorithms," *Ocean Engineering*, vol. 288, p. 115991, Nov. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801823023752

[28] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*, 1st ed. John Wiley & Sons, Ltd, 2011, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119994138.

[29] A. Abbas, "19 - Fuzzy logic control in support of autonomous navigation of humanitarian de-mining robots," in *Using Robots in Hazardous Environments*, Y. Baudoin and M. K. Habib, Eds. Woodhead Publishing, Jan. 2011, pp. 453–475. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9781845697860500198

[30] K. Woerner, M. R. Benjamin, M. Novitzky, and J. J. Leonard, "Quantifying protocol evaluation for autonomous collision avoidance," *Autonomous Robots*, vol. 43, no. 4, pp. 967–991, Apr. 2019. [Online]. Available: https://doi.org/10.1007/s10514-018-9765-y

[31] P. G. Stankiewicz and G. E. Mullins, "Improving Evaluation Methodology for Autonomous Surface Vessel COLREGS Compliance," in *OCEANS 2019 - Marseille*, Jun. 2019, pp. 1–7. [Online]. Available: https://ieeexplore.ieee.org/document/8867549

[32] A. Vagale, R. T. Bye, and O. L. Osen, "Evaluation of Path Planning Algorithms of Autonomous Surface Vehicles Based on Safety and Collision Risk Assessment," in *Global Oceans 2020: Singapore – U.S. Gulf Coast*, Oct. 2020, pp. 1–8, iSSN: 0197-7385. [Online]. Available: https://ieeexplore.ieee.org/document/9389481

Dhanika Mahipala is currently pursuing the Ph.D. degree with Kongsberg Maritime AS, Norway as an early stage researcher of the Marie Skłodowska-Curie Actions ETN AUTOBarge project. He is also affiliated with the Department of Engineering Cybernetics at Norwegian University of Science and Technology (NTNU), Norway. His research is focused on Scenario-based Model Predictive Control for collision avoidance and traffic control of autonomous ships in inland waterways.

Tor Arne Johansen (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 1989 and 1994, respectively. He was a Researcher with SINTEF Electronics and Cybernetics before he was appointed as an Associate Professor with the Norwegian University of Science and Technology in 1997 and later promoted as a Professor in 2001. In 2002, he Co-Founded Marine Cybernetics AS, where he was the Vice President until 2008. He is currently a Principal Researcher with the Center of Excellence on Autonomous Marine Operations and Systems and the Director of the Unmanned Aerial Vehicle Laboratory, NTNU. He has published more than 100 articles in international journals and numerous conference papers and book chapters in the areas of control, estimation, and optimization with applications in the marine, automotive, biomedical, and process industries.