IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 30, 2020, accepted November 8, 2020, date of publication November 11, 2020,
date of current version November 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3037171

Optimization-Based Automatic Docking and Berthing of ASVs Using Exteroceptive Sensors: Theory and Experiments

ANDREAS B. MARTINSEN[1], GLENN BITAR[1,2], ANASTASIOS M. LEKKAS[1,2], AND SÉBASTIEN GROS[1]

[1]Department of Engineering Cybernetics, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway
[2]Centre for Autonomous Marine Operations and Systems, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway

Corresponding author: Andreas B. Martinsen (andreas.b.martinsen@ntnu.no)

This work was supported in part by the Research Council of Norway under Project 269116, and in part by the Centers of Excellence funding Scheme under Project 223254.

ABSTRACT Docking of autonomous surface vehicles (ASVs) involves intricate maneuvering at low speeds under the influence of unknown environmental forces, and is often a challenging operation even for experienced helmsmen. In this paper, we propose an optimization-based trajectory planner for performing automatic docking of a small ASV. The approach formulates the docking objective as a nonlinear optimal control problem, which is used to plan collision-free trajectories. Compared to recent works, the main contributions are the inclusion of a map of the harbor and additional measurements from range sensors, such as LIDAR and ultrasonic distance sensors, to account for map inaccuracies as well as unmapped objects, such as moored vessels. To use the map and sensor data, a set generation method is developed, which in real-time computes a safe operating region, this is then used to ensure the planned trajectory is safe. To track the planned trajectory, a trajectory-tracking dynamic positioning controller is used. The performance of the method is tested experimentally on a small ASV in confined waters in Trondheim, Norway. The experiments demonstrate that the proposed method is able to perform collision-free docking maneuvers with respect to static obstacles, and achieves successful docking.

INDEX TERMS Autonomous surface vehicles, berthing, collision avoidance, docking, marine vehicles, motion planning, optimal control, trajectory optimization.

I. INTRODUCTION

Autonomy, and autonomous systems is a rapidly growing area of interest in a wide variety of industries. This includes the maritime industry, where autonomous surface vehicles (ASVs) have been proposed for surveying and mapping, surveillance, and transportation, to name a few application areas. With the motivation factors including lower costs, higher availability and flexibility, better safety and reliability, and reduced environmental impact. In the case of transportation, autonomous operations can be roughly split into the following three phases.

- Undocking – moving from the quay in a confined harbor area to open waters,
- transit – crossing a body of water towards the destination harbor,
- docking – moving from open waters towards the docking position along the quay in a harbor area.

In this paper we will focus on docking of a vessel in a confined harbor area. While the method we propose in this paper is able to solve both the docking and undocking problem, the focus is on docking, as it is the most challenging of the two and requires very precise movements [1] when performing the final controlled collision with the quay.

The problem of automatic docking and berthing is an important part of performing autonomous transportation, and hence the problem has seen a lot of interest, with a variety of solutions. However, due to the complexity of performing docking, most of the existing methods rely on simplifying the docking problem, this has lead to a lack of experimental results. The traditional approach for docking large under-actuated vessels, requires the use of tug boats, as support vessels, in order to push and pull the vessel to

The associate editor coordinating the review of this manuscript and approving it for publication was Xiwang Dong.

perform the docking maneuver. This has lead to research into synchronizing the movement of multiple tugboats, in order to perform the desired maneuvers [2]–[5]. With many newer vessels being fully actuated, or even over-actuated, research has shifted to seeking methods for automatically performing docking without the use of additional support vessels. One such approach to solving the docking problem is via fuzzy control. The control system is then based on fuzzy logic, and its behaviour changes based on a set of predetermined rules [6]–[8]. An other approach for docking, that has seen a lot of interest, is the use of Artificial Neural Networks (ANNs) [2], [5], [9]–[15]. For these approaches, an ANN is used as a function approximator for the policy, and is tasked with learning to imitate pre-recorded docking trajectories, and hence learning how to perform the docking maneuvers. More recent learning-based methods have expanded on this by using advances in Deep Learning (DL) [16]–[18]. Additional approaches include docking using a rule-based expert system [19], docking by target tracking [20], and docking using artificial potential fields [21]. Within industry, several companies have developed methods for automatic docking [22]–[24], however details about the different approaches remain sparse. The most promising approaches however, rely on optimization-based planning [25]–[34], where trajectories are planned using convex optimization. These methods are often preferable, as they allow for explicitly including dynamics and constraints when planning a trajectory.

When developing automatic docking systems to facilitate berthing of vessels, it is important to have accurate and reliable positioning systems in place. This is required in order accurately determine the position of the vessel hull relative to the berth. While this is possible to do using satellite positioning systems, it requires high precision satellite positioning and the position of the berth must be well known, which may not always be the case. In order to overcome these problems, the use of quay-mounted laser or radar ranging systems [35]–[37] is often used in larger ports, in order to independently identify the position and velocity of the vessel relative to the quay.

The docking method from [32] is a nonlinear model predictive controller (NMPC) that takes into account vessel dynamics in the form of its dynamic model, as well as collision avoidance by planning trajectories within a convex set, based on the harbor layout. Advantages of this approach include the explicit handling of static obstacles, the planning of dynamically feasible trajectories, and a flexible behavior shaping via the nonlinear cost function. The method does not handle moving obstacles or account for external unknown disturbances. Additionally, due to the non-convexity of the optimal control problem, guarantees on run time or feasibility are not provided. In this paper, we build on [32], [33] and propose a novel algorithm for dynamically creating a convex safety set, based on a map of the environment. We also show how this method can be combined with sensor data from on board sensors such as LIDAR pointclouds, and ultrasonic distance sensors, in order to account for missing or inaccurate map data. This allows to plan and perform docking maneuvers in harbors without the need for land-based sensor systems, even if the harbor layout changes. We also propose some modifications to the cost function, in order to generate more efficient docking trajectories. Finally, we validate the method in full-scale experiments on the experimental autonomous urban passenger ferry *milliAmpere*, seen in Figure 1, and show how the proposed approach is able to successfully plan and perform safe and collision free docking maneuvers in confined waters in Trondheim, Norway. The contributions of this work can be split into two main categories, namely methodology, and implementation.

FIGURE 1. Experimental platform milliAmpere.

1) Methodology builds on the work in [32], with the following improvements:
   - A set generation method for identifying a safe operating region in real-time (Section IV-A).
   - Improvements to the cost function, which give more refined docking maneuvers (Section III-B).
2) Implementation builds on the work in [33], with the following improvements:
   - Addition of exteroceptive sensor data to account for map inaccuracies as well as unmapped objects (Section IV-B).
   - Improved interplay between the tracking controller and planner for better tracking performance (Section V-A).
   - Improvements to the cost function, which gives better tracking performance (Section III-B).

To the authors' best knowledge, this is the first work to demonstrate fully automatic docking using only on-board sensors and map data, and use this data to plan a safe and feasible trajectory in real-time.

II. VESSEL MODEL

This section presents the kinematic, dynamic and thruster models of an ASV, which have to be taken into account when planning a safe and feasible docking trajectory.

A. KINEMATICS AND DYNAMICS

When modeling vessels for the purpose of autonomous docking, we consider only the vessel movement on the ocean

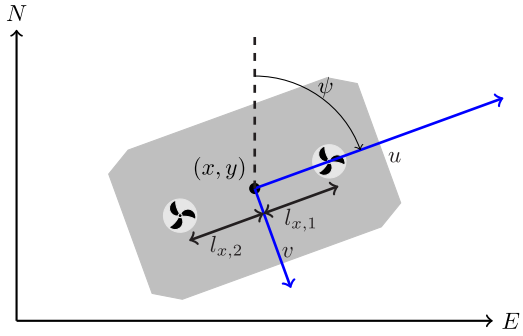


FIGURE 2. 3-DOF vessel centered at $p_c = [x, y]^\top$, with surge velocity u, sway velocity v, heading ψ in a North-East-Down (NED) reference frame.

surface, neglecting the roll, pitch and heave motions. The mathematical model used to describe the system can then be kept reasonably simple as it is limited to the planar position and orientation of the vessel. Given $\mathbb{R}$ as the set of real numbers, $\mathbb{S} = [0, 2\pi]$ as the set of angles, and $SO(n) = \{\boldsymbol{R} | \boldsymbol{R} \in \mathbb{R}^{n\times n}, \boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^\top = \boldsymbol{I}, \det(\boldsymbol{R}) = 1\}$ as the special orthogonal group in n dimensions, the motion of a surface vessel can be represented by the pose vector $\boldsymbol{\eta} = [x, y, \psi]^\top \in \mathbb{R}^2 \times \mathbb{S}$, and velocity vector $\boldsymbol{\nu} = [u, v, r]^\top \in \mathbb{R}^3$. Here, $\boldsymbol{p}_c = [x, y]^\top$ describe the Cartesian position in the Earth-fixed reference frame, ψ is yaw angle, (u, v) is the body fixed linear velocities, and r is the yaw rate, an illustration is given in Figure 2. Using the notation in [38] we can describe a 3-DOF vessel model as follows

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

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}, \tag{2}$$

where $\boldsymbol{M} \in \mathbb{R}^{3\times3}$, $\boldsymbol{C}(\boldsymbol{\nu}) \in \mathbb{R}^{3\times3}$, $\boldsymbol{D}(\boldsymbol{\nu}) \in \mathbb{R}^{3\times3}$, $\boldsymbol{\tau} \in \mathbb{R}^3$ and $\boldsymbol{J}(\psi) \in SO(3)$ are the inertia matrix, Coriolis matrix, dampening matrix, control forces and moments, and transformation matrix, respectively. The transformation matrix $\boldsymbol{J}(\psi)$ is given by

$$\boldsymbol{J}(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

and represent the rotation from the body frame to the Earth-fixed reference frame. For detailed information about the *milliAmpere* model parameters used in this paper, the reader is referred to [33].

B. THRUST CONFIGURATION

The control surfaces of the vessel are specified by the thrust configuration matrix $\boldsymbol{T} \in \mathbb{R}^{3,n_{\text{thrusters}}}$, which maps the thrust $\boldsymbol{f} \in \mathbb{R}^{n_{\text{thrusters}}}$ from each thruster into the surge, sway and yaw forces and moments in the body frame of the vessel

$$\boldsymbol{\tau} = \boldsymbol{T}\boldsymbol{f}. \tag{4}$$

Each column $\boldsymbol{T}_i$ in $\boldsymbol{T}$ gives the configuration of the forces and moments of a thruster i as follows:

$$\boldsymbol{T}_i f_i = \begin{bmatrix} F_{x,i} \\ F_{y,i} \\ F_{y,i} \cdot l_{x,i} - F_{x,i} \cdot l_{y,i} \end{bmatrix}, \tag{5}$$

where $F_{x,i}$ and $F_{y,i}$ are the forces in the body frame, and $l_{x,i}$ and $l_{y,i}$ is the position of the thruster in the body frame. Given a desired force vector $\boldsymbol{\tau}$, finding the individual thruster forces that produce it, is called the thrust allocation problem. While there are numerous ways of solving the thrust allocation problem [39], in order to account for the thrust configuration and individual thruster constraints, we want to include the thrust allocation as part of the optimization and planning for performing the docking operations. For the *milliAmpere* vessel, illustrated in Figure 2, there are two thrusters mounted along the center line of the vessel, giving the following control force and moments:

$$\boldsymbol{\tau} = \begin{bmatrix} F_{x,1} \\ F_{y,1} \\ F_{y,1} \cdot l_{x,1} \end{bmatrix} + \begin{bmatrix} F_{x,2} \\ F_{y,2} \\ F_{y,2} \cdot l_{x,2} \end{bmatrix}. \tag{6}$$

III. TRAJECTORY PLANNING AND CONTROL

Automatic docking is a complex problem, which includes planning and performing maneuvers to control a vessel to a desired orientation and position, while adhering to spatial constraints in order to avoid collisions. In order to perform the docking, we expand upon [32], [33], where we use a docking trajectory planner, constructed as an Optimal Control Problem (OCP). This allows the planner to take into account the vessel dynamics in terms of a mathematical model, as well as the harbor layout in terms of a map of landmasses. Additionally we include the use of ranging sensors, namely ultrasonic distance sensors and LIDAR, in order to account for obstacles not present in the map of the harbor layout.

A. OBSTACLE AVOIDANCE

Given a desired position x_d, y_d and a desired heading ψ_d, we define the docking problem as maneuvering a vessel as close as possible to the desired docking pose $\boldsymbol{\eta}_d = [x_d, y_d, \psi_d]^\top$, without running the vessel into obstacles, i.e. adhering to spatial constraints. As proposed in [32], the safe operation of the vessel can be formalised in terms of the vessel boundary $\mathbb{S}_v$ being contained within a convex inner approximation of the surrounding obstacles $\mathbb{S}_s$, called the spatial constraints, see Figure 3. Since the safe operating region, given in terms of the spatial constraints $\mathbb{S}_s$, is given as a convex set, the region can be defined in terms of a set of linear inequality constraints:

$$\mathbb{S}_s = \{\boldsymbol{p} \in \mathbb{R}^2 \mid \boldsymbol{A}_s\boldsymbol{p} \leq \boldsymbol{b}_s\},$$

where the matrix $\boldsymbol{A}_s \in \mathbb{R}^{n_{\text{constraints}},2}$ and vector $\boldsymbol{b}_s \in \mathbb{R}^{n_{\text{constraints}}}$ define the linear inequality constraints. Furthermore, we can note that if both the vessel set $\mathbb{S}_v$ and spatial constraint $\mathbb{S}_s$ are convex, then the vessel is contained within the spatial constraints so long as all the vertices of the vessel boundary are contained within the spatial constraints. This is useful as it allows us to simplify the safety constraints to the following:

$$\mathbb{S}_v \subseteq \mathbb{S}_s \iff \boldsymbol{A}_s\boldsymbol{p}_i^n \leq \boldsymbol{b}_s \quad \forall \boldsymbol{p}_i^n \in \text{Vertex}(\mathbb{S}_v), \tag{7}$$

where $\boldsymbol{p}_i^n$ denotes the position of vertex i in the North-East-Down (NED) reference frame. Since the vertices of the

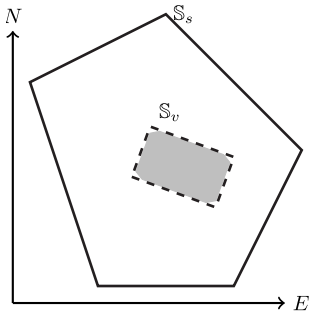


FIGURE 3. The gray convex polytope illustrates the vessel, whereas $\mathbb{S}_v$ is the vessel boundary, and $\mathbb{S}_s$ are the spatial constraints. The vessel will always lie within the spatial constraints $\mathbb{S}_s$ as long as all the vertices of $\mathbb{S}_v$ lie within the spatial constraints.

vessel boundary are given in the body frame of the vessel, we need to transform them from the body frame to the NED frame, giving the following nonlinear constraints:

$$\boldsymbol{A}_s \left(\boldsymbol{R}(\psi) \boldsymbol{p}_i^b + \boldsymbol{p}_c \right) \leq \boldsymbol{b}_s \quad \forall \boldsymbol{p}_i^b \in \text{Vertex}(\mathbb{S}_v), \tag{8}$$

where $\boldsymbol{p}_i^b$ denotes the position of vertex i in the body frame, $\boldsymbol{p}_c = [x, y]^\top$ is the vessel position and $\boldsymbol{R}$ is the rotation from the body frame to NED.

$$\boldsymbol{R}(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \tag{9}$$

The constraints in (8) can be directly implemented as inequality constraints in an optimization problem, and ensures the vessel is contained within a safe region given by the spatial constraints.

While this constraint is easily implemented in a nonlinear programming (NLP) problem, the constraint is not convex. As a result, a feasible solution of the problem is guaranteed to satisfy the spatial constraints, but we cannot guarantee that the NLP will converge to a global optimum.

B. OPTIMAL CONTROL PROBLEM

In order to plan a safe trajectory for the vessel, we formulate the problem as the following continuous time nonlinear optimal control problem:

$$\min_{\boldsymbol{x}_p(\cdot), \boldsymbol{u}_p(\cdot), \boldsymbol{s}(\cdot)} \int_{t_0}^{t_0+T} \left(l(\boldsymbol{x}_p(t), \boldsymbol{u}_p(t)) + \boldsymbol{k}_s^\top \boldsymbol{s}(t) \right) dt \tag{10a}$$

$$\text{s.t. } \dot{\boldsymbol{x}}_p(t) = \boldsymbol{f}(\boldsymbol{x}_p(t), \boldsymbol{u}_p(t)) \quad \forall t \in [t_0, t_0 + T] \tag{10b}$$

$$\boldsymbol{h}(\boldsymbol{x}_p(t), \boldsymbol{u}_p(t)) - \boldsymbol{s}(t) \leq \boldsymbol{0} \quad \forall t \in [t_0, t_0 + T] \tag{10c}$$

$$\boldsymbol{s}(t) \geq \boldsymbol{0} \quad \forall t \in [t_0, t_0 + T] \tag{10d}$$

$$\boldsymbol{x}_p(t_0) = \boldsymbol{x}(t_0), \tag{10e}$$

where $\boldsymbol{x}_p(t) = [\boldsymbol{\eta}_p, \boldsymbol{\nu}_p]^\top$ is the planned state trajectory of the vessel, $\boldsymbol{u}(t) = [F_{x,1}, F_{y,1}, F_{x,2}, F_{y,1}]^\top$, are the planned thruster forces, and $\boldsymbol{s}(t)$ are slack variables. The constraint relaxation (10c), is added in order to allow the planner to plan a trajectory from a possibly infeasible initial pose. This is useful, as it allows for re-planning the docking trajectory in a model predictive control (MPC) like fashion, and use low-level controllers to follow the planned trajectory.

The cost function (10a) includes a slack variable cost, and a stage cost. For the slack variable cost $\boldsymbol{k}_s^\top \boldsymbol{s}(t)$, the gain $\boldsymbol{k}_s$ is chosen large enough such that the slack variables are active only when the non-relaxed constraints are infeasible. The following stage cost was chosen:

$$\begin{aligned} l(\boldsymbol{x}_p(t), \boldsymbol{u}_p(t)) = q_{x,y} \cdot c_{x,y}(\boldsymbol{\eta}_p(t)) + q_\psi \cdot c_\psi(\boldsymbol{\eta}_p(t)) \\ + \boldsymbol{\nu}_p(t)^\top \boldsymbol{Q} \boldsymbol{\nu}_p(t) + \boldsymbol{u}_p(t)^\top \boldsymbol{R} \boldsymbol{u}_p(t). \end{aligned} \tag{11}$$

The velocity and control actions are penalized with a quadratic penalty, with weight matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$. The position cost $c_{x,y}(\boldsymbol{\eta}_p(t))$ is chosen as a pseudo-Huber function, penalizing the difference between the current pose and the docking pose $\boldsymbol{\eta}_d$, and is given as follows:

$$c_{x,y}(\boldsymbol{\eta}_p) = \delta^2 \left(\sqrt{1 + \frac{(x_p(t) - x_d)^2 + (y_p(t) - y_d)^2}{\delta^2}} - 1 \right). \tag{12}$$

Using a pseudo-Huber cost, provides a quadratic penalty when the quadrature position error is low and linear when the position error is high. This helps with numerical stability, as well as performance when large position errors are observed [40], [41]. For the heading cost function $c_\psi(\boldsymbol{\eta}_p(t))$, the following was chosen:

$$c_\psi(\boldsymbol{\eta}_p(t)) = \frac{1 - \cos(\psi_p(t) - \psi_d))}{2} e^{-\frac{(x_p(t) - x_d)^2 + (y_p(t) - y_d)^2}{2\delta^2}}. \tag{13}$$

The first factor of the heading cost function is the cost of the heading error, in a form that avoids the wraparound of the heading. The second part is a Gaussian function, which discounts the heading error when far away from the docking pose. This cost function has the effect of having the planner chose an efficient heading when far away from the dock, and then gradually rotate the vessel towards the desired heading when closing in on the dock. It is worth noting that the cost works similarly to a terminal cost, but is phased in more gradually than a genuine terminal cost. This has the benefit of making the OCP less sensitive to the length of the prediction horizon.

The first constraint (10b) ensures that the planned trajectory satisfies the kinematic and dynamic models of the vessel described by (1) and (2). The inequality constraint (10c) consists of the spatial constraints described in (8), as well as constraints on the maximum and minimum velocity, angular velocity, and thruster forces. The constraint in (10d) ensures that the slack variables are positive, and (10e) sets the initial state to that of the vessel.

In order to implement the continuous time problem given in (10) we need to transcribe the problem into the standard form.

$$\begin{aligned} &\min_{\boldsymbol{w}} \ \phi(\boldsymbol{w}) \\ &\text{s.t. } \boldsymbol{g}(\boldsymbol{w}) = 0 \\ &\quad \boldsymbol{h}(\boldsymbol{w}) \leq 0 \end{aligned}$$

This can be done in multiple ways, however the two main classes of methods are *sequential methods*, such as direct single shooting [42], and *simultaneous methods* such as direct multiple shooting [43], and direct collocation [44]. For this approach we chose to use direct collocation, in where implicit numerical integration of the ordinary differential equation (ODE) constraints (10b), as well as the objective function (10a), is performed as part of the nonlinear optimization. For the implementation of the docking planner, we defined a planning horizon of $T = 120s$, with $N = 60$ shooting intervals, and degree $d = 3$ Legendre polynomials. This was chosen as it gave a good trade-off in terms of horizon length, integration accuracy and computational complexity.

IV. AUTOMATIC CONSTRAINT GENERATION

Given the OCP formulation in the previous section, we are faced with the problem of finding a convex set:

$$\mathbb{S}_s = \{\boldsymbol{p} \in \mathbb{R}^2 \mid \boldsymbol{A}_s\boldsymbol{p} \leq \boldsymbol{b}_s\},$$

within which the vessel must be contained. The set $\mathbb{S}_s$ must be created in a way such that it does not intersect with any constraints stemming from the environmental obstacles. Ideally we would also like the set to be as large as possible, in order to not unnecessarily restrict the vessel movement. While a number of methods for performing constraint generation already exists [45]–[48], they can often be computationally expensive. In this section, we propose a method similar to [45], [46], where a constraint set is generated as a vessel-centered convex inner approximation of the environmental constraints. We will also show how we can easily and efficiently compute this set from known map data, as well as from range sensor such as LIDAR and ultrasonic distance sensors.

A. COMPUTING A CONVEX INNER APPROXIMATION

In this section, we show how to compute the safe set as a vessel-centered convex inner approximation of the environmental constraints, when the obstacles are represented by line segments making up obstacle polygons. Intuitively the method can be summarized as follows:

1) Given a center point $\boldsymbol{p}_c$, grow an ellipse centered at $\boldsymbol{p}_c$, until it touches an environmental constraint (Section IV-A1).
2) Create a linear constraint tangent to the expansion ellipse at the contact point between the ellipse and the environmental constraint (Section IV-A2).
3) Continue growing and creating linear constraints until no further growth is possible, and combine the linear constraints into the final convex inner approximation of the environment (Section IV-A3)

In the next subsections we will show how to quickly and efficiently perform the steps above in order to end up with a simple closed form solution to generating the convex inner approximation.

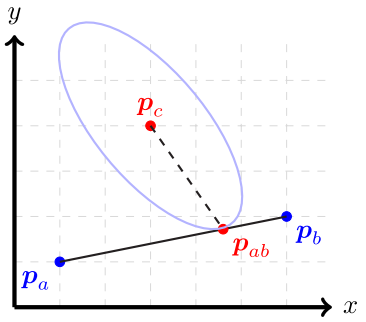


FIGURE 4. The contact point p_{ab} is given by the shortest non-Euclidean distance from point p_c to line segment (p_a, p_b).

1) COMPUTING THE CONTACT POINT BETWEEN ELLIPSE AND ENVIRONMENTAL CONSTRAINT

Given a line segment $(\boldsymbol{p}_a, \boldsymbol{p}_b)$ making up the environmental constraints, we want to find the contact point $\boldsymbol{p}_{ab}$ between the line segment and the expansion ellipse centered at $\boldsymbol{p}_c$, this is illustrated in Figure 4. We can formulate this as an optimization problem which minimizes the distance between $\boldsymbol{p}_c$ and the parametric line segment:

$$\boldsymbol{p}_{ab}(\omega) = \boldsymbol{p}_a(1-\omega) + \boldsymbol{p}_b\omega, \tag{14}$$

where $\omega \in [0, 1]$. For the optimization problem we wish to find the parameter ω that minimizes a non-Euclidean distance from $\boldsymbol{p}_{ab}(\omega)$ to $\boldsymbol{p}_c$, giving the following:

$$\min_{\omega} f(\omega) = \left(\boldsymbol{p}_{ab}(\omega) - \boldsymbol{p}_c\right)^\top \boldsymbol{\Sigma} \left(\boldsymbol{p}_{ab}(\omega) - \boldsymbol{p}_c\right) \tag{15}$$

$$s.t.\ 0 \leq \omega \leq 1, \tag{16}$$

where $\boldsymbol{\Sigma}$ is a positive definite symmetric projection matrix, defining the expansion ellipse. Choosing $\boldsymbol{\Sigma} = \boldsymbol{I}$ gives the Euclidean distance, while choosing $\boldsymbol{\Sigma}$ as a different positive definite symmetric matrix, allows prioritizing a direction, when minimizing the distance. For the unconstrained optimization problem, the necessary conditions give:

$$\begin{aligned}\frac{\mathrm{d}f(\omega)}{\mathrm{d}\omega} &= (\boldsymbol{p}_a(1-\omega) + \boldsymbol{p}_b\omega - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma}(\boldsymbol{p}_b - \boldsymbol{p}_a) = 0 \\ &\Rightarrow \omega = -\frac{(\boldsymbol{p}_a - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma}(\boldsymbol{p}_b - \boldsymbol{p}_a)}{(\boldsymbol{p}_b - \boldsymbol{p}_a)^\top \boldsymbol{\Sigma}(\boldsymbol{p}_b - \boldsymbol{p}_a)}.\end{aligned} \tag{17}$$

This gives the parameterized variable ω for the unconstrained problem. The constrained $\omega \in [0, 1]$, due to the simplicity of the constraints and convexity of the optimization problem, can be found by simply clipping ω between 0 and 1:

$$\omega = \mathrm{clip}\left(-\frac{(\boldsymbol{p}_a - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma}(\boldsymbol{p}_b - \boldsymbol{p}_a)}{(\boldsymbol{p}_b - \boldsymbol{p}_a)^\top \boldsymbol{\Sigma}(\boldsymbol{p}_b - \boldsymbol{p}_a)}, 0, 1\right) \tag{18}$$

The closest point $\boldsymbol{p}_{ab}$, constrained to being on the line segment $(\boldsymbol{p}_a, \boldsymbol{p}_b)$ is then given by inserting the parameter w from (18) into (14). We should note that this gives a closed form solution for the contact point $\boldsymbol{p}_{ab}$, which means it can be efficiently computed.

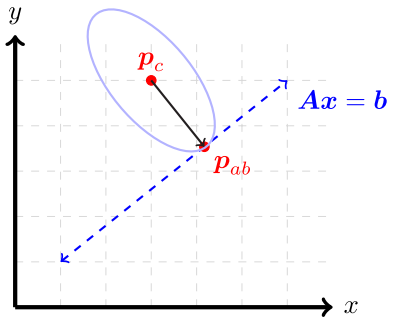


FIGURE 5. Finding the line $Ax = b$ tangent to the expansion ellipse at p_{ab}, i.e. normal line to the vector $\Sigma(p_{ab} - p_c)$ passing through p_{ab}.

2) FINDING THE TANGENT LINE TO THE EXPANSION ELLIPSE

Given the closest point $\boldsymbol{p}_{ab}$ on a line segment, the next step is to find the line $\boldsymbol{Ax} = \boldsymbol{b}$ which is tangent to the expansion ellipse, and hence normal to the ellipse gradient $\boldsymbol{\Sigma}(\boldsymbol{p}_{ab} - \boldsymbol{p}_c)$, as illustrated in Figure 5. Given the expansion ellipse gradient for a point $\boldsymbol{p}_{ab}$ as:

$$\boldsymbol{\Sigma}(\boldsymbol{p}_{ab} - \boldsymbol{p}_c) \tag{19}$$

we can note that any nonzero vector $[x, y]^\top$ is orthogonal to the expansion ellipse, if the inner product is zero.

$$(\boldsymbol{p}_{ab} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \begin{bmatrix} x \\ y \end{bmatrix} = 0 \;\Rightarrow \text{Orthogonal.} \tag{20}$$

Using (20), the tangent line of the expansion ellipse, passing through the point $[x_0, y_0]^\top$ is given as the following.

$$(\boldsymbol{p}_{ab} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \left(\begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \right) = 0 \tag{21}$$

Since we want the tangent line of the expansion ellipse to pass through the point $\boldsymbol{p}_{ab}$, the tangent line is given by all points $[x, y]^\top$ which fulfill the following equality.

$$\underbrace{(\boldsymbol{p}_{ab} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}}_{\boldsymbol{x}} = \underbrace{(\boldsymbol{p}_{ab} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \boldsymbol{p}_{ab}}_{\boldsymbol{b}} \tag{22}$$

3) LINEAR INEQUALITY CONSTRAINT GENERATION

In order to generate the convex constraints around a point $\boldsymbol{p}_c$, our proposed method is based on finding the tangent line to the expansion ellipse (see Section IV-A2) from the point $\boldsymbol{p}_c$ to the closest point $\boldsymbol{p}_{ab,i}$ (see Section IV-A1) on each line segment $i \in 1, 2, \ldots N$, making up environmental constraints. By stacking the tangent lines we get a half-space representation of the convex inner approximation as the following linear inequality constraints.

$$\underbrace{\begin{bmatrix} (\boldsymbol{p}_{ab,1} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \\ (\boldsymbol{p}_{ab,2} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \\ \vdots \\ (\boldsymbol{p}_{ab,N} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \end{bmatrix}}_{\boldsymbol{A}} \boldsymbol{p} \leq \underbrace{\begin{bmatrix} (\boldsymbol{p}_{ab,1} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \boldsymbol{p}_{ab,1} \\ (\boldsymbol{p}_{ab,2} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \boldsymbol{p}_{ab,2} \\ \vdots \\ (\boldsymbol{p}_{ab,N} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \boldsymbol{p}_{ab,N} \end{bmatrix}}_{\boldsymbol{b}} \tag{23}$$

We can note that since (18) is piecewise linear and smooth, the constraints given above are continuous with respect to the center point $\boldsymbol{p}_c$. This is a useful property, as this means that the shape of the convex inner approximation will change continuously with the center point $\boldsymbol{p}_c$.

While (23) gives a set of linear inequalities that may be used directly in an OCP, it is worth noting that the rows of the constraint may contain a large variance in magnitude. Since the inequalities are linear however, we can multiply each row of the inequalities by a normalizing factor. One such convenient normalizing factor is:

$$\frac{1}{||(\boldsymbol{p}_{ab,i} - \boldsymbol{p}_c)\boldsymbol{\Sigma}||_2} \tag{24}$$

The reason this normalizing factor is convenient, is the fact that the resulting matrix $\boldsymbol{A}$ becomes dimensionless with every row having unit length, and hence the the constraint will have the same units as $\boldsymbol{p}$. This has several benefits, including that $\boldsymbol{Ap} - \boldsymbol{b}$ will have a physical meaning in terms of distance until the constraint is reached. This allows for adding a back-off or margin in order to shrink the constraints, and make them more conservative. An other benefit is when using a slack variabels in order to relax the constraints in the OCP, the slack variable will have a physical meaning. Using this normalization factor we get the linear inequality constraints given below.

$$\underbrace{\begin{bmatrix} \frac{(\boldsymbol{p}_{ab,1} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma}}{||(\boldsymbol{p}_{ab,1} - \boldsymbol{p}_c)\boldsymbol{\Sigma}||_2} \\ \frac{(\boldsymbol{p}_{ab,2} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma}}{||(\boldsymbol{p}_{ab,2} - \boldsymbol{p}_c)\boldsymbol{\Sigma}||_2} \\ \vdots \\ \frac{(\boldsymbol{p}_{ab,N} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma}}{||(\boldsymbol{p}_{ab,N} - \boldsymbol{p}_c)\boldsymbol{\Sigma}||_2} \end{bmatrix}}_{\boldsymbol{A}} \boldsymbol{p} \leq \underbrace{\begin{bmatrix} \frac{(\boldsymbol{p}_{ab,1} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \boldsymbol{p}_{ab,1}}{||(\boldsymbol{p}_{ab,1} - \boldsymbol{p}_c)\boldsymbol{\Sigma}||_2} \\ \frac{(\boldsymbol{p}_{ab,2} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \boldsymbol{p}_{ab,2}}{||(\boldsymbol{p}_{ab,2} - \boldsymbol{p}_c)\boldsymbol{\Sigma}||_2} \\ \vdots \\ \frac{(\boldsymbol{p}_{ab,N} - \boldsymbol{p}_c)^\top \boldsymbol{\Sigma} \boldsymbol{p}_{ab,N}}{||(\boldsymbol{p}_{ab,N} - \boldsymbol{p}_c)\boldsymbol{\Sigma}||_2} \end{bmatrix}}_{\boldsymbol{b}} \tag{25}$$

4) REMOVING REDUNDANT CONSTRAINTS

The inequality constraints in (25) can be further reduced to $M \leq N$ constraints, by removing redundant constraints (Figure 6). Given a system of linear inequalities,

$$\boldsymbol{Ap} \leq \boldsymbol{b} \tag{26}$$

a given row $\boldsymbol{A}_k, \boldsymbol{b}_k$ can be identified as being a redundant constraint by solving the following Linear Programming (LP) problem:

$$\begin{aligned} &\min_{\boldsymbol{p}} \; y_k = \boldsymbol{b}_k - \boldsymbol{A}_k \boldsymbol{x} \\ &\text{s.t. } \boldsymbol{A}_i \boldsymbol{p} \leq \boldsymbol{b}_i \quad \forall i \neq k, \end{aligned} \tag{27}$$

and checking if the above problem has a solution $\boldsymbol{p}$ for which $y_k \geq 0$ [49]. For large numbers of constraints, solving an LP to check if each constraint is redundant is however very inefficient. Fortunately, a number of other approaches exists, [50]. For our approach, we used the qhull library [51], which efficiently perform constraint reduction for large numbers of

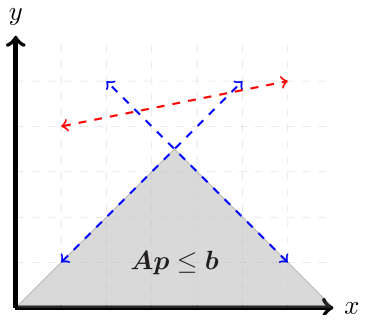
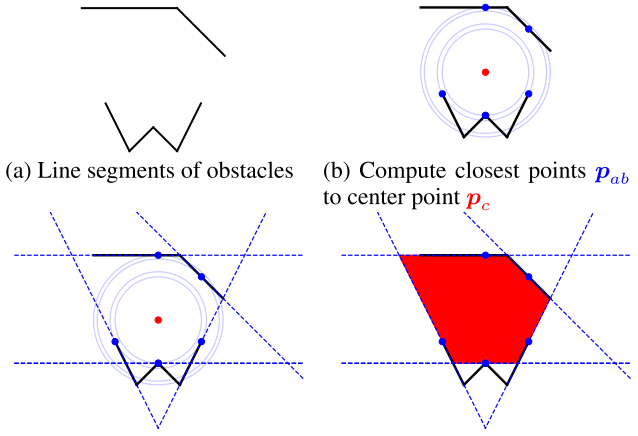


FIGURE 6. Redundant constraints are constraints that don't make up the support of the intersecting half-plane $\{p \in \mathbb{R}^2 \mid Ap \leq b\}$.



(a) Line segments of obstacles

(b) Compute closest points $\boldsymbol{p}_{ab}$ to center point $\boldsymbol{p}_c$

(c) Find tangent lines to the expansion ellipse at each point $\boldsymbol{p}_{ab}$

(d) Final convex inner approximation

FIGURE 7. Illustration of how to compute the convex spatial constraints.

constraints. It does this by first computing the dual points of the constraints as:

$$\boldsymbol{d}_i = \frac{\boldsymbol{A}_i}{\boldsymbol{b}_i - \boldsymbol{A}_i \boldsymbol{p}}, \tag{28}$$

where $\boldsymbol{p}$ is an interior point of the constraints. Then computes the convex hull of the dual points as:

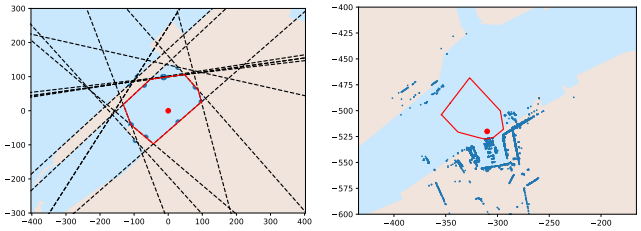
$$\mathcal{D} = \text{Conv}(\{\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots \boldsymbol{d}_N\}). \tag{29}$$

The support of the constraints are then given by the rows i for which the dual points $\boldsymbol{d}_i$ are extreme points of the convex hull $\mathcal{D}$, and redundant for dual points which are not extreme points of $\mathcal{D}$.

Given a set of line segments making up the boundary of obstacles, we can use the method above to first compute the closest points to all obstacles, then find the tangent line to the expansion ellipse, and generate the final constraint as a set of linear inequality constraints. This procedure is illustrated in Figure 7.

B. COMPUTING SPATIAL CONSTRAINTS FROM MAP AND SENSOR DATA

In order to compute the spatial constraints for the docking problem, it is possible to use a known map of the



(a) Set generation from map data, where blue dots are the Closest points $\boldsymbol{p}_{ab}$ on the line segments of the land masses.

(b) Set generation from map and LIDAR data, where blue dots represent the point cloud from the LIDAR sensor.

FIGURE 8. Set generation is performed by growing a region (red polygon) of water around p_c (red dot), that does not intersect with land.

environment. Given a map where landmasses are represented as polygons, we can use the proposed constraint generation method with the line segments making up the edges of the polygons. This will give a convex inner approximation which can be used in the docking planner. This is shown in Figure 8a, where we compute the safe region of water, not intersecting polygonal land masses around a certain point.

In many real world applications however, relying only on map data may not be sufficient, as the maps may be inaccurate, out of date, or missing information. In order to compensate for this we propose using additional sensory information, in order to account for inaccurate map information. For our proposed constraint generation method, point cloud data–such as that generated by LIDAR, radar, or other types of proximity sensors–can be easily incorporated. This can be done by directly using the points in the point cloud as the close points $\boldsymbol{p}_{ab}$. An example of a map augmented with LIDAR data can be seen in Figure 8b. For sensor data such as short range ultrasonic distance measurements, where the sensors are configured as in Figure 9, we can approximate the constraints seen by the sensors as the line segment between the measurement of each sensor. This line segment can then be added to the constraint set similarly to the map data. Using redundant and various exteroceptive sensors is beneficial, as additional sensor may be used to improve coverage and avoid blind spots, which will improve the accuracy of spatial constraints.

When computing the constraint set, it is also possible to use the projection matrix $\boldsymbol{\Sigma}$ in order to create an axis for which we prioritize the set generation. In the case of a vessel, it it can be useful to prioritize constraint generation in the longitudinal direction of the vessel's body frame. This can be done by using the following projection matrix:

$$\boldsymbol{\Sigma} = \boldsymbol{R}(\psi) \begin{bmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{bmatrix} \boldsymbol{R}(\psi)^\top, \tag{30}$$

where choosing $\sigma_x < \sigma_y$ will prioritize set expansion in the direction of the vessel heading. For docking or set point tracking, it is alternatively possible to expand the spatial constraints in the direction of the dock or set point, as this is the direction that we ideally want the vessel to travel.

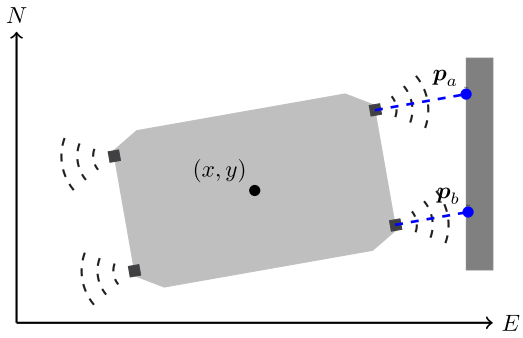


FIGURE 9. Ultrasonic distance sensors attached to the front and rear of the vessel, where the line segments (p_a, p_b) can be added to the spatial constraints.

C. CONSIDERATIONS WHEN USING THE CONVEX SET GENERATION FOR PLANNING

The set generation method we have detailed above, is a computationally efficient way of computing a good inner approximation of a set of non-convex constraints. The goal of the method is not to maximize the area of the convex inner approximation, as this is in general a computationally expensive, and is not guaranteed to create a set which expands in a desired direction. The goal of the method is rather to create an inner approximation with a preferred expansion direction, in our case this is controlled by the expansion ellipse, which is efficient to compute even when handling large numbers of constraints.

When using the constraints from the constraint generation method in the docking planner, it is useful to be able to guarantee recursive feasibility of the planner when it is run in an MPC like fashion. By updating the constraints frequently, and only choosing a new constraint set if it remains feasible for the previous iteration, recursive feasibility of the planner can be guaranteed. We can note that the set generation will always remain feasible for the point $\boldsymbol{p}_c$, however since we are considering all the vessel vertices when planning a safe trajectory, we can not guarantee that the constraint generation method will results in a feasible constraints for all the vessel vertices.

In order to use the linear constraints in an optimization problem, it is practical to have a fixed number of constraints, such that the optimization problem does not need to be transcribed, and built each time a new number of constraints change. In order to do this we use the full constraints generated by the constraint generation method described above, and create a reduced constraint as the K closest constraints in terms of the distance $\boldsymbol{p}_{ab} - \boldsymbol{p}_c$. This reduced constraint, will then be an outer approximation of the full constraints.

V. EXPERIMENTS

A. EXPERIMENTAL PLATFORM

For the sea trials, we used the experimental autonomous urban passenger ferry *milliAmpere*, as shown in Figure 1 with the specifications listed in Table 1, and the planning parameters gien in Appendix. The *milliAmpere* platform has been in constant development at the Norwegian University of Science and Technology (NTNU) since 2017, and has served as a platform for testing and developing autonomous technology, including software, sensor arrays, as well as hardware solutions. A larger version is currently being designed and built by the research group Autoferry,[1] and is planned to operate as an on-demand ferry in the Trondheim harbor.

TABLE 1. ***milliAmpere*** **specifications.**

Dimensions	5m by 2.8m symmetric footprint
Position and heading reference system	Vector VS330 dual GNSS with Real-time kinematic (RTK) capabilities
Thrusters	Two azimuth thrusters on the center line, 1.8m aft and fore of center
Existing control modules	Trajectory tracking DP controller and thrust allocation system
Obstacle detection	Velodyne Puck VLP-16 LIDAR Sensor, and two front mounted ultrasonic range sensors (rear mounted ultrasonic sensor were not available).

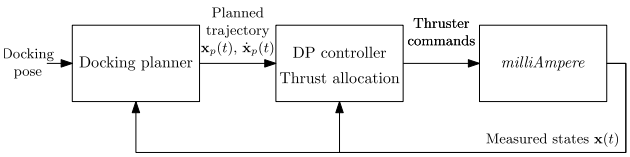


FIGURE 10. Block diagram of the docking system setup. The DP controller and thrust allocation are existing functions on *milliAmpere*.

For the experiments, the docking planner (10) was run with a sampling rate of 0.1Hz, with the output of the docking planner being used as the reference trajectory for a Dynamic Positioning (DP) tracking controller, which was already implemented on *milliAmpere*. The DP tracking controller was a proportional-integral-derivative (PID) controller, with velocity and acceleration feed-forward. Based on the forces and torque computed by the DP controller, the force and angles of the azimuth thrusters were calculated by an optimization-based control allocation scheme [52]. The block diagram in Figure 10 illustrates how the planner, DP controller, and actuators were connected.

Instead of using the DP controller with control allocation, it would have been possible to implement the docking planner (10) as a Nonlinear Model Predictive Control (NMPC) scheme, where the thruster forces computed by the planner are directly used as setpoints for the vessel thrusters. There are however several practical reasons why we chose not to do this:

- The planner does not account for drift, disturbances or modeling errors, while the tracking controller does so through feedback.

[1] Autoferry website: https://www.ntnu.edu/autoferry.

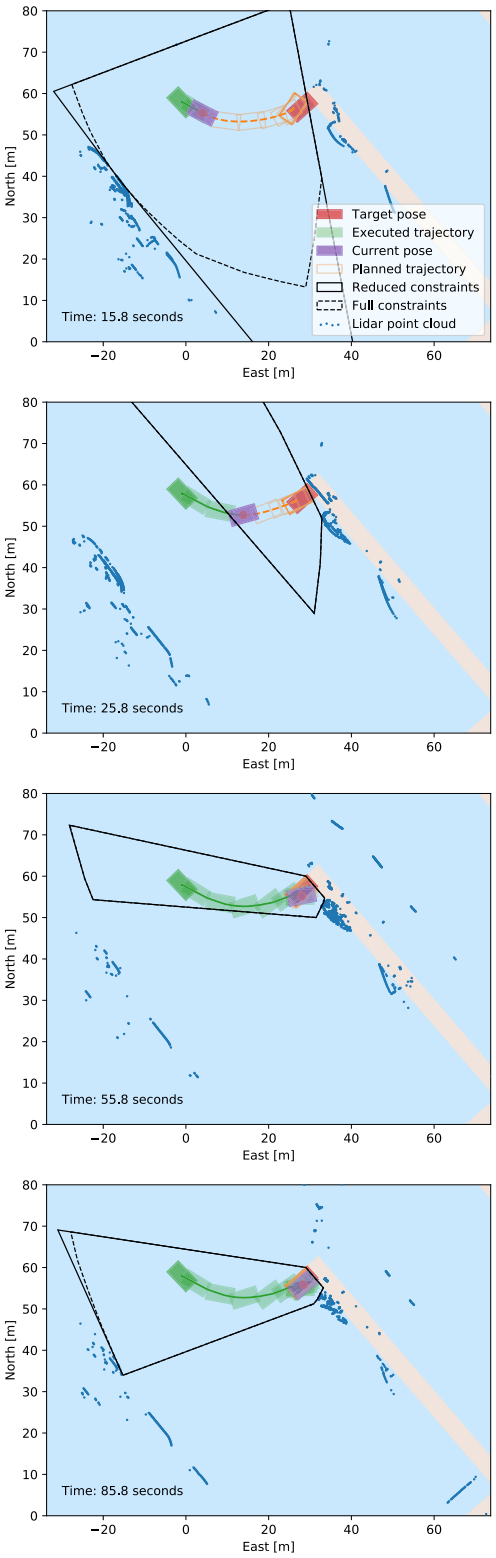


FIGURE 11. Visualisation of the docking motion during the experiments on September 7th 2020 (E1).

- While the planner is iteration-based with no formal performance guarantees, the tracking controller provides a robust bottom layer that acts also as a safety measure.

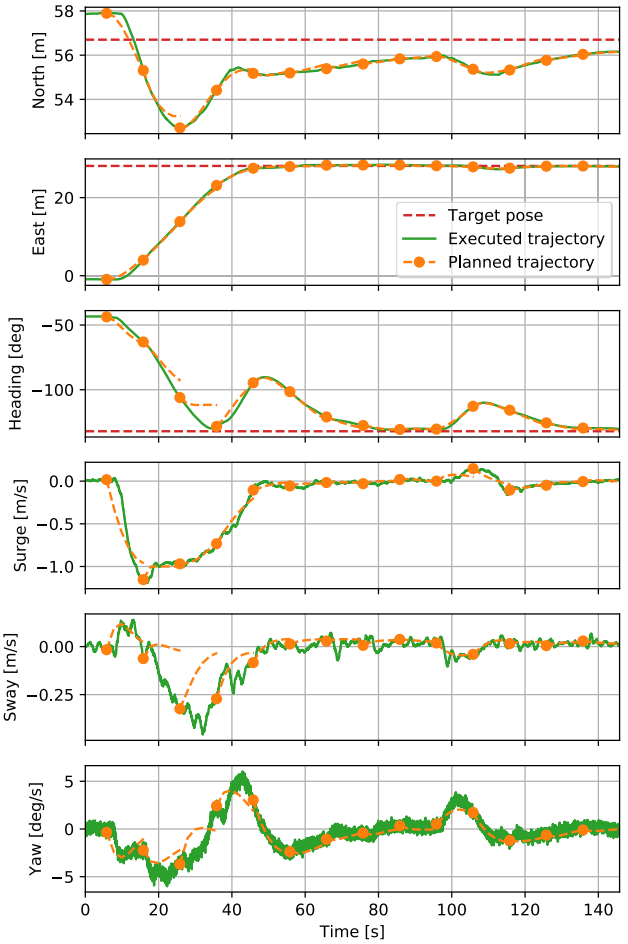


FIGURE 12. Planned and executed trajectory during the experiments on September 7th 2020 (E1).

- Due to the computational demand of solving the planning problem, it is difficult to achieve a sampling rate that is fast enough to stabilize the vessel through feedback from the planner.
- Using this multi-layer architecture separates the planner from the vessel control systems, allowing the planner to easily be retrofitted onto the existing implementation.

Choosing such a multi-layered structure, where planning and motion control are separated, provides flexibility in the trajectory planner, disturbance rejection through feedback, robustness to failures in the planning level, and ease of implementation on platforms with existing motion control systems.

B. RESULTS

Experiments[2] were performed with the *milliAmpere* platform in confined waters in Trondheim, Norway on September 7th (E1) and 11th (E2), 2020. The weather conditions were calm with winds between $1 m/s$ and $3\ m/s$. The results from E1 are shown in Figure 11 and 12, and the results from the E2 are shown in Figure 13 and 14, with photos from the experiments in Figure 15. It should be noted that the ultrasonic distance

[2]Video of experiments is available at: https://youtu.be/AyaWlJvI6K8

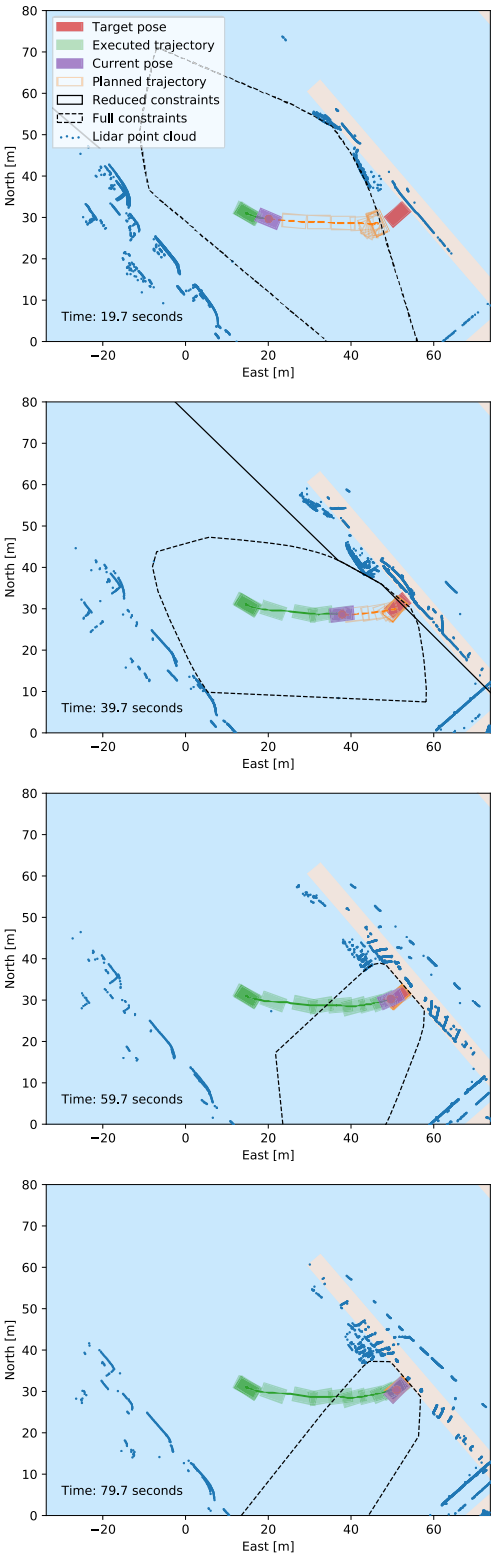


FIGURE 13. Visualisation of the docking motion during the experiments on September 11th 2020 (E2).

measurements were not used due to technical problems with the sensors at the time, meaning that only lidar and mapping data was used for computing the spatial constraints during the tests.

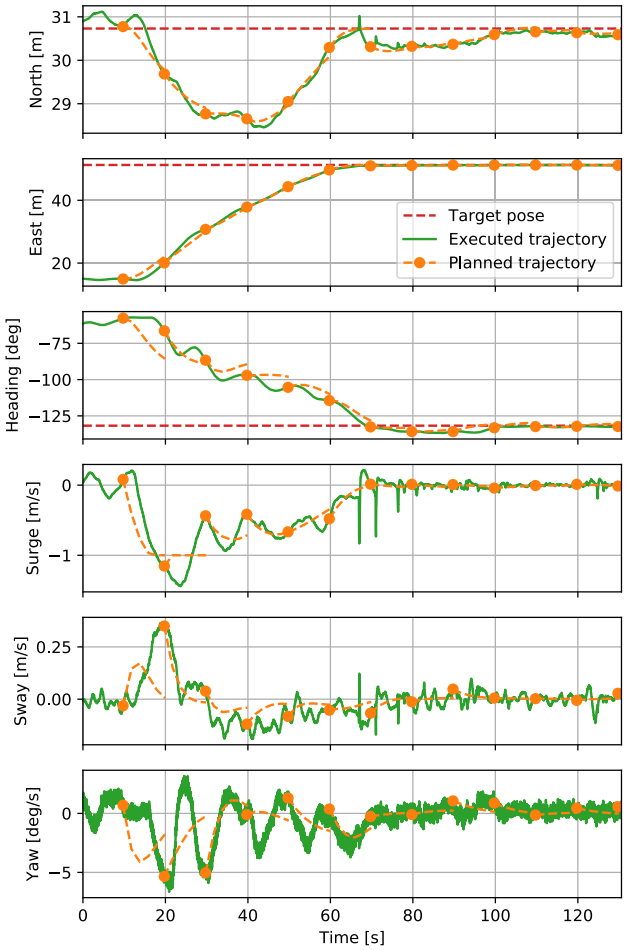


FIGURE 14. Planned and executed trajectory during the experiments on September 11th 2020 (E2).

The experiments E1 were performed at the end of a floating dock, while E2 at the center of the floating dock, due to to space availability on each day of the experiments. The difference in final docking pose had an influence on the complexity of the constraint sets, which can be seen in Figure 11 and 13. The *full constraints* pertain to the full set of constraints generated by the constraint generation method (25), while the *reduced constraints* were chosen as the 8 closest constraints eventually used in the optimization problem. This was done since the optimization problem needs a fixed number of constraints, and 8 gives a good balance between accuracy and computational cost when solving the OCP. For E1, shown in Figure 11, we see that the reduced constraints are much closer to the full constraints, compared to E2 Figure 13. This indicates that more potential obstacles were present in E2. The results show that the proposed constraint generation method is able to construct a good convex inner approximation of the free region, within which the vessel is allowed to operate, and that by choosing the number of constraints to reduce the full constraints to, we can achieve a good balance in terms of computation and constraint accuracy. We should however note the unexpected set of constraints at 25.8 seconds, in Figure 11, which was caused by rain

FIGURE 15. The *milliAmpere* while performing the docking maneuver, including closing in on the dock, and the final docking pose.

during the experiment, leading to the LIDAR misclassifying a raindrop as a potential obstacle. This can be avoided by better filtering the incoming LIDAR data before feeding them to the set generation algorithm. It is also worth noting the LIDAR point clouds in Figure 11 and 13, are not capturing the dock itself when the vessel is close to the final docking pose. This is due to limitations in the vertical transmitting angle of the LIDAR causing the dock to end up in the LIDAR blind spot. This problem can be solved by using the ultrasonic distance sensors at close range, adding redundant LIDAR sensors to improve coverage, or in this case relying on the map data, as the ultrasonic distance sensors were unavailable.

Figures 11 and 13 also show how the LIDAR helps in detecting unmapped static obstacles, in this case mostly docked vessels, especially as the milliAmpere gets closer to them. Accounting for these surrounding obstacles is of critical importance when planning and tracking a safe docking trajectory, and would not have been possible if only map data were used.

The results indicate that the planned trajectory does not violate any of the spatial constraints, and ensures that the vessel does not collide with surrounding obstacles. Also, the generated trajectory is intuitive for a docking operation, as the vessel initially moves in the surge direction towards the dock with a reasonably high velocity, then it slows down as it gets closer to the final docking pose, and finally rotates in order to reach the desired docking heading. Figures 12 and 14, indicate that the final trajectory mostly converges to the desired docking pose, with one exception being the North direction in Figure 12. This discrepancy was due to the docking pose overlapping with the dock, as can be seen in Figure 11, and hence the desired docking pose is not possible to be reached without violating the spatial constraints.

In Figure 12 and 14, we see the executed trajectory given in green, and the planned trajectory given in orange, where every 10 seconds the start of a replanned trajectory is marked with a dot. The observed discontinuity in the planned trajectory is due to the replanned trajectory being initialized to the state of the vessel at the time of the replanning. For both experiments we see that the tracking performance is very good. The tracking performance is highly reliant on not only the performance of the underlying DP controller and thrust allocation algorithm, but also the accuracy of the model used in the optimization-based planner. The most notable discrepancies in the tracking performance are found in the heading. We believe these are due to the the inherent heading instability of the vessel, as well as unmodeled thruster dynamics, which may cause a slight delay between desired and produced thrust.

VI. CONCLUSION

We have presented a method for planning and performing docking maneuvers in a confined harbor. The method utilizes map data, which is known in advance, as well as sensor data gathered in real time, to iteratively and safely plan a trajectory that brings the vessel to a desired docking pose. To perform the docking maneuvers given by the planner, we used an existing trajectory tracking DP controller, which added robustness to disturbances, and helped demonstrate that the planner is easy to retrofit on an existing platform. In order to validate the proposed control scheme, we conducted full-scale experiments in a confined harbor area with the *milliAmpere* ferry developed at NTNU, and demonstrated how the proposed method is able to plan and well as execute safe and collision-free docking maneuvers. To the best of our knowledge, there's no existing published work that involves field trials of docking operations for autonomous surface vehicles using only exteroceptive sensors.

For future work, we would like to look at the possibility of integrating additional sensors, such as radar and cameras, in order to generate an even better view of the environment. Additionally, we would like to look at ways of filtering the sensor data in order to get more reliable sensor readings. We would also like to integrate the docking system in a control structure that handles transportation phases autonomously. Since our proposed approach is able to handle the docking and undocking phase, what remains to achieve a fully autonomous operation with mission objective "Navigate from Dock A to Dock B", is the development of control and planning strategies for handling the transit phase of the journey. This development will include additional work on collision avoidance, situational awareness, and planning methods that comply with the maritime navigation rules.

APPENDIX

LIST OF PARAMETER VALUES

A list of parameter values is given in Table 2, while details on the vessel model can be found in [33].

TABLE 2. List of parameters.

Symbol	Value	Description
$q_{x,y}$	1	Huber penalty weight
q_ψ	20	Heading penalty weight
$\boldsymbol{Q}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 200 & 0 \\ 0 & 0 & 100 \end{bmatrix}$	Velocity weight matrix
$\boldsymbol{R}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	Thruster force weight matrix
δ	10	Huber penalty slope
$\boldsymbol{\Sigma}$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	Expansion ellipse weights
T	120s	OCP prediction horizon
N	60	OCP shooting intervals
d	3	OCP Legendre polynomial degree
K	8	Reduced constraint size

ACKNOWLEDGMENT

The authors would like to thank Emil Hjelseth Thyri for his help during the experiments.

REFERENCES

[1] E. Murdoch, C. Clarke, I. W. Dand, and B. Glover, *A Master's Guide to Berthing*. Livingston, U.K.: Witherby & Company, 2004.
[2] K. Hasegawa and T. Fukutomi, "On harbor maneuvering and neural control system for berthing with tug operation," in *Proc. 3rd Int. Conf. Manoeuvring Control Mar. Craft*, 1994, pp. 197–210.
[3] V. P. Bui, H. Kawai, Y. B. Kim, and K. S. Lee, "A ship berthing system design with four tug boats," *J. Mech. Sci. Technol.*, vol. 25, no. 5, pp. 1257–1264, May 2011.
[4] P. Van Bui and Y. B. Kim, "Development of constrained control allocation for ship berthing by using autonomous tugboats," *Int. J. Control, Autom. Syst.*, vol. 9, no. 6, pp. 1203–1208, Dec. 2011.
[5] V. L. Tran and N. Im, "A study on ship automatic berthing with assistance of auxiliary devices," *Int. J. Nav. Archit. Ocean Eng.*, vol. 4, no. 3, pp. 199–210, Sep. 2012.
[6] G. J. S. Rae, S. M. Smith, D. T. Anderson, and A. M. Shein, "A fuzzy rule based docking procedure for two moving autonomous underwater vehicles," in *Proc. Amer. Control Conf.*, Jun. 1993, pp. 580–584.
[7] K. Teo, B. Goh, and O. K. Chai, "Fuzzy docking guidance using augmented navigation system on an AUV," *IEEE J. Ocean. Eng.*, vol. 40, no. 2, pp. 349–361, Apr. 2015.
[8] V.-S. Nguyen and N.-K. Im, "Automatic ship berthing based on fuzzy logic," *Int. J. FUZZY Log. Intell. Syst.*, vol. 19, no. 3, pp. 163–171, Sep. 2019.
[9] H. Yamato, "Automatic berthing by the neural controller," in *Proc. 9th Ship Control Syst. Symp.*, vol. 3, 1990, pp. 3183–3201.
[10] K. Hasegawa, "Automatic berthing control system using network and knowledgebase," *J. Soc. Naval Architects Jpn.*, vol. 220, pp. 135–143, 1993.
[11] Y. Zhang, G. E. Hearn, and P. Sen, "A multivariable neural controller for automatic ship berthing," *IEEE Control Syst.*, vol. 17, no. 4, pp. 31–45, Aug. 1997.
[12] N. Im and K. Hasegawa, "A Srudy on automatic ship berthing using parallel neural controller," *J. Kansai Soc. Naval Architects, Jpn.*, vol. 2001, no. 236, pp. 65–70, 2001.
[13] Y. A. Ahmed and K. Hasegawa, "Automatic ship berthing using artificial neural network trained by consistent teaching data using nonlinear programming method," *Eng. Appl. Artif. Intell.*, vol. 26, no. 10, pp. 2287–2304, Nov. 2013.
[14] N.-K. Im and V.-S. Nguyen, "Artificial neural network controller for automatic ship berthing using head-up coordinate system," *Int. J. Nav. Archit. Ocean Eng.*, vol. 10, no. 3, pp. 235–249, May 2018.
[15] V.-S. Nguyen, V.-C. Do, and N.-K. Im, "Development of automatic ship berthing system using artificial neural network and distance measurement system," *Int. J. FUZZY Log. Intell. Syst.*, vol. 18, no. 1, pp. 41–49, Mar. 2018.
[16] D. Lee, S.-J. Lee, and S. Yu-Jeong, "Application of recent developments in deep learning to ann-based automatic berthing systems," *Int. J. Eng. Technol. Innov.*, vol. 10, no. 1, p. 75, 2019.
[17] N. Mizuno and R. Kuboshima, "Implementation and evaluation of nonlinear optimal feedback control for ship's automatic berthing by recurrent neural network," *IFAC-PapersOnLine*, vol. 52, no. 21, pp. 91–96, 2019.
[18] E.-L. H. Rørvik, "Automatic docking of an autonomous surface vessel," M.S. thesis, Dept. Eng. Cybern., Norwegian Univ. Sci. Technol. (NTNU), Trondheim, Norway, 2020. [Online]. Available: https://hdl.handle.net/11250/2656724
[19] H. Yamato, T. Koyama, and T. Nakagawa, "Automatic berthing using the expert system," *IFAC Proc. Volumes*, vol. 25, no. 3, pp. 173–184, Apr. 1992.
[20] M. Breivik and J.-E. Loberg, "A virtual target-based underway docking procedure for unmanned surface vehicles," *IFAC Proc. Volumes*, vol. 44, no. 1, pp. 13630–13635, Jan. 2011.
[21] J. Woo and N. Kim, "Vector field based guidance method for docking of an unmanned surface vehicle," in *Proc. 12th ISOPE Pacific/Asia Offshore Mech. Symp.* Mountain View, CA, USA: International Society of Offshore and Polar Engineers, 2016, pp. 1–6.
[22] A. Farnsworth, *Auto-docking ferry successfully tested in Norway*. Helsinki, Finland: Wärtsilä, 2018. [Online]. Available: https://www.wartsila.com/twentyfour7/innovation/look-ma-no-hands-auto-d ocking-ferry-successfully-tested-in-norway
[23] Wärtsilä, Helsinki, Finland. (2018). *Rolls-Royce and Finferries Demonstrate World's First Fully Autonomous Ferry*. Rolls-Royce Press Release. [Online]. Available: https://www.rolls-royce.com/media/press-releases/2018/03-12-2018-rr-and -finferries-demonstrate-worlds-first-fully-autonomous-ferry.aspx
[24] Kongsberg. (2020). *Automatic Ferry Enters Regular Service Following World-First Crossing With Passengers Onboard*. Kongsberg Press Release. [Online]. Available: https://www.kongsberg.com/maritime/about-us/news-and-media/news-archive /2020/first-adaptive-transit-on-bastofosen-vi/
[25] K. Shouji, K. Ohtsu, and S. Mizoguchi, "An automatic berthing study by optimal control techniques," *IFAC Proc. Volumes*, vol. 25, no. 3, pp. 185–194, Apr. 1992.
[26] K. Djouani and Y. Hamam, "Minimum time-energy trajectory planning for automatic ship berthing," *IEEE J. Ocean. Eng.*, vol. 20, no. 1, pp. 4–12, 1995.
[27] K. Ohtsu, K. Shoji, and T. Okazaki, "Minimum-time maneuvering of a ship, with wind disturbances," *Control Eng. Pract.*, vol. 4, no. 3, pp. 385–392, Mar. 1996.
[28] T. Okazaki, K. Ohtsu, and N. Mizuno, "A study of minimum time berthing solutions," *IFAC Proc. Volumes*, vol. 33, no. 21, pp. 135–139, Aug. 2000.
[29] N. Mizuno, Y. Uchida, and T. Okazaki, "Quasi real-time optimal control scheme for automatic berthing," *IFAC-PapersOnLine*, vol. 48, no. 16, pp. 305–312, 2015.
[30] M. C. Nielsen, T. A. Johansen, and M. Blanke, "Cooperative rendezvous and docking for underwater robots using model predictive control and dual decomposition," in *Proc. Eur. Control Conf. (ECC)*, Jun. 2018, pp. 14–19.
[31] N. Mizuno, T. Kita, and T. Ishikawa, "A new solving method for nonlinear optimal control problem and its application to automatic berthing problem," in *Proc. 44th Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Oct. 2018, pp. 2183–2188.
[32] A. B. Martinsen, A. M. Lekkas, and S. Gros, "Autonomous docking using direct optimal control," *IFAC-PapersOnLine*, vol. 52, no. 21, pp. 97–102, 2019.
[33] G. Bitar, A. B. Martinsen, A. M. Lekkas, and M. Breivik, "Trajectory planning and control for automatic docking of ASVs with full-scale experiments," 2020, *arXiv:2004.07793*. [Online]. Available: https://arxiv.org/abs/2004.07793
[34] S. Li, J. Liu, R. R. Negenborn, and Q. Wu, "Automatic docking for underactuated ships based on multi-objective nonlinear model predictive control," *IEEE Access*, vol. 8, pp. 70044–70057, 2020.
[35] T. Takai and K. Ohtsu, "Automatic berthing experiments using 'Shioji-maru,'" *The J. Jpn. Inst. Navigat.*, vol. 83, pp. 267–276, Mar. 1990.

[36] L. Gucma, A. Bak, M. Gucma, S. Jankowski, P. Zalewski, and M. Perkovic, "Laser docking system integrated with pilot navigation support system. Background to high precision, fast and reliable vessel docking," in *Proc. 17th Saint Petersburg Int. Conf. Integr. Navigat. Syst. (ICINS)*, 2010, pp. 268–275.
[37] M. Perkovic, M. Gucma, B. Luin, L. Gucma, and T. Brcko, "Accommodating larger container vessels using an integrated laser system for approach and berthing," *Microprocessors Microsyst.*, vol. 52, pp. 106–116, Jul. 2017.
[38] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. Hoboken, NJ, USA: Wiley, 2011.
[39] T. A. Johansen and T. I. Fossen, "Control allocation—A survey," *Automatica*, vol. 49, no. 5, pp. 1087–1103, May 2013.
[40] S. Gros and M. Zanon, "Penalty functions for handling large deviation of quadrature states in NMPC," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3848–3860, Aug. 2017.
[41] S. Gros and M. Diehl, "NMPC based on huber penalty functions to handle large deviations of quadrature states," in *Proc. Amer. Control Conf.*, Jun. 2013, pp. 3159–3164.
[42] G. A. Hicks and W. H. Ray, "Approximation methods for optimal control synthesis," *Can. J. Chem. Eng.*, vol. 49, no. 4, pp. 522–528, Aug. 1971.
[43] P. Deuflhard, "A modified Newton method for the solution of ill-conditioned systems of nonlinear equations with application to multiple shooting," *Numerische Math.*, vol. 22, no. 4, pp. 289–315, Aug. 1974.
[44] T. H. Tsang, D. M. Himmelblau, and T. F. Edgar, "Optimal control via collocation and non-linear programming," *Int. J. Control*, vol. 21, no. 5, pp. 763–768, May 1975.
[45] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Algorithmic Foundations of Robotics XI*. Cham, Switzerland: Springer, 2015, pp. 109–124.
[46] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1688–1695, Jul. 2017.
[47] T. Schoels, P. Rutquist, L. Palmieri, A. Zanelli, K. O. Arras, and M. Diehl, "CIAO*: MPC-based safe motion planning in predictable dynamic environments," 2020, *arXiv:2001.05449*. [Online]. Available: https://arxiv.org/abs/2001.05449
[48] K. Bergman, O. Ljungqvist, J. Linder, and D. Axehill, "An optimization-based motion planner for autonomous maneuvering of marine vessels in complex environments," 2020, *arXiv:2005.02674*. [Online]. Available: http://arxiv.org/abs/2005.02674
[49] J. Telgen, "Identifying redundant constraints and implicit equalities in systems of linear constraints," *Manage. Sci.*, vol. 29, no. 10, pp. 1209–1222, Oct. 1983.
[50] S. Paulraj and P. Sumathi, "A comparative study of redundant constraints identification methods in linear programming problems," *Math. Problems Eng.*, vol. 2010, Dec. 2010, Art. no. 723402.
[51] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, no. 4, pp. 469–483, Dec. 1996.
[52] T. R. Torben, A. H. Brodtkorb, and A. J. Sørensen, "Control allocation for double-ended ferries with full-scale experimental results," in *Proc. 12th IFAC CAMS*, Daejeon, South Korea, 2019, pp. 45–50.

ANDREAS B. MARTINSEN received the M.Sc. degree in engineering cybernetics from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 2018, where he is currently pursuing the Ph.D. degree in engineering cybernetics.

His research interests include reinforcement learning, optimal control, and machine learning, with a focus on marine and maritime applications.

GLENN BITAR received the B.S. degree in computer science and industrial automation from Telemark University College, Porsgrunn, Norway, in 2015, and the M.Sc. degree in cybernetics and robotics from the Department of Engineering Cybernetics, NTNU, Trondheim, Norway, in 2017, where he is currently pursuing the Ph.D. degree in researching energy-optimized autonomous motion control for ships.

ANASTASIOS M. LEKKAS received the M.Sc. degree in mechanical and aeronautical engineering from the University of Patras, Patras, Greece, in 2008, and the Ph.D. degree in engineering cybernetics from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 2014.

He is currently an Associate Professor of autonomous systems with the Department of Engineering Cybernetics, NTNU. He is the Project Manager of the EXAIGON project, where the main goal is to develop explainable AI methods for safety- and business-critical applications. His research interests include merging artificial intelligence with control engineering in order to develop cyber-physical systems with increased autonomy.

SÉBASTIEN GROS received the Ph.D. degree in mechatronics from the École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, in 2007.

He was part of a Research and Development Group hosted at Strathclyde University, Glasgow, U.K., focusing on wind turbine control. In 2011, he joined the University of KU Leuven, Leuven, Belgium, where his main research focus was on optimal control and fast NMPC for complex mechanical systems. He joined the Department of Signals and Systems, Chalmers University of Technology, Göteborg, Sweden, in 2013, where he became an Associate Professor, in 2017. He is currently a Full Professor with the Norwegian University of Science and Technology, Norway, and an Affiliate Professor with Chalmers. His research interests include numerical methods, real-time optimal control, reinforcement learning, and the optimal control of energy-related applications.