Proceedings of the ASME 2023 42nd International Conference on Ocean, Offshore and Arctic Engineering
OMAE2023
June 11–16, 2023, Melbourne, Australia

OMAE2023-103740

MODULAR COLLISION AVOIDANCE USING PREDICTIVE SAFETY FILTERS

Aksel Vaaler[1], Haakon Robinson[1,*], Trym Tengesdal[1], Adil Rasheed[1]

[1]Norwegian University of Science and Technology, Trondheim, Norway

ABSTRACT

The number of maritime projects is increasing yearly, including offshore applications, underwater robotics for ocean condition monitoring, and autonomous ship transport. Many of these activities are safety-critical, making it essential to have a robust closed-loop control system that satisfies constraints arising from underlying physical limitations and safety aspects. However, this is often challenging to achieve for real-world systems. For example, autonomous ships at sea have non-linear and uncertain dynamics and are subject to numerous time-varying environmental disturbances such as waves, currents, and wind. There is growing interest in using machine learning-based approaches to adapt these systems to more complex scenarios. However, there is currently no standard framework to guarantee the safety and stability of such systems. Predictive safety filters have emerged recently as a valuable method for ensuring constraint satisfaction, even when unsafe control inputs are used. The safety filter approach leads to a modular separation of the problem, allowing the usage of arbitrary control policies in a task-agnostic way. In this work, a predictive safety filter is developed to ensure anti-grounding and ship collision avoidance for a small prototype ferry. The filter takes in a nominal input sequence from a potentially unsafe controller and solves an optimization problem to compute a minimal perturbation of the nominal control inputs, which adheres to physical and safety-related constraints. The system is validated by simulations for several realistic scenarios with map data from Trondheim, Norway. It is demonstrated that the predictive safety filter can avoid collisions with static and dynamic obstacles. The predictive safety filter approach is flexible and can be used to improve the robustness of various offshore applications, e.g. wind turbine stabilization, autonomous vessels, and marine robotics.

Keywords: Safety filter, Marine craft, Safe learning

*Corresponding author

1. INTRODUCTION

Learning-based control approaches are gaining popularity today, particularly for highly complex, nonlinear, and stochastic systems where standard control design methods do not always perform satisfactorily [1–3]. Learning-based methods can discover new patterns and actions, adapting to the environment for better performance. However, in practice, it is challenging to guarantee that such a controller will always respect system constraints, especially during the initial training phase [4]. For example, an autonomous ship with azimuthal thrusters must avoid all collisions while keeping in mind the maximum output and turning rate of its thrusters. These issues limit the applicability of such methods to many real-world systems.

A natural solution to this problem is to use a *predictive safety filter* [5]. This auxiliary system detects when the controlled system is headed towards an unsafe state and immediately falls back to a control policy known to be safe. This setup resembles how a driving instructor might intervene during a lesson. The essential characteristic of a predictive safety filter is that it attempts to find a minimal modification to the control inputs that respects the constraint set over a finite number of future time steps. This separation of concerns yields a modular approach, where the learning-based component can be freely designed and optimized while the safety filter guarantees constraint satisfaction. The advantages are two-fold. First, the predictive safety filter is a more straightforward optimization problem than a predictive controller that optimizes performance and safety. Secondly, the learning-based component can be trained using more complex, sparser cost functions without compromising the convergence of the safety filter, which can yield higher performance after training. For maritime systems, the predictive safety filter is comparable to collision avoidance (COLAV) systems for autonomous ships.

A rich set of studies on automatic maritime collision avoidance exist today, and recent review articles can be found that summarize the majority of state-of-the-art methods [6, 7]. A non-exhaustive review is given here. A common way of structuring the COLAV system is using a hierarchy of planners [8–11]. The

authors in [11] divide the COLAV problem into three levels separated by their timescale: (a) High-level planning, (b) Mid-level planning, and (c) Low-level planning. A high-level planner generates a trajectory or path to the final destination, considering static obstacle data from, e.g., Electronic Navigational Charts (ENC). The trajectory is represented by a series of waypoints η_i to follow in marine navigation. Because of the larger timescales (minutes/hours/days), the dynamics of the ship are often simplified, reducing this to a path-finding problem. Grid-based or lattice-based methods typically discretize the map geometry [12] or partition it into, e.g., Voronoi cells [13]. Randomized sampling methods that explore the space using random steps have also been successfully applied [14, 15].

Mid-level COLAV planning algorithms try to avoid static and dynamic obstacles near the own-ship. Dynamic obstacles are typically detected and tracked online using the onboard exteroceptive sensors and Automatic Identification System (AIS) data. At this level and timescale (seconds/minutes), the planner should simulate the vessel to suggest feasible maneuvers for the ship to execute. Because there may be multiple obstacles, the suggested manoeuvre should ideally guarantee safety at all time steps considered in its horizon. For sampling-based methods such as Scenario-based model predictive control (MPC) approaches, one considers multiple possible maneuvers that the own-ship (and dynamic obstacles) can take at multiple decision points in time [16]. Other methods in this category include e.g. A-star-based planning [17], Voronoi-diagram-based planning [13] and Rapidly-exploring Random Trees (RRT) [14, 18].

At the lowest level (seconds or milliseconds), the planner must be able to react quickly to unexpected situations. Due to the short time scales involved, these methods are also referred to as *reactive* COLAV. Such situations can arise when a nearby vessel makes a sudden and dangerous maneuver or loses control of the vessel. Higher-level planners can also fall back to reactive COLAV if there is high uncertainty regarding the positions of nearby ships due to sensor malfunctions. Classic examples include Potential Field (PF) methods [19, 20] and Velocity Obstacle (VO) methods. VO based approaches identify unsafe own-ship velocities that result in collision [21, 22]. These computations are very efficient when all ships are modeled with constant velocities. However, this assumption is only valid on short-time scales.

The predictive safety filter is best formulated as an optimization problem over multiple time steps [5] and is, therefore, most comparable to a mid-level COLAV algorithm. The novelty of this work is the implementation and validation of a safety filter on a maritime vessel that can act as a safety harness around, e.g., learning-based planning algorithms higher up in the planning hierarchy. These methods have the potential to enable safer navigation, which is illustrated in the context of a passenger ferry voyaging in a narrow water canal. The contributions of this article are thus:

- The implementation of a predictive safety filter on a passenger ferry platform for both anti-grounding and ship collision avoidance.

FIGURE 1: THE MILLIAMPERE FERRY

- Validation of the method via simulation

This paper is structured as follows. Relevant modeling of the own-ship platform is presented in Section 2. The safety filter is detailed in Section 3. Results and their discussions are presented in Section 4 and finally, Section 5 concludes the current work.

2. MODELING

This work considers milliAmpere 1, a small passenger ferry prototype intended for urban environments. It aims to be a safe, flexible, and on-demand replacement for bridges [23]. The ferry is small and maneuverable, with two azimuthal thrusters mounted on the underside, which allow it to navigate crowded waterways with other ships. The following subsections present the equations of motion of the craft, the thruster dynamics, and a simple Line-of-Sight (LOS) controller for waypoint tracking. A more in-depth model derivation and experimental identification of the parameters can be found in [24].

2.1 Kinematics and Kinetics

The model formulation is based on the 3 degrees of freedom (DOF) Robot-Inspired Model for marine craft [25]. The position $[x \quad y]$ is expressed in North-East (NE) coordinates, and the yaw ψ is defined as the angle between the North axis and the forward-facing direction of the vessel. The forward and side velocities of the ship (relative to a coordinate frame fixed to the ship) are denoted u and v, while the yaw rate is denoted r. The equations of motion are given by

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \mathbf{R}(\psi)\boldsymbol{\nu} \\ \mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} &= \boldsymbol{\tau} \end{aligned} \tag{1}$$

where the pose of the ship is $\boldsymbol{\eta} = [x \quad y \quad \psi]^\top$, the velocity is $\boldsymbol{\nu} = [u \quad v \quad r]^\top$, and the rigid body mass $\mathbf{M}$, Coriolis matrix $\mathbf{C}(\boldsymbol{\nu})$ and damping matrix $\mathbf{D}(\boldsymbol{\nu})$ are all 3×3 matrices. The generalized force $\boldsymbol{\tau}$ represents all forces acting on the ship, including that of the actuators. External forces due to waves, wind, and currents are neglected for simplicity. The kinematics simplify to:

$$\dot{\boldsymbol{\eta}} = \mathbf{R}(\psi)\boldsymbol{\nu} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ r \end{bmatrix} \tag{2}$$

and the kinetics can be written

$$\dot{\boldsymbol{\nu}} = \mathbf{M}^{-1}\left(\boldsymbol{\tau} - \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu}\right) \tag{3}$$

The inertia and Coriolis matrices can be written as follows [24]:

$$\mathbf{M} = \begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & m_{23} \\ 0 & m_{32} & m_{33} \end{bmatrix}, \quad \mathbf{C}(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & c_{13}(\boldsymbol{\nu}) \\ 0 & 0 & c_{23}(\boldsymbol{\nu}) \\ c_{31}(\boldsymbol{\nu}) & c_{32}(\boldsymbol{\nu}) & 0 \end{bmatrix}$$
$$\begin{aligned} c_{13}(\boldsymbol{\nu}) &= -m_{22}v - m_{23}r \\ c_{23}(\boldsymbol{\nu}) &= m_{11}u \\ c_{31}(\boldsymbol{\nu}) &= -c_{13}(\boldsymbol{\nu}) \\ c_{32}(\boldsymbol{\nu}) &= -c_{23}(\boldsymbol{\nu}) \end{aligned} \tag{4}$$

Note that these matrices combine rigid body and "added mass" terms, the latter being a virtual mass added to the system due to the volume of fluid that is accelerated along with the vessel. This modification reduces the identification problem to identifying suitable values for m_{ij}. The decoupled damping matrix is written as:

$$\mathbf{D}(\boldsymbol{\nu}) = \begin{bmatrix} d_{11}(\boldsymbol{\nu}) & 0 & 0 \\ 0 & d_{22}(\boldsymbol{\nu}) & d_{23}(\boldsymbol{\nu}) \\ 0 & d_{32}(\boldsymbol{\nu}) & d_{33}(\boldsymbol{\nu}) \end{bmatrix}$$
$$\begin{aligned} d_{11}(\boldsymbol{\nu}) &= -X_u - X_{|u|u}|u| - X_{uuu}u^2 \\ d_{22}(\boldsymbol{\nu}) &= -Y_v - Y_{|v|v}|v| - Y_{|r|v}|r| - Y_{vvv}v^2 \\ d_{23}(\boldsymbol{\nu}) &= -Y_r - Y_{|v|r}|v| - Y_{|r|r}|r| \\ d_{32}(\boldsymbol{\nu}) &= -N_v - N_{|v|v}|v| - N_{|r|v}|r| \\ d_{33}(\boldsymbol{\nu}) &= -N_r - N_{|v|r}|v| - N_{|r|r}|r| - N_{rrr}r^2 \end{aligned} \tag{5}$$

All coefficients are identified experimentally. Refer to [24] for the precise values.

2.2 Actuator dynamics

The ferry has two azimuthal thrusters that rotate freely, as shown in Figure 2. These angles are denoted as $\boldsymbol{\alpha}$, the motor speeds (in RPM) as $\boldsymbol{\omega}$, and the resulting net thrusts as $\boldsymbol{f}$. The subscript i refers to the properties of an individual thruster, e.g., α_i is the angle of the ith thruster. From Figure 2, the net force and moment on the ship are:

$$\boldsymbol{\tau} = \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_m \end{bmatrix} = \mathbf{T}(\boldsymbol{\alpha})\mathbf{f} = \begin{bmatrix} \cos\alpha_1 & \cos\alpha_2 \\ \sin\alpha_1 & \sin\alpha_2 \\ \ell_1\sin\alpha_1 & \ell_2\sin\alpha_2 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}. \tag{6}$$

Choosing the thrust $\mathbf{f}$ to achieve some force $\boldsymbol{\tau}$ is known as the *control allocation* problem. The simplest approach is to extend $\mathbf{T}(\boldsymbol{\alpha})$ by decomposing the thrust vectors into their x and y components:

$$\begin{aligned} \boldsymbol{\tau} &= \mathbf{T}_e\mathbf{f}_e \\ &= \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & \ell_1 & 0 & \ell_2 \end{bmatrix} \begin{bmatrix} f_{1,x} \\ f_{1,y} \\ f_{2,x} \\ f_{2,y} \end{bmatrix} \end{aligned} \tag{7}$$

thereby yielding the linear transformation $\mathbf{T}_e$. Appropriate values for $\mathbf{f}$ and $\boldsymbol{\alpha}$ can then be found by taking the pseudoinverse of $\mathbf{T}_e$.

The control allocation mapping is denoted as:

$$(\boldsymbol{\alpha}, \mathbf{f}) = \mathbf{T}^{-1}(\boldsymbol{\tau}) \quad \text{s.t.} \quad \begin{cases} \mathbf{f}_e = \mathbf{T}_e^{\dagger}\boldsymbol{\tau} \\ f_i = \sqrt{f_{i,x}^2 + f_{i,y}^2} & i \in \{1,2\} \\ \alpha_i = \arctan2(f_{i,y}, f_{i,x}) & i \in \{1,2\} \end{cases} \tag{8}$$

This approach does not consider actuator constraints and can therefore yield infeasible control sequences in practice. In this case, the control allocation problem $\mathbf{T}^{-1}(\boldsymbol{\tau})$ is best formulated as a constrained Nonlinear Program (NLP) [25]. Control allocation is implicitly handled by the predictive safety filter framework when actuator constraints are included in the formulation. The thrust f_i is related to the motor RPM ω_i by the following invertible function:

$$\mathbf{f} = \mathbf{L}(\boldsymbol{\omega}) \tag{9}$$

This mapping was determined experimentally and modeled as an invertible polynomial [24]. The relationship between the desired motor RPM $\boldsymbol{\omega}_d$ and the actual value $\boldsymbol{\omega}$ is modeled as a proportional control law with gain K_{ω_i} corresponding to thruster i:

$$\dot{\omega}_i = K_{\omega_i}(\omega_{d,i} - \omega_i) \tag{10}$$

Finally, the azimuthal thrusters have a constant turning rate K_{α_i}. This is approximated with the help of the function:

$$\lambda(\beta;\epsilon) = \frac{\mathrm{ssa}(\beta)}{\sqrt{\mathrm{ssa}(\beta)^2 + \epsilon}} \tag{11}$$

where $\mathrm{ssa}(\cdot)$ is the smallest signed angle function. Given the desired angles $\boldsymbol{\alpha}_d$ and the actual angles $\boldsymbol{\alpha}$, the turning rate for the ith thruster is then:

$$\dot{\alpha}_i = K_{\alpha_i}\lambda\left(\alpha_{d,i} - \alpha_i; \epsilon_i\right) \tag{12}$$

where $K_{\alpha_i}, \epsilon_i, K_{\omega_i}$ are thruster-specific parameters. Refer to [24] for the precise values.

2.3 Full dynamics

The full dynamics of the ferry can now be written as:

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \mathbf{R}(\psi)\boldsymbol{\nu} \\ \dot{\boldsymbol{\nu}} &= \mathbf{M}^{-1}\left(\mathbf{T}(\boldsymbol{\alpha})\,\mathbf{L}(\boldsymbol{\omega}) - \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu}\right) \\ \dot{\boldsymbol{\omega}} &= \mathbf{K}_{\omega}(\boldsymbol{\omega}_d - \boldsymbol{\omega}) \\ \dot{\alpha}_i &= K_{\alpha_i}\lambda\left(\alpha_{d,i} - \alpha_i; \epsilon_i\right) \end{aligned} \tag{13}$$

The inputs to this model are the desired thruster angles $\boldsymbol{\alpha}_d$ and thrust vector $\boldsymbol{\omega}_d$, which are typically computed from a desired force $\boldsymbol{\tau}_d$ using Equation (8). However, as will be seen later, it is simpler to skip this step when formulating the predictive safety filter and select $\boldsymbol{\alpha}_d$ and $\boldsymbol{\omega}_d$ as decision variables.

2.4 Naive controller

The safety filter is validated using a simple LOS guidance law controller that tracks a path specified by a sequence of waypoints. When the own-ship comes within a specified radius of the current target waypoint, the reference is switched to the next waypoint in the sequence [25]. The LOS guidance law is defined as:

$$\psi_d = \pi_p - \arctan\left(\frac{y_e}{\Delta}\right) \tag{14}$$

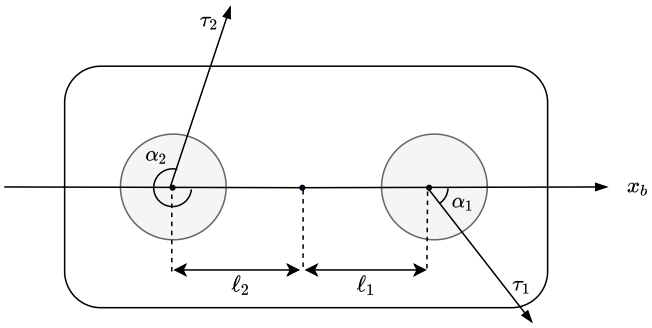


FIGURE 2: ACTUATORS ON THE MILLIAMPERE FERRY. THERE ARE TWO AZIMUTHAL THRUSTERS THAT CAN ROTATE FREELY.

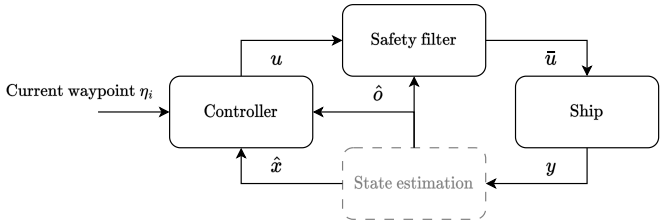


FIGURE 3: INTERACTIONS BETWEEN SAFETY FILTER AND TYPICAL GUIDANCE SYSTEM. STATE ESTIMATION IS NOT INCLUDED IN THIS WORK.

Where π_p is the angle of the vector from the previous waypoint to the current target waypoint defined in the NE-frame, Δ is the *look-ahead distance*, and y_e is the *cross-track error*, calculated from Equation (12.43) in [25]. The desired force on the ship is then defined as:

$$\boldsymbol{\tau}_d = \begin{bmatrix} \tau_{x,d} \\ \tau_{y,d} \\ \tau_{m,d} \end{bmatrix} = \begin{bmatrix} T_{x,d} \\ 0 \\ K_p(\psi_d - \psi) \end{bmatrix} \tag{15}$$

where the $\boldsymbol{\tau}_d$ vector is a constant desired force $T_{x,d}$ in the forward ship direction, zero desired force laterally, and a desired moment proportional to the difference between the desired heading and actual heading. Finally, the desired force vector is mapped to control inputs $(\mathbf{f}_d, \boldsymbol{\alpha}_d)$ by:

$$\begin{aligned} (\mathbf{f}_d, \boldsymbol{\alpha}_d) &= \mathbf{T}^{-1}(\boldsymbol{\tau}_d) \\ (\boldsymbol{\alpha}, \boldsymbol{\omega}) &= (\boldsymbol{\alpha}_d, \mathbf{L}^{-1}(\mathbf{f}_d)) \end{aligned} \tag{16}$$

These control inputs are likely to be unsafe or infeasible, and are therefore passed on to the safety filter for evaluation and modification.

3. SAFETY FILTER

The predictive safety filter is formulated as an optimization problem constrained by the ferry dynamics, state and actuator limits, and anti-collision conditions. The objective of the problem is to find a minimal perturbation $\boldsymbol{\delta}$ to the input $\boldsymbol{\tau}$ such that the safety requirements are satisfied. The modified input $\bar{\boldsymbol{\tau}}$ is then passed to the system. Figure 3 shows how the safety filter interacts with an idealized guidance and navigation system. Note that perfect knowledge of the ship's and obstacles' state is assumed; handling uncertainty and robust constraint satisfaction is left as future work.

The system dynamics given by Equation (13) are discretized using an explicit Runge-Kutta method of order 1 with constant time-step h. The inputs and states at each time step are taken as decision variables, also known as direct multiple shooting. In the following, the subscripts k are used to refer to the time step, i for vector elements, and j for the jth obstacle. The full NLP is written as

$$\begin{aligned} \min_{\boldsymbol{\eta}_k, \boldsymbol{\nu}_k, \boldsymbol{\alpha}_k, \boldsymbol{\omega}_k \boldsymbol{\delta}_k} \quad & \sum_{k=1}^{N} \gamma_\alpha^2 \|\boldsymbol{\delta}_{\alpha,k}\|^2 + \gamma_\omega^2 \|\boldsymbol{\delta}_{\omega,k}\|^2 \\ \text{s.t.} \quad & \alpha_{lb} \leq \alpha_{i,k} \leq \alpha_{ub} \quad \forall i,k \\ & \omega_{lb} \leq \omega_{i,k} \leq \omega_{ub} \quad \forall i,k \\ & \bar{\boldsymbol{\alpha}}_k = \boldsymbol{\alpha}_k + \boldsymbol{\delta}_{\alpha,k} \\ & \bar{\boldsymbol{\omega}}_k = \boldsymbol{\omega}_k + \boldsymbol{\delta}_{\omega,k} \\ & \boldsymbol{\eta}_{k+1} = \boldsymbol{\eta}_k + h\mathbf{R}(\psi)\boldsymbol{\nu}_k \\ & \boldsymbol{\nu}_{k+1} = \boldsymbol{\nu}_k + h\mathbf{M}^{-1}[\boldsymbol{\tau}(\bar{\boldsymbol{\alpha}}_k, \bar{\boldsymbol{\omega}}_k) - \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu}] \\ & |\bar{\alpha}_{i,k+1} - \bar{\alpha}_{i,k}| \leq \Delta\alpha \quad \forall i,k \\ & |\bar{\omega}_{i,k+1} - \bar{\omega}_{i,k}| \leq \Delta\omega \quad \forall i,k \\ & (\mathbf{A}\mathbf{p}_k - \mathbf{b}) + d \leq 0 \quad \forall k \\ & g_j(\mathbf{p}_k, \mathbf{o}_{j,k}) > 0 \quad \forall j,k \\ & \mathbf{o}_{j,k+1} = \mathbf{o}_{j,k} + \mathbf{v}_j h \quad \forall j,k \end{aligned} \tag{17}$$

The position $[x \quad y]$ of the own-ship at time-step k is denoted $\mathbf{p}_k$, and d is the safe radius. The perturbed inputs $\bar{\boldsymbol{\alpha}}$ and $\bar{\boldsymbol{\omega}}$ are bounded by $(\alpha_{lb}, \alpha_{ub})$ and $(\omega_{lb}, \omega_{ub})$ respectively, and rate-limited by the constants $\Delta\alpha$ and $\Delta\omega$ respectively. The rate-limiting constraints were chosen instead of directly modeling the actuator dynamics given in Equation (13). This was found to simplify the problem greatly and is a sufficiently accurate approximation, particularly for $\boldsymbol{\alpha}$. Anti-grounding is achieved by defining a safe water region using the linear constraint set defined by $\mathbf{A}$ and $\mathbf{b}$, regularly updated as the vessel advances along the path. Section 3.1 describes how $\mathbf{A}$ and $\mathbf{b}$ are computed. In this formulation, both static and dynamic obstacles are modeled as ellipses centered at the positions $\boldsymbol{o}_{j,k} = [x_{obs} \quad y_{obs}]$, where k again refers to the time-step and j is the obstacle index. This is achieved using signed distance functions $g_j(\cdot)$, which measure the distance between the safe radius of the own-ship and the surface of the jth ellipse, and are defined precisely in Section 3.2. The distance function returns a negative value if the safe radius and the ellipse intersect. Dynamic obstacle movement is modeled with a constant velocity $\mathbf{v}_j$, which updates the position $\boldsymbol{o}_{j,k}$ at each time step. Further implementation details and parameter values can be found in Section 3.3.

3.1 Anti-grounding constraint representation

In the vicinity of land, the area in which the ship can safely navigate within a given timespan is generally a non-convex set. This constraint is challenging to model in an optimization problem. Instead, a convex subset of this area is identified online from cartographic data. This area is referred to as the Convex Safe

Set (CSS). The procedure is based on the algorithm presented by [26] and is summarized in Algorithm 1. The algorithm is implemented using the Shapely python package [27]. The estimated CSS is updated every 10 seconds in the simulation. The advantages of this approach are that it is relatively cheap to compute the CSS, it can be represented as a linear constraint set, and it can be pre-computed at regular points along a nominal trajectory if needed. The disadvantage is that a CSS can be overly conservative, especially when the vessel is close to either land or some obstacle.

Algorithme 1 : INNER CONVEX SAFE SET ESTIMATION

$p \leftarrow$ Current position of ship;
$H \leftarrow$ Rectangle with center at p and width,height = D_{max};
$S \leftarrow$ Polygonal representation of safe region around p, extracted from map data;
$S' \leftarrow H \cap S$;
$B \leftarrow$ Boundary(S');
$C \leftarrow$ Empty table for storing constraints;

while $B \neq Empty$ **do**
  $p_n \leftarrow$ Nearest point on B, as seen from p;
  $C' \leftarrow$ Constraint line orthogonal to $(p_n - p)$, with mid-point at p_n;
  Remove segments of B that are outside of constraint line C', as seen from p;
  Add constraint line C' to table C
return C

3.2 Obstacle constraint representation

In this work, obstacles are represented as oriented ellipses, which provides more design flexibility than a circular representation and more closely matches the profile of a typical vessel. In theory, it is possible to use Algorithm 1 to compute a safe set without obstacles. However, requiring the safe set to be convex can be conservative, particularly when the vessel must perform a sharp turn. Figure 5 illustrates such an example. Here the vessel must navigate between two obstacles (denoted obs_1 and obs_2), and the nominal trajectory is already safe. However, due to the angle of the approach, there is no possible convex set that contains the nominal trajectory and the own-ship. The safety filter, therefore, corrects course and takes an evasive maneuver instead. This issue arguably resolves itself when the ship passes obs_1, and a CSS that contains the nominal path can then be found. However, depending on the vessel's speed, it may already be too late to correct course again and avoid obs_2. In the best case, the vessel may take the turn later than desirable, which breaks Rules 8 and 16 of Convention on the International Regulations for Preventing Collision at Sea (COLREGS) (i.e., maneuvers must be made or signaled in ample time). By representing the obstacles directly using additional constraints, the predictive safety filter can plan much more effectively. This problem is nonetheless very tractable despite the added complexity, as will be shown experimentally

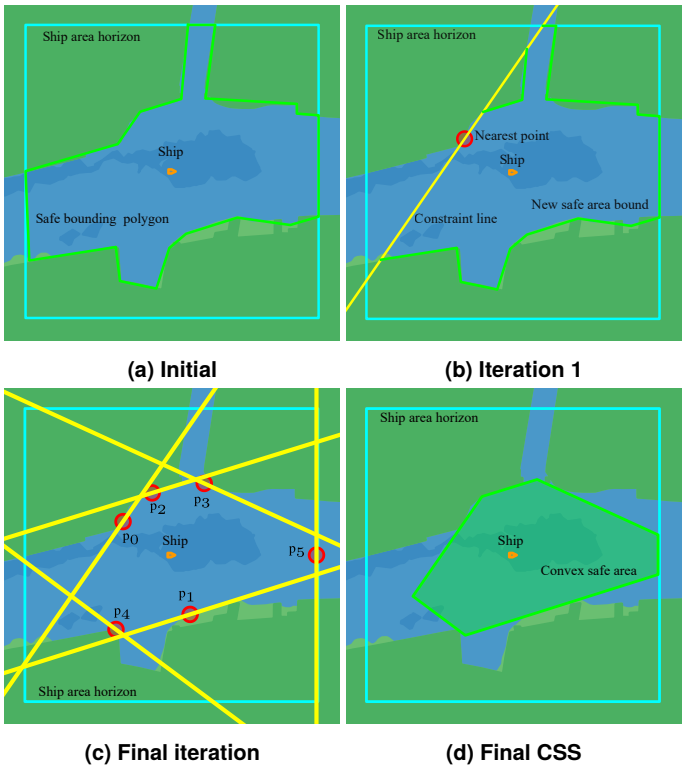


FIGURE 4: ALGORITHM FOR COMPUTING CONVEX SAFE SET BY SAMPLING NEAREST POINTS ON CONSTRAINTS (MARKED P_i).

in Section 4 that this problem is still very tractable despite the added complexity.

The parameters for the jth obstacle are defined as the tuple $(\mathbf{o}_j, \mathbf{v}_j, a_{obs,j}, b_{obs,j}, \theta_{obs,j})$, where $\mathbf{o}_j$ is the coordinate vector for the center of the obstacle, $\mathbf{v}_j$ is the velocity of the obstacle, $a_{obs,j}$ and $b_{obs,j}$ is $1/2$ the length of the ellipse in its semi-major and semi-minor axes respectively, and θ_{obs} denotes the angle between the coordinate-frame x-axis and the semi-major axis of the elliptical obstacle. The formula for an elliptical disk can be written as:

$$E(\boldsymbol{x}, a, b) \leq 0 \tag{18}$$

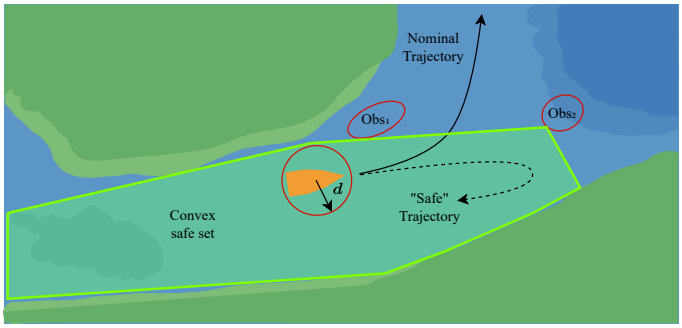


FIGURE 5: ILLUSTRATION OF A CONVEX SAFE SET THAT ALSO TAKES OBSTACLES (MARKED OBS) INTO ACCOUNT. DESPITE THE NOMINAL TRAJECTORY BEING SAFE, THE REQUIREMENT THAT THE SAFE SET BE CONVEX IS OVERLY CONSERVATIVE.

where

$$E(\boldsymbol{x}, a, b) = \frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} - 1 \tag{19}$$

The constraint function $g_j(\mathbf{p}, \mathbf{o})$ for the jth obstacle is defined as:

$$g_j(\mathbf{p}, \mathbf{o}) = E\left[\boldsymbol{R}(\theta_{obs})(\mathbf{p} - \mathbf{o}),\ a_{obs} + d,\ b_{obs} + d\right] \tag{20}$$

where $\boldsymbol{R}(\theta_{obs})$ is the rotation matrix:

$$\boldsymbol{R}(\theta_{obs}) = \begin{bmatrix} \cos\theta_{obs} & -\sin\theta_{obs} \\ \sin\theta_{obs} & \cos\theta_{obs} \end{bmatrix} \tag{21}$$

Equation (20) can be expanded as:

$$g_j(\mathbf{p}, \mathbf{o}) = \frac{(\cos\theta_{obs}(x - x_{obs}) - \sin\theta_{obs}(y - y_{obs}))^2}{(a_{obs} + d)^2} + \frac{(\sin\theta_{obs}(x - x_{obs}) + \cos\theta_{obs}(y - y_{obs}))^2}{(b_{obs} + d)^2} - 1 \tag{22}$$

The dynamic obstacle motion over the prediction horizon is taken into account using a movement constraint with the form $\mathbf{o}_{i,k+1} = \mathbf{o}_{i,k} + \mathbf{v}_i h$, where the linear velocity of the ith obstacle is denoted $\mathbf{v}_i$. Straight-line obstacle motion is thus assumed, which is deemed reasonable as longer time horizons are not considered.

3.3 Implementation

The `CasADi` symbolic framework is used to efficiently encode the resulting optimization scheme [28], which is then solved using the open-source `IPOPT` software [29]. A time-step of $h = 0.5s$ yielded sufficiently accurate state predictions for the relatively slow dynamics of the ship. Furthermore, a prediction horizon of $N = 30$ was selected, giving a fair balance between performance and computational complexity. The total number of decision variables to be computed for each solver iteration is $N(n_x + n_u) = 30(6 + 4) = 300$. All experiments were run on a consumer-grade laptop. The cost parameters were chosen as follows:

$$\begin{aligned} \gamma_\alpha^2 &= \frac{1}{(\alpha_{ub} - \alpha_{lb})^2} \\ \gamma_\omega^2 &= \frac{10}{(\omega_{ub} - \omega_{lb})^2} \end{aligned} \tag{23}$$

Due to the relatively high cost of perturbing $\boldsymbol{\omega}$, the safety filter prioritizes turning the ship by modifying $\boldsymbol{\alpha}$, rather than slowing down by setting $\bar{\boldsymbol{\omega}} \approx \mathbf{0}$. Table 1 shows the parameters that were used in the experiments.

TABLE 1: SAFETY FILTER PARAMETERS

Parameter	Value	Description
Δ	100 m	Lookahead distance
K_ψ	200	Heading gain
$T_{x,d}$	350 N	Constant forward force
N	30	Horizon length
h	0.5 s	time-step
$\Delta\alpha$	0.5 rad	Rate limit ($\boldsymbol{\alpha}$)
$\Delta\omega$	0.875 krpm	Rate limit ($\boldsymbol{\omega}$)
α_{lb}	$-\pi$ [rad]	Lower bound ($\boldsymbol{\alpha}$)
α_{ub}	π [rad]	Upper bound ($\boldsymbol{\alpha}$)
ω_{lb}	-4 krpm	Lower bound ($\boldsymbol{\omega}$)
ω_{ub}	4 krpm	Upper bound ($\boldsymbol{\omega}$)
d	5 m	Own-ship safe radius
γ_α	0.159 rad^{-1}	Perturbation cost ($\boldsymbol{\alpha}$)
γ_ω	0.39 krpm^{-1}	Perturbation cost ($\boldsymbol{\omega}$)

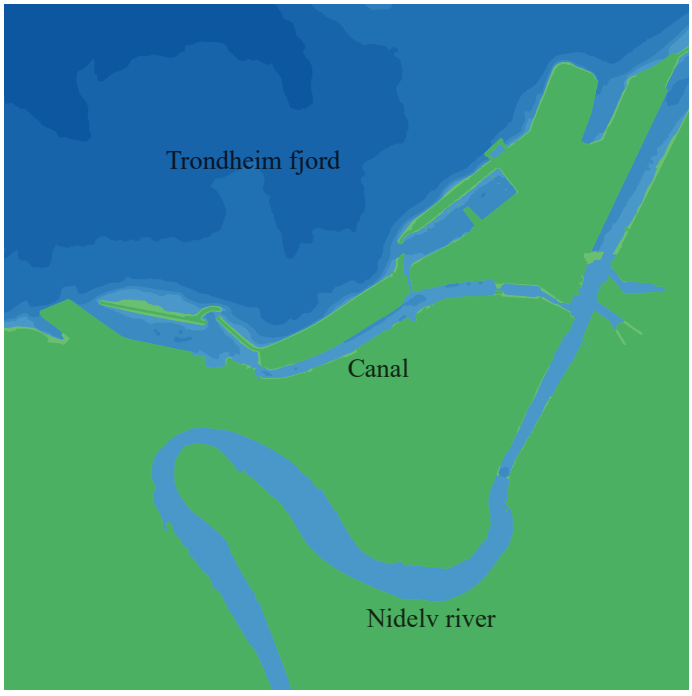


FIGURE 6: SIMPLIFIED NAVIGATIONAL CHART OF TRONDHEIM AREA

4. RESULTS AND DISCUSSIONS

Realistic scenarios in the Trondheim canal were constructed using the `seacharts` library for Python [30]. All scenarios were designed in the Trondheim canal, as shown in Figure 6, and can be summarized as:

(a) Two wide barriers blocking the canal

(b) Planned path cuts through land

(c) Curved barrier forcing the ship to backtrack

(d) Case (c) with a longer prediction horizon

(e) Single incoming ship

(f) Two incoming ships

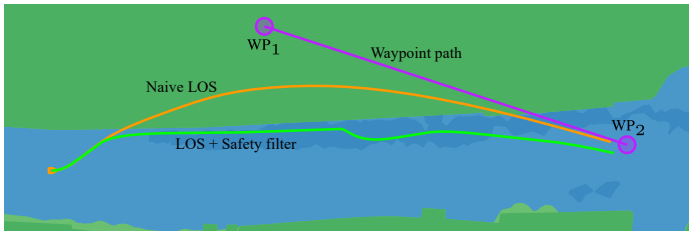
Figure 7 shows how the safety filter corrects unsafe control inputs to avoid collisions in each test case. The results of each case are discussed individually, but cases (c) and (d) are emphasized because they demonstrate the consequences of handling goal fulfillment and safety separately. The actual perturbations done to the azimuthal angles in cases (c) and (d) are shown in Figures 8 and 9 respectively, and the computation time throughout each case is plotted in Figure 10. The motor speed perturbations were relatively small due to the high cost placed on them and were therefore not included.

Figure 7a shows that the system can perform effective anti-grounding even when the reference waypoint is infeasible. Notably, a kink is introduced into the otherwise smooth trajectory. This abnormality is possibly due to a small outcrop of land further along the path, yielding an overly conservative anti-grounding safe set. Further investigation found that this behavior can also occur when there is a tight chokepoint in the canal, as seen in Figure 11.

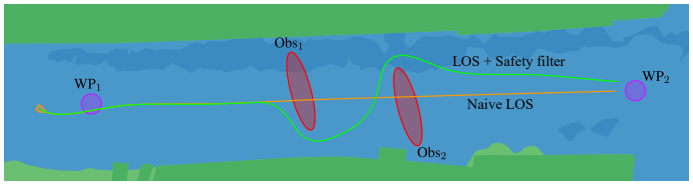
Figure 7b demonstrates that the safety filter can avoid large obstacles by turning early. This scenario represents the worst-case, where the own-ship needs to move from one canal bank to the other in relatively little time.

Figures 7c and 7d show how the behavior of the safety filter for a planning horizon of 30 s and 50 s respectively. In the first case, the concavity of the barrier forces the own-ship to backtrack. In this situation, the safety filter has to force the own-ship to move in the opposite direction of the desired path. A shorter planning horizon causes the own-ship to turn quite late, causing an aggressive turning maneuver. The looping behavior can be explained by the fact that the safety filter no longer activates when the own-ship moves away, such that the naive controller moves the own-ship towards the barrier again. This issue is easily mitigated by increasing the planning horizon, as shown in Figure 7d. The longer planning incurs a higher solve-time for the safety filter, which occasionally reaches the threshold for computation time. While this might cause the solver to return a sub-optimal solution, in practice, the solve-times quickly decay due to the warm-start strategy (see Figure 10b).

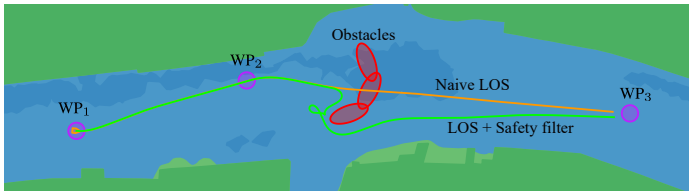
Figure 7e shows how the safety filter can easily handle a large moving obstacle. The positions of the own-ship and the dynamic obstacle are shown at three different time steps for clarity. The safety filter can also avoid multiple dynamic obstacles, as shown in Figure 7f. Again, the positions of the own-ship and obstacles are plotted at multiple time steps. These two test cases highlight that the safety filter does not follow the COLREGS as outlined in [31]. Specifically, the own-ship should give way to the right (rules 14 and 15), with a maneuver initiated in ample time to signal its intention to the other ships (rule 8). The simple objective function in Equation (17) does not capture these considerations. Instead, the role of the safety filter is to serve as a "last line of defense" for a learning-based algorithm trained to maneuver correctly in traffic situations.



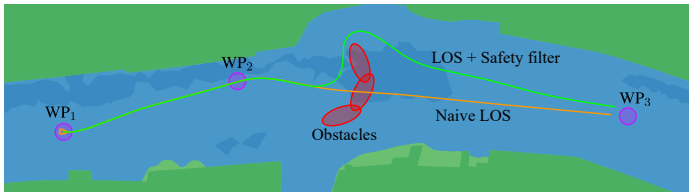
(a) Anti-grounding



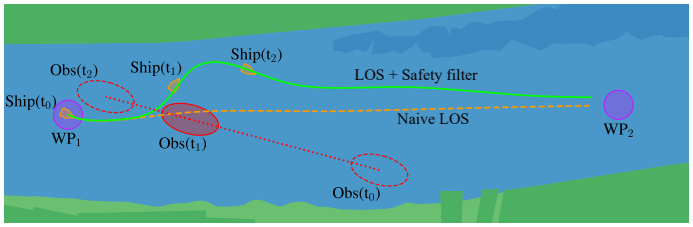
(b) Static obstacles



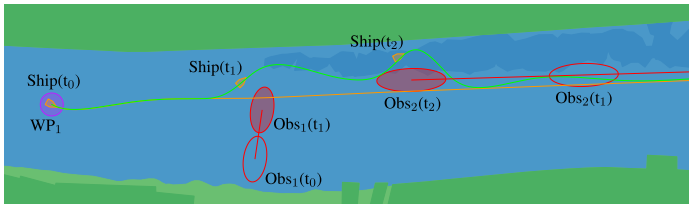
(c) Concave static obstacles: Horizon $N = 30$



(d) Concave static obstacles: Horizon $N = 50$

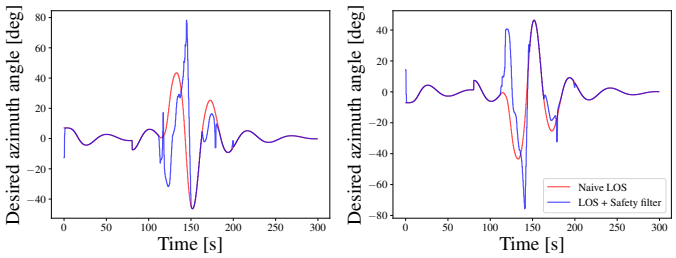


(e) Single dynamic obstacle



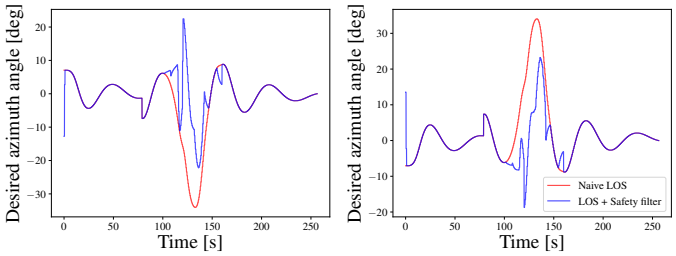
(f) Multiple dynamic obstacles

FIGURE 7: OVERVIEW OF WAYPOINT (MARKED WP_i) TRACKING RESULTS FOR ALL TEST CASES.



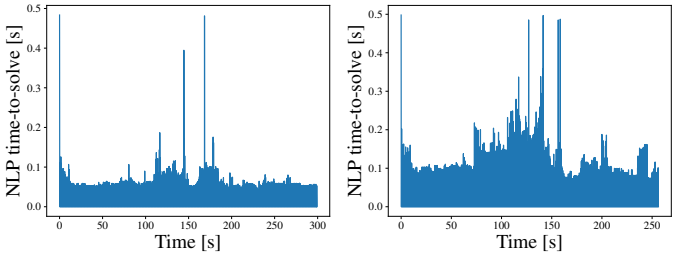
(a) α_1 (b) α_2

FIGURE 8: AZIMUTH ANGLE CONTROL INPUT MODIFICATION FOR CASE (C)



(a) α_1 (b) α_2

FIGURE 9: AZIMUTH ANGLE CONTROL INPUT MODIFICATION FOR CASE (D)



(a) Case (c) (b) Case (d)

FIGURE 10: SOLVE TIME OVER THE COURSE OF THE TRAJECTORY FOR CASES (C) AND (D)

(a) Open area (b) Chokepoint

FIGURE 11: COMPARISON OF BEST -AND WORST CASE PERFORMANCE FOR CONVEX SAFE SET ESTIMATION ALGORITHM

5. CONCLUSION

Self-improving systems that can automatically learn from experience and optimize their performance are increasingly becoming a reality. In practice, it is challenging to guarantee safe operation due to the learning subsystems without significantly restricting their model class. In order to retain this flexibility, other systems that robustly guarantee safety are essential tools in designing control systems that incorporate learning components. In this work, the predictive safety filter proposed by [5] has been adapted to the domain of autonomous collision avoidance for ships. The filter activates when the control system proposes a potentially unsafe trajectory, and it computes a minimal adjustment of the input in order to satisfy the constraints. Anti-grounding is achieved via the computation of a convex safe set from cartographic data using the method proposed by [26]. In addition, static and dynamic obstacles are modeled as ellipses, and the corresponding distance functions are used as constraints when formulating the predictive safety filter.

The predictive safety filter was implemented using open-source software and is shown to be feasible for real-time applications (< 1 Hz), despite the nonlinear obstacle constraints. The performance was not optimized further in this work, but this can be achieved without significant effort by using the `acados` library [32] to compile the solver code or simply by using faster hardware. The limitations of the approach include (i) Sub-optimal behavior when the planning horizon is too short (ii) Conservative safe set estimation that performs poorly in narrow canals (iii) Static obstacles are modeled with constant velocities (iv) No handling of uncertainties (v) The safe trajectories do not follow traffic rules according to the COLREGS [31]. Points (ii-v) are of particular interest and will be the subject of future work. The flexibility and efficiency of the approach make it suitable for use with learning-based planning or control methods, which will be explored in future work.

ACKNOWLEDGEMENTS

This work is part of SFI AutoShip, an 8-year research-based innovation center. We want to thank our partners, including the Research Council of Norway, under project number 309230.

REFERENCES

[1] Heiberg, A., Larsen, T. N., Meyer, E. et al. "Risk-Based Implementation of COLREGs for Autonomous Surface Vehicles Using Deep Reinforcement Learning." *Neural Networks* Vol. 152 (2022): pp. 17–33. DOI 10.1016/j.neunet.2022.04.008.

[2] Tearle, B., Wabersich, K. P., Carron, A. et al. "A Predictive Safety Filter for Learning-Based Racing Control." *IEEE Robotics and Automation Letters* Vol. 6 (2021): pp. 7635–7642. DOI 10.1109/LRA.2021.3097073.

[3] Tengesdal, T., Johansen, T. A., Grande, T. D. et al. "Ship Collision Avoidance and Anti Grounding Using Parallelized Cost Evaluation in Probabilistic Scenario-Based Model Predictive Control." *IEEE Access* Vol. 10 (2022): pp. 111650–111664. DOI 10.1109/ACCESS.2022.3215654.

[4] Meyer, E., Robinson, H., Rasheed, A. et al. "Taming an Autonomous Surface Vehicle for Path Following and Collision

Avoidance Using Deep Reinforcement Learning." *IEEE Access* Vol. 8 (2020): pp. 41466–41481. DOI 10.1109/ACCESS.2020.2976586.

[5] Wabersich, K. P. and Zeilinger, M. N. "A Predictive Safety Filter for Learning-Based Control of Constrained Nonlinear Dynamical Systems." *Automatica* Vol. 129 (2021): p. 109597. DOI 10.1016/j.automatica.2021.109597.

[6] Huang, Y., Chen, L., Chen, P. et al. "Ship Collision Avoidance Methods: State-of-the-art." *Safety Science* Vol. 121 (2020): pp. 451–473. DOI 10.1016/j.ssci.2019.09.018.

[7] Vagale, A., Oucheikh, R., Bye, R. T. et al. "Path Planning and Collision Avoidance for Autonomous Surface Vehicles I: A Review." *Journal of Marine Science and Technology* DOI 10.1007/s00773-020-00787-6.

[8] Larson, J., Bruch, M., Halterman, R. et al. "Advances in Autonomous Obstacle Avoidance for Unmanned Surface Vehicles." Technical Report No. ADA475547. Space and Naval Warfare Systems Center, San Diego, CA. 2007. URL https://apps.dtic.mil/sti/citations/ADA475547.

[9] Loe, Øivind Aleksander G. "Collision Avoidance for Unmanned Surface Vehicles." Master's Thesis, Norwegian University of Science and Technology. 2008. URL http://hdl.handle.net/11250/259696.

[10] Casalino, G., Turetta, A. and Simetti, E. "A Three-Layered Architecture for Real Time Path Planning and Obstacle Avoidance for Surveillance USVs Operating in Harbour Fields." *OCEANS 2009-EUROPE*: pp. 1–8. 2009. DOI 10.1109/OCEANSE.2009.5278104.

[11] Eriksen, B. H., Bitar, G., Breivik, M. et al. "Hybrid Collision Avoidance for ASVs Compliant With COLREGs Rules 8 and 13-17." *Frontiers in Robotics and AI* Vol. 7. DOI 10.3389/frobt.2020.00011.

[12] Elfes, A. "Sonar-Based Real-World Mapping and Navigation." *IEEE Journal on Robotics and Automation* Vol. 3 No. 3 (1987): pp. 249–265. DOI 10.1109/JRA.1987.1087096.

[13] Candeloro, M., Lekkas, A. M. and Sørensen, A. J. "A Voronoi-diagram-based Dynamic Path-Planning System for Underactuated Marine Vessels." *Control Engineering Practice* Vol. 61 (2017): pp. 41–54. DOI 10.1016/j.conengprac.2017.01.007.

[14] Lavalle, S. M. "Rapidly-Exploring Random Trees: A New Tool for Path Planning." Technical. Department of Computer Science, Iowa State University. 1998. URL http://msl.cs.illinois.edu/~lavalle/papers/Lav98c.pdf.

[15] Kavraki, L., Svestka, P., Latombe, J. C. et al. "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces." *Robotics and Automation, IEEE Transactions on* Vol. 12 (1996): pp. 566–580. DOI 10.1109/70.508439.

[16] Tengesdal, T., Johansen, T. A. and Brekke, E. F. "Ship Collision Avoidance Utilizing the Cross-Entropy Method for Collision Risk Assessment." *IEEE Transactions on Intelligent Transportation Systems* Vol. 23 No. 8 (2022): pp. 11148–11161. DOI 10.1109/TITS.2021.3101007.

[17] Schuster, M., Blaich, M. and Reuter, J. "Collision Avoidance for Vessels Using a Low-Cost Radar Sensor." *Proceedings of the 19th IFAC World Congress, Cape Town, South Africa* Vol. 47 No. 3 (2014): pp. 9673–9678. DOI 10.3182/20140824-6-ZA-1003.01872.

[18] Chiang, H. L., Hsu, J., Fiser, M. et al. "RL-RRT: Kinodynamic Motion Planning via Learning Reachability Estimators From RL Policies." *IEEE Robotics and Automation Letters* Vol. 4 No. 4 (2019): pp. 4298–4305. DOI 10.1109/LRA.2019.2931199.

[19] Khatib, O. "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots." *IEEE International Conference on Robotics and Automation Proceedings*, Vol. 2: pp. 500–505. 1985. DOI 10.1109/ROBOT.1985.1087247.

[20] Lee, S., Kwon, K. and Joh, J. "A Fuzzy Logic for Autonomous Navigation of Marine Vehicles Satisfying COLREG Guidelines." *International Journal of Control, Automation, and Systems* Vol. 2 No. 2 (2004): pp. 171–181. URL https://koreascience.kr/article/JAKO200411922338133.page.

[21] Fiorini, P. and Shiller, Z. "Motion Planning in Dynamic Environments Using Velocity Obstacles." *The International Journal of Robotics Research* Vol. 17 No. 7 (1998): pp. 760–772. DOI 10.1177/027836499801700706.

[22] Kuwata, Y., Wolf, M. T., Zarzhitsky, D. et al. "Safe Maritime Autonomous Navigation With COLREGS, Using Velocity Obstacles." *IEEE Journal of Oceanic Engineering* Vol. 39 No. 1 (2014): pp. 110–119. DOI 10.1109/JOE.2013.2254214.

[23] Brekke, E. F., Eide, E., Eriksen, B. H. et al. "milliAmpere: An Autonomous Ferry Prototype." *Journal of Physics: Conference Series* Vol. 2311 No. 1 (2022): p. 012029. DOI 10.1088/1742-6596/2311/1/012029.

[24] Pedersen, A. A. "Optimization Based System Identification for the milliAmpere Ferry." Master's Thesis, NTNU. 2019. URL http://hdl.handle.net/11250/2625699.

[25] Fossen, T. I. *Handbook of Marine Craft Hydrodynamics and Motion Control*, 2nd ed. Wiley (2021). DOI 10.1002/9781119994138.

[26] Bitar, G., Martinsen, A. B., Lekkas, A. M. et al. "Two-Stage Optimized Trajectory Planning for ASVs Under Polygonal Obstacle Constraints: Theory and Experiments." *IEEE Access* Vol. 8 (2020): pp. 199953–199969. DOI 10.1109/ACCESS.2020.3035256.

[27] Gillies, S., van der Wel, C., Van den Bossche, J. et al. "Shapely." Github repository (2022). URL https://github.com/shapely/shapely.

[28] Andersson, J. A. E., Gillis, J., Horn, G. et al. "CasADi: A Software Framework for Nonlinear Optimization and Optimal Control." *Mathematical Programming Computation* Vol. 11 No. 1 (2019): pp. 1–36. DOI 10.1007/s12532-018-0139-4.

[29] Wächter, A. and Biegler, L. T. "On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming." *Mathematical Programming* Vol. 106 No. 1 (2006): pp. 25–57. DOI 10.1007/s10107-004-0559-y.

[30] Blindheim, S. and Johansen, T. A. "Electronic Navigational Charts for Visualization, Simulation, and Autonomous Ship

Control." *IEEE Access* Vol. 10 (2022): pp. 3716–3737. DOI 10.1109/ACCESS.2021.3139767.

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

[32] Verschueren, R., Frison, G., Kouzoupis, D. et al. "Acados - a Modular Open-Source Framework for Fast Embedded Optimal Control." *Mathematical Programming Computation* Vol. 14 (2021): pp. 147–183. DOI 10.1007/s12532-021-00208-8.