Mechatronics 69 (2020) 102399

Contents lists available at ScienceDirect

Mechatronics

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

Set-based collision avoidance applications to robotic systems☆

Signe Moe[a,b,*], Kristin Y. Pettersen[c], Jan Tommy Gravdahl[a]

[a] Department of Engineering Cybernetics (ITK), Norwegian University of Science and Technology (NTNU), Trondheim, Norway
[b] Department of Mathematics and Cybernetics, SINTEF Digital, Oslo, Norway
[c] NTNU Center for Autonomous Marine Operations and Systems (NTNU AMOS), ITK, NTNU, Trondheim, Norway

ARTICLE INFO

Article history:
Received 30 October 2019
Revised 25 March 2020
Accepted 13 June 2020

Keywords:
Collision avoidance
Kinematic control
Robotic manipulator
Unmanned surface vessel
Unicycle
Multi-agent system

ABSTRACT

A robotic system can consist of a single or multiple agents with a fixed or mobile base, with full or under-actuation, and possibly redundancy. Collision avoidance is a crucial task for any robotic system and is necessary to ensure safe operation. In this paper, we use a set-based approach to ensure collision avoidance as a high-priority task of a robotic system while simultaneously defining one or more tasks for the system to achieve. The set-based approach is highly generic and flexible, and we present theoretical results, practical implementation and experimental results of the approach applied to a redundant, fully actuated robot manipulator, a full-scale underactuated surface vessel and a multi-agent system of unicycles.

1. Introduction

Traditionally, robotic systems are controlled in the joint space. A typical system for joint space control is illustrated in Fig. 1. The kinematic controller calculates reference states based on the desired behavior (which is often specified in task space), and the current state of the system. The reference states are the input of the dynamic controller, which calculates and imposes forces and torques on the actual system through the actuators. Note that in kinematic control it is common to assume that the reference state is tracked and simply use this as feedback to the kinematic controller rather than the actual state [1]. This is illustrated by the dashed arrow in Fig. 1.

Robotic systems may be required to perform one or several tasks, for instance obtaining a certain desired end effector position and/or orientation. Hence, a variety of inverse kinematics algorithms have been developed to map tasks from task space to the joint space and thus generate reference trajectories for the dynamic controller, all the while being able to handle singularities and non-square matrices. The most common approach is to use a Jacobian-based method [2–4]. In particular, the pseudo-inverse Jacobian is, like the damped least-squares solution, defined for systems that are not square nor have full rank and is a widely used solution to the inverse kinematics problem [5–7].

A robotic system is said to be kinematically redundant if it possesses more degrees of freedom (DOFs) than those required to perform a certain task [8]. In this case, the "excess" DOFs can be utilized in order to perform several tasks using null-space-based (NSB) behavioral control. Furthermore, it is useful to sort tasks in a prioritized order to handle potentially conflicting tasks. The singularity-robust multiple task-priority inverse kinematics framework (SRMTPIK) is a kinematic control method which ensures that the task errors converge to zero given that certain, specified assumptions are satisfied. [1,9]. This well-known framework may be utilized on a number of different robotic systems for a wide range of applications [10–12].

The SRMTPIK framework has been developed for *equality tasks*. Equality tasks specify exactly one desired value for given states of the system, for instance the position and orientation of the end effector. However, for a general robotic system, several goals may not be described as equality tasks, but rather as *set-based tasks*, which are tasks that have a desired interval of values rather than one exact desired value. Such tasks are also referred to as inequality constraints. Examples of such tasks are staying within joint limits [13], collision/obstacle avoidance [14] and field of view (FOV). As recognized in [15], the multiple task-priority inverse kinematics algorithm is not suitable to handle set-based tasks directly, and these tasks are therefore usually transformed into more and unnecessary

☆ This paper was recommended for publication by Associate Editor Michael Ruderman, Dr.-Ing

* Corresponding author at: Department of Engineering Cybernetics (ITK), Norwegian University of Science and Technology (NTNU), Trondheim, Norway.
E-mail addresses: signe.moe@ntnu.no (S. Moe), kristin.y.pettersen@ntnu.no (K.Y. Pettersen), jan.tommy.gravdahl@ntnu.no (J.T. Gravdahl).

https://doi.org/10.1016/j.mechatronics.2020.102399

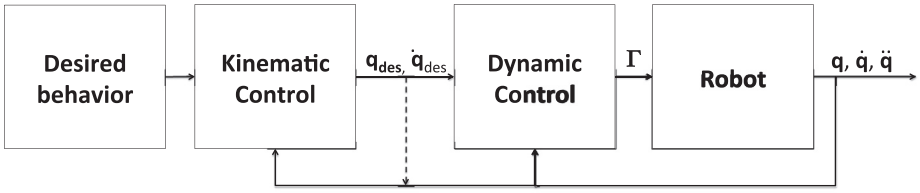


Fig. 1. A typical control structure for a robotic system with joint configuration $\boldsymbol{q}$ and joint velocities and accelerations $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$. Based on the desired behavior and the current state, the kinematic controller calculates reference states and the dynamic controller calculates actuator forces and torques $\boldsymbol{\Gamma}$ which are imposed on the actual system.

restrictive equality constraints through potential fields or cost functions [16,17].

In [18], the SRMTPIK framework is extended to systematically handle set-based tasks in addition to equality tasks, thereby expanding an already generic, widely used method to handle all types of tasks in a consistent, unified manner. It is proven that high-priority set-based tasks remain in their valid set at all times, whereas lower-priority set-based tasks cannot be guaranteed to be satisfied due to the influence of the higher-priority equality tasks. Furthermore, it is proven that the equality task errors converge asymptotically to zero given certain assumptions, and experimental results are presented that illustrate the effectiveness of the method.

The set-based task of collision avoidance is crucial both for fixed and floating base systems. A robot manipulator must avoid collisions with the environment, itself and humans in its workspace to prevent injuries, damages on equipment and delayed operations. Similarly, mobile robots must prevent collisions with both static and dynamic obstacles. The motion of a mobile robot such as an underactuated surface vessel (USV) or a ground vehicle is commonly controlled through a guidance, navigation and control (GNC) system [19], where the guidance and control system corresponds to the kinematic and dynamic control in Fig. 1. Such vehicles are generally underactuated since they typically lack control inputs in the sideways direction (sway). Thus, the guidance and control system must fulfill the control objectives using only the available actuators in surge (e.g. thruster/engine force) and yaw (e.g. rudder/steering angle). The control system determines the required control inputs to track the reference states, which are provided by the guidance system. In the case when a mobile robot is given a path following task, the guidance system typically consists of guidance laws for the desired heading and forward velocity that, if tracked, result in the robot converging to and following the desired path. The SRMTPIK framework [9] and the extension to set-based tasks [18] have been developed for redundant and fully actuated systems and must therefore be adapted to the case of underactuated mobile robots.

There exist several methods to achieve collision avoidance both for fixed and floating base systems. Potential fields [20], dynamic window [21] and velocity obstacles [22] are widely used collision avoidance approaches. However, use of potential fields may result in oscillations [23], and the dynamic window method assumes that the sideways velocity is zero. Thus, it is unsuitable for marine vessels, which glide sideways when following curved paths and/or under the influence of ocean currents. The velocity obstacle (VO) approach is not computationally heavy and is straightforward to comply with the International Regulations for Preventing Collisions at Sea 1972 (COLREGs). However, it is challenging to implement and to combine with existing guidance methods for instance path following. In [24] a reactive collision avoidance algorithm using a constant avoidance angle approach is presented. The algorithm provably avoids a moving obstacle by steering the vehicle a constant avoidance angle to the side of it. The results are extended to 3D for underwater vehicles in [25] and experimentally verified for both surface and underwater vehicles. The approach is modular and able to incorporate COLREGs. However, when a large safety margin is required, the approach is quite conservative and developed specifically for marine vehicles. By applying set-based theory, the collision avoidance problem may be handled both for fixed and floating base systems in a unified manner. In [26], experimental results are presented for an industrial manipulator which successfully circumvents obstacles along its trajectory. In this case, the obstacles are defined by their position and a safe radius which the end effector will never enter. Thus, obstacles are modeled as spheres. The same principle is applied for surface vessels in [27,28], where the main task of the USV is to follow a predefined path. A COLREGs compliant system is and simulated achieved through a set-based guidance system which switches between path following and collision avoidance. In the case of collision avoidance, the USV converges to and tracks a safe radius about the obstacle.

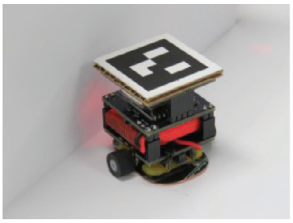
In this paper, we present experimental results for set-based collision avoidance for three different platforms, thereby illustrating the effectiveness and usefulness of this highly generic method: an industrial manipulator, unicycles and R/V Gunnerus, a 31 m research vessel (see Fig. 2). The adaptation to various platforms and the experimental results are the main contribution of this paper and are mostly based on theory presented in [18]. Experiments from the Robotarium [29] are presented for a multi-agent system of unicycles, where the same control system has been implemented on all agents to ensure path following and collision avoidance. Furthermore, the guidance system presented in [28] has been implemented on R/V Gunnerus and experimentally verified for a variety of COLREGs scenarios in Trondheimsfjorden, Norway. In addition, this paper presents new theoretical results where obstacles are represented as columns, which complements the previously presented representation of circular/spherical obstacles [18]. These results are experimentally verified on a manipulator, which illustrates that set-based collision avoidance is not limited to obstacles of a specific shape.

This paper is organized as follows. Section 2 gives a brief introduction to set-based kinematic control and the notation used in this paper. The remainder of the paper focuses on set-based control for collision avoidance in particular. In Section 3, collision avoidance for redundant, fully actuated systems is considered. In particular, a set-based task to avoid column-shaped obstacles is defined, and validated with experimental results using the UR5 manipulator in Section 4. Section 5 presents guidance laws for path following and collision avoidance for floating base vehicles. In Section 6 and 8, these guidance laws are applied in a set-based framework to achieve path following and collision avoidance for floating base single- and multi-agent systems, respectively. Furthermore, experimental results are presented for both cases in Section 7 and 9. Finally, conclusions are given in Section 10.

2. Set-based kinematic control

In this paper, we consider systems with one set-based task, namely collision avoidance, and one equality task, e.g. position control or path following. Due to the importance of this task, it is always considered high-priority, i.e. a task that must be satisfied

(a) The 6-DOF UR5 manipulator from Universal Robots.



(b) The GRITSBot unicycle used by the Robotarium.

(c) NTNU's research vessel, R/V Gunnerus.

Fig. 2. The test platforms in this paper.

at all times. Therefore, this section briefly describes the principle of set-based kinematic control for systems with one high-priority set-based task and one equality task. For more detail, a description of a system with a general number of both high- and low-priority set-based tasks and stability properties, the interested reader is referred to [18].

In this paper, equality tasks are denoted with number subscripts and set-based tasks, which are scalar, with letter subscripts, e.g. $\boldsymbol{\sigma}_1$ and σ_a. Consider a robotic system with n DOFs. The system configuration is given by $\boldsymbol{q}$. The equality task error is defined as $\tilde{\boldsymbol{\sigma}}_1(\boldsymbol{q}) \triangleq \boldsymbol{\sigma}_{1,\mathrm{des}} - \boldsymbol{\sigma}_1(\boldsymbol{q})$. The high-priority set-based task is denoted $\sigma_\mathrm{a}(\boldsymbol{q})$, and the valid set for this task is given by $C_\mathrm{a} = \left[\sigma_{\mathrm{a,min}}, \sigma_{\mathrm{a,max}}\right]$. Typically a collision avoidance task σ_a is defined as the distance between the robotic system and the obstacle center, whereas $\sigma_{\mathrm{a,min}}$ is defined as a safe minimum distance between the robot end effector/center of mass and an obstacle, and $\sigma_{\mathrm{a,max}} = \infty$.

The SRMTPIK framework is based on the Jacobian matrix of a task. In general, a task value can be expressed as a function of the system configuration $\boldsymbol{q}$. For instance, a typical task $\boldsymbol{\sigma}$ is the position of a robotic end effector $\boldsymbol{p}$, which is a function of the joint angles,

$$\boldsymbol{\sigma}(\boldsymbol{q}) = \boldsymbol{p}(\boldsymbol{q}) = \boldsymbol{f}(\boldsymbol{q}). \tag{1}$$

The exact function $\boldsymbol{f}$ is system specific and can be derived through for instance the Denavit-Hartenberg convention [31]. The Jacobian matrix of the task $\boldsymbol{\sigma}$ is then found through the chain rule of derivation,

$$\dot{\boldsymbol{\sigma}} = \frac{\delta \boldsymbol{f}(\boldsymbol{q})}{\delta \boldsymbol{q}} \dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{2}$$

In particular, the configuration-dependent Jacobian matrix relates a change in the configuration to a corresponding change in the task, e.g. calculates the velocity of the end effector based on the joint angles and velocity.

The principle of set-based control is to ignore the set-based task as long as it remains in its valid interval and control only the equality task. However, should this result in the set-based task being violated, it is inserted into the task priority order and actively controlled to remain on the boundary of its valid set until such a time that controlling only the equality task naturally brings the set-based task back into its valid interval. Thus, two modes must be considered.

Mode 1 is the "default" solution, whereas mode 2 should be activated only when necessary. Using the SRMTPIK framework [1], mode 1 corresponds to

$$\dot{\boldsymbol{q}}_{1,\mathrm{des}}(\boldsymbol{q}) = \boldsymbol{J}_1^\dagger(\boldsymbol{q})\boldsymbol{\Lambda}_1\tilde{\boldsymbol{\sigma}}_1(\boldsymbol{q}) \tag{3}$$

where $\boldsymbol{\Lambda}_1$ is a positive definite gain matrix for task 1 and $\boldsymbol{J}_1^\dagger(\boldsymbol{q})$ denotes the pseudo-inverse of the Jacobian matrix of task 1 [9]. Note that for readability, the argument $\boldsymbol{q}$ is omitted from the equations from here on.

In mode 1, the set-based task evolves freely according to the equality task without being actively controlled. If mode 1 would result in σ_a leaving the set C_a, σ_a is considered the first priority task and mode 2 is activated. An equality task with the goal of keeping σ_a at its current value is added as the highest priority task, i.e. the desired task value $\sigma_{\mathrm{a,des}}$ is equal to the current task value σ_a. Thus, the task error $\tilde{\sigma}_\mathrm{a} = \sigma_{\mathrm{a,des}} - \sigma_\mathrm{a} \equiv 0$. The system is then defined by the following equation:

$$\dot{\boldsymbol{q}}_{2,\mathrm{des}} = \boldsymbol{J}_\mathrm{a}^\dagger\tilde{\sigma}_\mathrm{a} + \boldsymbol{N}_\mathrm{a}\boldsymbol{J}_1^\dagger\boldsymbol{\Lambda}_1\tilde{\boldsymbol{\sigma}}_1 = \boldsymbol{N}_\mathrm{a}\boldsymbol{J}_1^\dagger\boldsymbol{\Lambda}_1\tilde{\boldsymbol{\sigma}}_1 \tag{4}$$

Here, $\boldsymbol{N}_\mathrm{a}$ is the null-space matrix for the set-based task. The switching between modes is determined by the *tangent cone* [18]. The implementation of the tangent cone T_C of the set $C = [\sigma_{\min}, \sigma_{\max}]$ is defined by Algorithm 1 . By applying the common

Algorithm 1: The boolean function in_T_C.

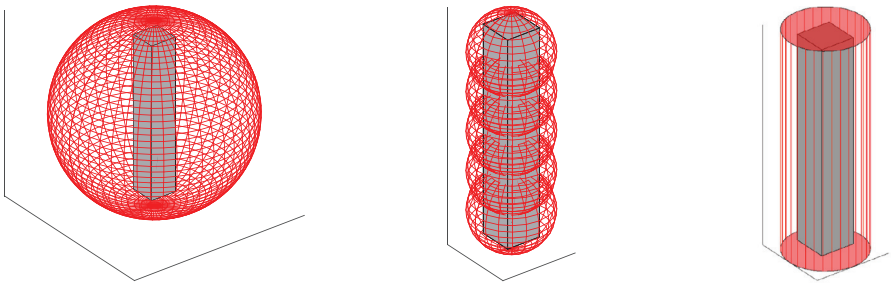
Input: σ, $\dot{\sigma}$, $\sigma_{\min}$, $\sigma_{\max}$
if $\sigma_{\min} < \sigma < \sigma_{\max}$ **then**
  return True;
else if $\sigma \leq \sigma_{\min}$ *and* $\dot{\sigma} \geq 0$ **OR** $\sigma \geq \sigma_{\max}$ *and* $\dot{\sigma} \leq 0$ **then**
  return True;
else
  return False;
end

assumption of $\boldsymbol{q} \equiv \boldsymbol{q}_{\mathrm{des}}$ [1], in mode 1 $\dot{\sigma}_\mathrm{a} = \boldsymbol{J}_\mathrm{a}\dot{\boldsymbol{q}}_{1,\mathrm{des}}$ by definition. Thus, the active mode is determined by Algorithm 2 .

Algorithm 2: Activation of modes.

a=in_T_C($\sigma_\mathrm{a}, \boldsymbol{J}_\mathrm{a}\dot{\boldsymbol{q}}_{1,\mathrm{des}}, \sigma_{\mathrm{a,min}}, \sigma_{\mathrm{a,max}}$)
if *a is True* **then**
  $\dot{\boldsymbol{q}}_{\mathrm{des}} = \dot{\boldsymbol{q}}_{1,\mathrm{des}}$ (3)
else
  $\dot{\boldsymbol{q}}_{\mathrm{des}} = \dot{\boldsymbol{q}}_{2,\mathrm{des}}$ (4)
end

Note that to apply set-based kinematic control for collision avoidance, the position and size of the obstacle must be known. In this paper, it is assumed that this information is available through a priori knowledge, e.g. sea maps, or online using sensor information, e.g. radar information. For instance, in the robotarium experiments presented in Section 9, the position and velocity of the unicycles are measured using a camera system and used continuously



(a) Single, spherical obstacle. (b) Multiple, spherical obstacles. (c) Single column-shaped obstacle.

Fig. 3. Options for set-based collision avoidance of a long, thin obstacle (red). A single, spherical set-based task as defined in (6) covers the entire obstacle, but also includes a lot of space that could be safely accessed by the robot. Several small, spherical obstacles stacked together covers the obstacle much better, but results in a more complex control system than necessary. A single, column-shaped set-based task completely covers the obstacle without also including safe space around the obstacle. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

by the set-based control system. In addition, collision avoidance will result in a deviation from the planned motion. Depending on the application, it may be necessary to limit/control this maneuver, e.g. ensure a motion which is in accordance with COLREGs for ships. The general collision avoidance algorithm described above does not consider any such limitations and the direction of motion is decided based on task 1. This approach is used in this paper for the industrial manipulator, whereas a rule-based application-specific system is applied to determine the direction of motion during collision avoidance in the USV and unicycle experiments.

Furthermore, as a kinematic control system, the set-based approach is dependent on dynamic controllers to ensure tracking of the calculated desired behavior. As the activation of modes in Algorithm 2 results in discontinuous desired system velocities, the system low-level constraints such as controller saturation will affect the overall behavior. It is possible to accommodate such constraints in the set-based control system, for instance by defining a mode change limit $> \sigma_{\mathrm{a,min}}$ to activate mode 2 before the limit $\sigma_{\mathrm{a,min}}$ and thereby preventing overshooting and violating the valid set. This approach is applied in Section 6 and 8 and results in a more conservative control system. The tuning of the mode change limit will depend on the specific application, the dynamics of the system, the operating state, actuator limitations and size and velocity of the obstacles.

3. Set-based collision avoidance for redundant, fully actuated systems

In [26] and [18], experimental results of the method described in the previous section are presented for the UR5 robotic manipulator. Here, the obstacles are defined by a position $\boldsymbol{p}_{\mathrm{o}}$ and a radius R_{s}, thus effectively being modeled as a sphere that the end effector moves around rather than enter. Hence,

$$C_{\mathrm{a}} = [R_{\mathrm{s}}, \infty) \tag{5}$$

In this case, the set-based task σ_{a} is defined as the distance between the end-effector position $\boldsymbol{p}$ and the obstacle center $\boldsymbol{p}_{\mathrm{o}}$

$$\sigma_{\mathrm{a}} = \sqrt{(\boldsymbol{p}_{\mathrm{o}} - \boldsymbol{p})^T(\boldsymbol{p}_{\mathrm{o}} - \boldsymbol{p})} \tag{6}$$

and the corresponding Jacobian is defined as in [30]. This is derived through the differentiating (6) w.r.t. time. In particular, the Jacobian relates changes in the joints $\dot{\boldsymbol{q}}$ to change in the distance $\dot{\sigma}_{\mathrm{a}}$ and depends on the current configuration $\boldsymbol{q}$.

$$\dot{\sigma}_{\mathrm{a}} = \boldsymbol{J}_{\mathrm{a}}\dot{\boldsymbol{q}} = -\frac{(\boldsymbol{p}_{\mathrm{o}} - \boldsymbol{p})^T}{||\boldsymbol{p}_{\mathrm{o}} - \boldsymbol{p}||}\boldsymbol{J}\dot{\boldsymbol{q}} \tag{7}$$

Here, $\boldsymbol{p}$ denotes the position of the end effector and $\boldsymbol{J}$ is the corresponding position Jacobian (2).

3.1. Column-shaped obstacles

Although general, a spherical shape of an obstacle as (6) is not necessarily the optimal representation for any obstacle, as illustrated in Fig. 3. For example, a spherical shape around a long, thin object would be highly conservative as a lot of safe space would be excluded from the workspace. Another option is to define multiple small spheres to avoid defining available space as non-accessible, but this would lead to a more complex control system with multiple set-based tasks, which, although feasible, is not as optimal as defining a set-based task based on a more fitting shape. The kinematics and implementation to include column-shaped obstacles in the set-based framework for redundant robotic systems are presented below. Experimental results that verify the proposed method will follow in Section 4. Note that for more complex structures, multiple columns may be defined and, if desired, combined with spherical objects. Furthermore, as most manipulators are constructed by long links, column-shaped obstacles are highly relevant in multi-agent systems to avoid self-collisions. Note that a possible alternative approach can be found in the field of animation and rendering using a sweeping sphere to represent obstacles [32].

A column-shaped obstacle may be represented by a line in $\mathbb{R}^3$ and a radius R_{s}, where the line is defined by a point $\boldsymbol{p}_{\mathrm{o}} = [x_{\mathrm{o}}, y_{\mathrm{o}}, z_{\mathrm{o}}]^T$, through which the line passes, and a unit direction vector $\boldsymbol{a} = \left[a_{\mathrm{x}}, a_{\mathrm{y}}, a_{\mathrm{z}}\right]^T$. Thus, a general line l may be parameterized by $k \geq 0$ as

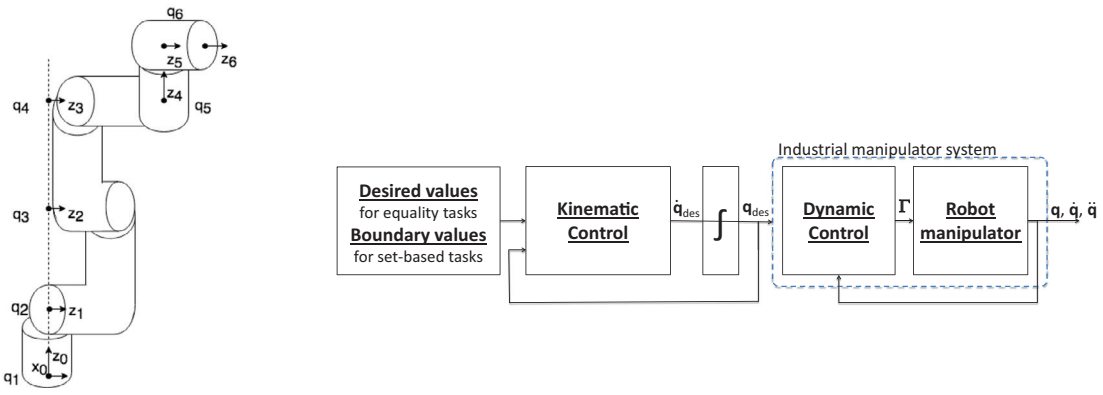
$$\boldsymbol{l}(k) := \boldsymbol{p}_{\mathrm{o}} + \boldsymbol{a}k = \begin{bmatrix} x_{\mathrm{o}} + a_{\mathrm{x}}k \\ y_{\mathrm{o}} + a_{\mathrm{y}}k \\ z_{\mathrm{o}} + a_{\mathrm{z}}k \end{bmatrix}. \tag{8}$$

Similarly to (6), it is necessary to define the set-based task σ_{a} as the distance between the position of the robotic system $\boldsymbol{p} = [x, y, z]$ and the obstacle. We define this distance as the perpendicular distance between the point $\boldsymbol{p}$ and the line $\boldsymbol{k}$. It is straightforward to show that this occurs at the point $k = \bar{k}$, where the cross-product

$$(\boldsymbol{p} - \boldsymbol{l}(\bar{k})) \cdot (\boldsymbol{l}(\bar{k}) - \boldsymbol{p_o}) = 0. \tag{9}$$

Solving (9), we obtain

$$\bar{k} = (x - x_{\mathrm{o}})a_{\mathrm{x}} + (y - y_{\mathrm{o}})a_{\mathrm{y}} + (z - z_{\mathrm{o}})a_{\mathrm{z}}, \tag{10}$$



(a) Coordinate frames corresponding to the D-H parameters in Table 1.

(b) The control structure of the experiments. The tested algorithm is implemented in the kinematic controller block.

Fig. 4. The UR5 coordinate system and control structure.

Table 1
Table of the D-H parameters of the UR5. The corresponding coordinate systems can be seen in Fig. 2a.

Joint	a_i [m]	α_i [rad]	d_i [m]	θ_i [rad]
1	0	$\pi/2$	0.089	q_1
2	−0.425	0	0	q_2
3	−0.392	0	0	q_3
4	0	$\pi/2$	0.109	q_4
5	0	$-\pi/2$	0.095	q_5
6	0	0	0.082	q_6

$$\sigma_a = \left|\boldsymbol{p} - \boldsymbol{l}(\bar{k})\right| = \sqrt{\alpha^2 + \beta^2 + \gamma^2} \tag{11}$$

$$\dot{\sigma}_a = \boldsymbol{J}_a\dot{\boldsymbol{q}} = \frac{1}{\sigma_a}(\alpha(a_y\dot{x} - a_x\dot{y}) + \beta(a_z\dot{x} - a_x\dot{z}) + \gamma(a_z\dot{y}_e - a_y\dot{z}_e)), \tag{12}$$

where

$$\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \begin{bmatrix}a_y(x - x_o) - a_x(y - y_o)\\ a_z(x - x_o) - a_x(z - z_o)\\ a_z(y - y_o) - a_y(z - z_o)\end{bmatrix}, \tag{13}$$

$$\dot{\boldsymbol{p}} = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix} = \boldsymbol{J}\dot{\boldsymbol{q}} \tag{14}$$

and $\boldsymbol{J}$ is the corresponding position Jacobian as described in (2). With the set-based task defined as above, it may be entered directly into Algorithm 1 and 2.

4. Experimental results: UR5 manipulator

4.1. UR5 Manipulator platform

The UR5 is a manipulator with 6 revolute joints denoted $\boldsymbol{q} \triangleq \begin{bmatrix}q_1 & q_2 & q_3 & q_4 & q_5 & q_6\end{bmatrix}^T$ (illustrated in Fig. 4a). In this paper, the Denavit–Hartenberg (D–H) parameters [31] are used to derive the forward kinematics. The parameters are given in Table 1.

The UR5 is equipped with a dynamic controller that can control the robot both in joint and Cartesian space. In the experiments presented here, a joint reference $\boldsymbol{q}_{des}$ is calculated in the kinematic controller and sent to the dynamic controller, which is assumed to function nominally such that $\boldsymbol{q} \approx \boldsymbol{q}_{des}$. The dynamic controller extrapolates $\dot{\boldsymbol{q}}_{des}$ and $\ddot{\boldsymbol{q}}_{des}$ based on $\boldsymbol{q}_{des}$ and uses this and the measured, actual joint angles $\boldsymbol{q}$ to control the torque in each joint to track the reference.

The structure of the system is illustrated in Fig. 4b. The set-based control system presented in Section 2 and Algorithm 1–2 is implemented. Every timestep, a reference for the joint velocities is calculated and integrated to the desired joint angles $\boldsymbol{q}_{des}$. This is used as feedback to close the kinematic loop and as input to the dynamic controller, which in turn applies torques to the joint motors. The communication between the implemented algorithm and the industrial manipulator system occurs through a TCP/IP connection which operates at 125 Hz. The algorithm itself is implemented in Python.

4.2. Implemented example

In these experiments, Algorithm 2 was implemented to determine the active mode. Three tasks make up the basis for the experiments: Position control, field of view (FOV) and collision avoidance. The analytical expressions can be found thorough the D-H parameters. The position task and Jacobian is denoted

$$\boldsymbol{\sigma}_{pos} = \boldsymbol{p} \in \mathbb{R}^3, \tag{15}$$

$$\dot{\boldsymbol{\sigma}}_{pos} = \boldsymbol{J}\dot{\boldsymbol{q}} = \frac{d\boldsymbol{p}}{d\boldsymbol{q}}\dot{\boldsymbol{q}}. \tag{16}$$

The field of view is defined as the outgoing vector of the end effector, i.e. the z_6-axis in Fig. 4a:

$$\boldsymbol{v} = \boldsymbol{g} \in \mathbb{R}^3 \tag{17}$$

$$\dot{\boldsymbol{v}} = \boldsymbol{J}_{FOV,3DOF}\dot{\boldsymbol{q}} = \frac{d\boldsymbol{g}}{d\boldsymbol{q}}\dot{\boldsymbol{q}} \tag{18}$$

FOV is a useful task when directional devices or sensors are mounted on the end-effector and they are desired to point in a certain direction $\boldsymbol{v}_{des} \in \mathbb{R}^3$. The task is defined as the norm of the error between $\boldsymbol{v}$ and $\boldsymbol{v}_{des}$:

$$\sigma_{FOV} = \sqrt{(\boldsymbol{v}_{des} - \boldsymbol{v})^T(\boldsymbol{v}_{des} - \boldsymbol{v})} \in \mathbb{R} \tag{19}$$

$$\dot{\sigma}_{FOV} = \boldsymbol{J}_{FOV}\dot{\boldsymbol{q}} = -\frac{(\boldsymbol{v}_{des} - \boldsymbol{v})^T}{\sigma_{FOV}}\boldsymbol{J}_{FOV,3DOF}\dot{\boldsymbol{q}} \tag{20}$$

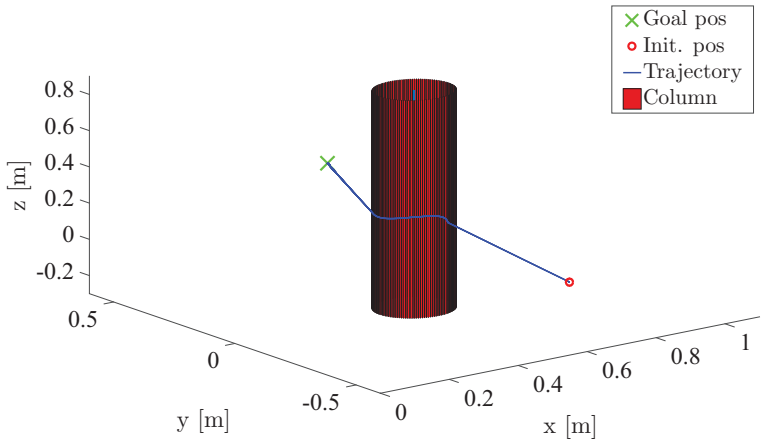


Fig. 5. Trajectory of the UR5 end effector. The robot reaches the goal position while circumventing the column-shaped obstacle.

Note that in (20), $\boldsymbol{J}_{\text{FOV}}$ is not defined for $\sigma_{\text{FOV}} = 0$. In the implementation, this is solved by adding a small constant $\epsilon > 0$ to the denominator of this Jacobian.

For this experiment, position control and FOV are stacked together to form the first and only equality task, whereas column-shaped collision avoidance as defined in (11)-(12) is the only set-based task. The numeric values are chosen as follows:

$$\boldsymbol{\sigma}_{1,\text{des}} = \begin{bmatrix} 0.55 \text{ m} & 0.40 \text{ m} & 0.30 \text{ m} & 0 \end{bmatrix}^T \tag{21}$$

$$\boldsymbol{v}_{\text{des}} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \tag{22}$$

$$\boldsymbol{\Lambda}_1 = \begin{bmatrix} 0.4 & 0 & 0 & 0 \\ 0 & 0.4 & 0 & 0 \\ 0 & 0 & 0.4 & 0 \\ 0 & 0 & 0 & 0.4 \end{bmatrix} \tag{23}$$

$$\boldsymbol{p}_{\text{o}} = \begin{bmatrix} 0.52 \text{ m} & 0.0 \text{ m} & 0.15 \text{ m} \end{bmatrix}^T \tag{24}$$

$$\boldsymbol{a} = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T \tag{25}$$

$$R_{\text{s}} = 0.10 \text{ m} \tag{26}$$

This implies that the end effector should move to the goal position of (0.55, 0.40, 0.30) m while pointing in a direction parallel to the x-axis. The tunable task gains are given by $\boldsymbol{\Lambda}_1$. Furthermore, the end effector should avoid a vertical column with radius $R_{\text{s}} = 0.10$ m and a center point $\boldsymbol{p}_{\text{o}}$. The results are shown in Figs. 5–6. The end effector reaches the goal position while circumventing the column-shaped obstacle. Mode 2, i.e. collision avoidance mode, is activated when the end effector reaches the minimum safe distance R_{s} to the column center. The distance is then locked and the end effector motion is limited to the edge of the column until mode 1 is reactivated. This occurs when the end effector has a clear path to the goal position and the tangent cone function returns `True`. Note that in this particular case, the industrial robot is able to activate mode 2 on the limit of the safe set without overshooting. Should this not be the case, a larger mode change radius as applied in Sections 6 and 8 is necessary.

5. Path following and collision avoidance for mobile robots

This section introduces the kinematic equations for mobile robots, as well as the path following and collision avoidance methods used in the set-based approaches presented in Sections 6 and 8 for a single- and multi-agent system, respectively. Note that the set-based approach is independent of path following and collision avoidance method, so other approaches may be utilized without changing the main implementation.

In this paper, we consider unicycles and USVs, which are both examples of mobile robots. The unicycles are nonholonomic mobile robots, while the USVs are second-order nonholonomic (underactuated). The state of the vehicle is given by the position $\boldsymbol{p} = (x, y)$ and heading ψ. Furthermore, the forward and sideways velocity (often denoted surge and sway for USVs) are given by u and v, and the rotational velocity by r. This is illustrated in Fig. 7a, and the kinematic equations are given as

$$\begin{aligned} \dot{x} &= \cos(\psi)u - \sin(\psi)v, \\ \dot{y} &= \sin(\psi)u + \cos(\psi)v, \\ \dot{\psi} &= r. \end{aligned} \tag{27}$$

Note that while a USV can and will glide sideways through the water during turning maneuvers, thereby having a non-zero sway v, unicycles driving on the ground are unable to do this. For unicycles, the sideways velocity $v \equiv 0$. For the remainder of this paper, equations for USVs and unicycles are presented in a unified manner including terms containing v since the unicycle model is a special case of the general model (27). For all implementations, these terms have been removed for unicycles.

As mentioned in Section 1, set-based control must be adapted to nonholonomic systems such as unicycles and USVs. For instance, a unicycle is controlled through two motors that directly affect the forward velocity and the rotational velocity. Thus, certain combinations of motions are unfeasible, such as moving directly East while the unicycle is oriented due North. Thus, the primary task of a unicycle or a surface vessel is achieved by calculating a desired orientation ψ_{des} that, if tracked, will ensure the fulfillment of that task. In this paper, path following is considered the primary task. However, if path following will lead to a collision, the vehicle should deviate from the path and perform collision avoidance. This is in contrast to the original set-based framework described in Section 2, where several tasks are defined and performed simultaneously. For underactuated, non-redundant mobile robots, it is necessary to switch between the two tasks of the system, i.e. path following and collision avoidance, rather than attempt both at once.

5.1. Path following

There exist numerous approaches to achieve path following. In this paper, a line-of-sight (LOS) approach [19] is applied. This

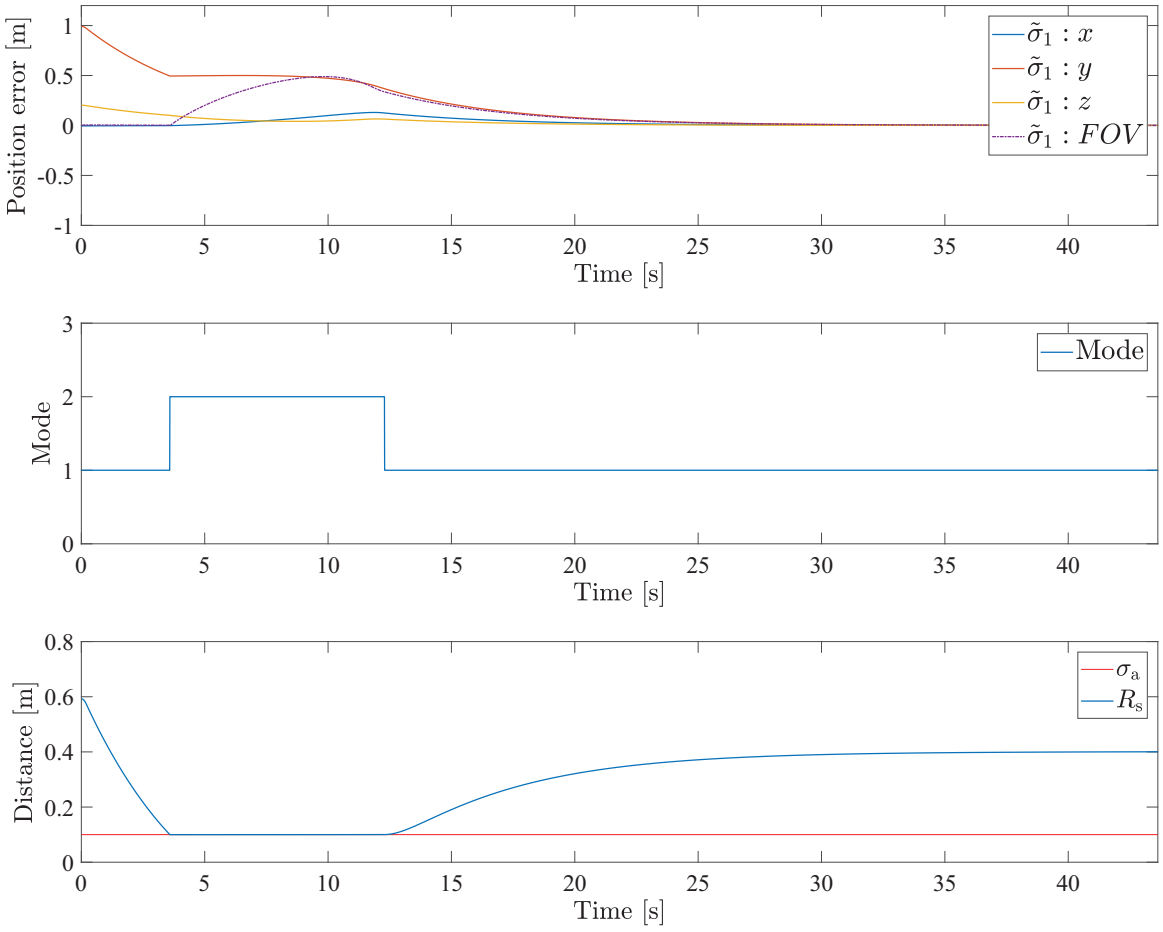
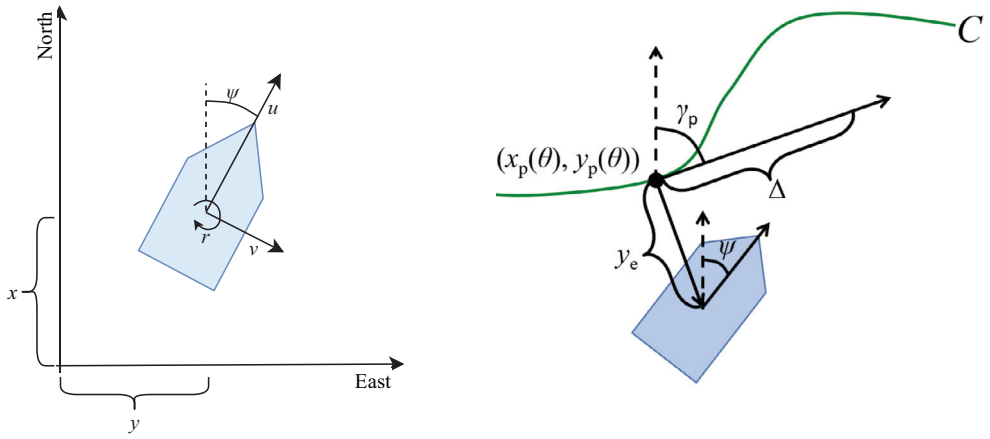


Fig. 6. Top: equality task error. Both the position and FOV task error converge to zero. Middle: active mode over time. Mode 2, i.e. collision avoidance mode, is activated as the end effector reaches the column and is deactivated when the end effector has circumvented the column and has a clear path to the goal position. Bottom: collision avoidance task. The task evolves freely in mode 1 and is controlled to the minimum value in mode 2.



(a) The reference coordinate system and states of a mobile robot.

(b) Desired path C, path-tangential reference frame with orientation $\gamma_p(\theta)$ and cross-track error y_e illustrated.

Fig. 7. System states and path following variables for mobile robots.

approach is briefly described in this section. It is proven that if the desired surge velocity $u_{des} > 0$ and heading ψ_{pf} is tracked, the agent will converge to and follow the path.

To follow a predefined path C parametrized by θ, i.e. $C = (x_p(\theta), y_p(\theta))$, the control objective is to drive the path cross-track error y_e to zero. The path parameters are illustrated in Fig. 7b. The guidance law is given below. For further details, the interested reader is referred to [33].

$$\psi_{pf} = \gamma_p(\theta) - \arctan\left(\frac{y_e}{\Delta}\right) - \underbrace{\arctan\left(\frac{v}{u_{des}}\right)}_{\triangleq \beta_{des}} \tag{28}$$

Here, γ_p is the heading of the path, $\Delta > 0$ is the look-ahead distance and v is the sway (sideways) velocity. Hence, the side-slip term β_{des} compensates for the sideways motion of the agent that occurs when turning.

5.2. Collision avoidance

To achieve collision avoidance for mobile robots, the distance between the agent and an obstacle with position $\boldsymbol{p}_o = (x_o, y_o)$ should always be greater than or equal to some safe distance R_s.

$$|\boldsymbol{p}(t) - \boldsymbol{p}_o(t)| \geq R_s \quad \forall \quad t \geq t_0 \tag{29}$$

One way to achieve this is to control the agent to track the safe radius R_s. In [27], the following guidance law giving the desired heading for collision avoidance is proposed:

$$\psi_{oa} = \phi + \lambda\left(\frac{\pi}{2} - \arctan\left(\frac{e+k}{\Delta}\right)\right) - \beta_{des} \tag{30}$$

Here, the obstacle velocity is defined as

$$U_o = \sqrt{\dot{x}_o^2 + \dot{y}_o^2}, \tag{31}$$

and

$$\phi = \arctan\left(\frac{y - y_o}{x - x_o}\right), \tag{32}$$

$$\beta_o = \arctan\left(\frac{\dot{y}_o}{\dot{x}_o}\right), \tag{33}$$

$$V_o = U_o \cos(\phi - \beta_o). \tag{34}$$

This is illustrated in Fig. 8. The velocity V_o describes the velocity of the obstacle relative to the position of the agent, where a positive V_o suggests that the obstacle is moving closer to the agent. Furthermore, the parameter $\lambda = \pm 1$ corresponds to clockwise and counter-clockwise motion, respectively. This is chosen based on agent regulations, e.g. COLREGs for USVs (see Section 6.2). Furthermore, e is the cross-track error of the circular path defined as

$$e = R_s - \sqrt{(x - x_o)^2 + (y - y_o)^2}, \tag{35}$$

and k is defined as

$$k = \begin{cases} k_1 & V_o \geq 0 \\ k_2 & V_o < 0, \end{cases} \tag{36}$$

$$k_{\{1,2\}} = \frac{-b\{+,-\}\sqrt{b^2 - 4ac}}{2a} \tag{37}$$

where

$$a = U_{des}^2 - V_o^2 = u_{des}^2 + v^2 - V_o^2, \tag{38}$$

$$b = -2V_o^2 e, \tag{39}$$

$$c = -V_o^2(\Delta^2 + e^2). \tag{40}$$

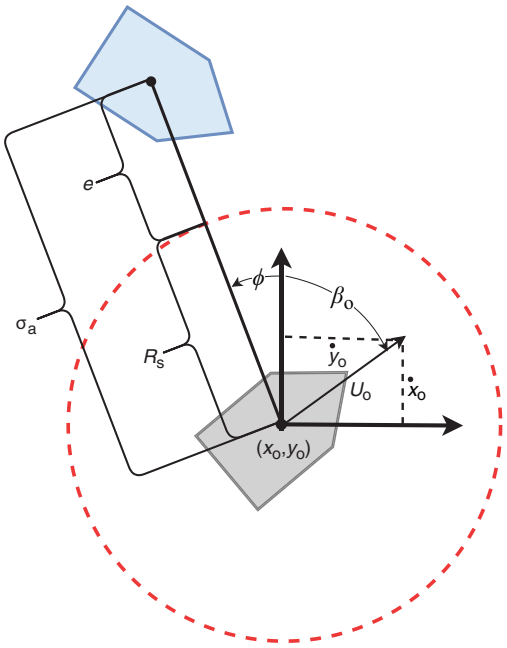


Fig. 8. Illustration of parameters used for collision avoidance for underactuated vehicles.

Note that k is a term designed to compensate for the movement of the obstacle. In case of stationary obstacles, $k = 0$. It is proven in [27] that if the desired surge velocity u_{des} and heading (30) is tracked, the agent will track a circle with a constant safe radius about the moving obstacle center.

6. Set-based collision avoidance for mobile robots: single-agent system

This section presents a set-based guidance system for a single agent which ensures collision avoidance and path following as long as this does not lead to a collision. In this paper, we consider a USV as the agent, and also suggest a method for complying with COLREGs [22]. The suggested algorithm has been implemented and validated in full-scale experiments on R/V Gunnerus (see Section 7).

In this section, the USV is considered the only controlled agent, i.e. the motion of the obstacles are independent and uncontrollable. Furthermore, we assume that the obstacles are non-overlapping, i.e. the USV encounters only one obstacle at a time. Overlapping obstacles are considered in Section 8.

6.1. Set-based guidance system

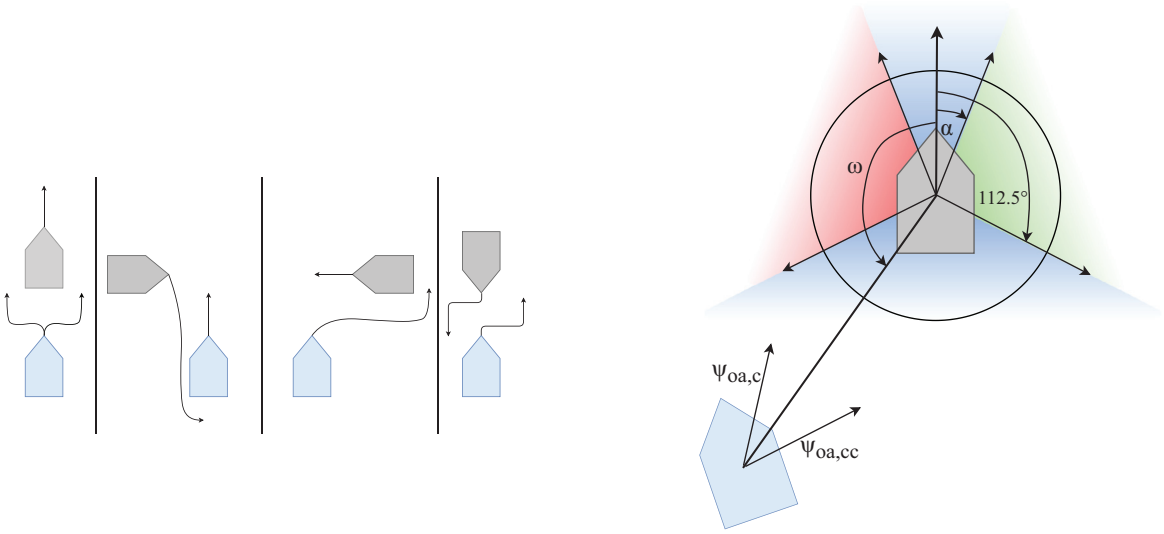
The control objective (29) can be rewritten into a set-based task as follows.

$$\sigma_a = \sqrt{(\boldsymbol{p} - \boldsymbol{p}_o)^T(\boldsymbol{p} - \boldsymbol{p}_o)}, \quad C_a = [R_s, \infty) \tag{41}$$

It is shown in [27] that

$$\dot{\sigma}_a = U\cos(\phi - \psi - \beta) - V_o, \tag{42}$$

where $U = \sqrt{u^2 + v^2}$. However, the mobile robots considered in this paper are unable to immediately change direction. Therefore, we introduce a mode change radius $R_m > R_s$ which is sufficiently large so that in case collision avoidance is activated, the agent can converge to and track R_o around the obstacle without overshoot. Therefore, obstacle avoidance should be activated if $\sigma_a \leq R_m$ and



(a) COLREGs scenarioes and the correct behavior of the involved vessels. USV in shown in blue and obstacle in orange. From left to right: Overtaking, crossing from left, crossing from right, head-on.

(b) The different COLREGs scenarios as a function of ω. From the top and clockwise: Head-on, crossing from left, overtaking, crossing from right. This specific illustration displays an overtaking situation where the motion most aligned with the current heading is chosen (in this case clockwise), given by $\psi_{\mathrm{oa,c}}$.

Fig. 9. Correct behavior of marine vehicles and the classification of COLREGs scenarios used in this paper.

Algorithm 3: Activation of modes for floating base agent encountering non-overlapping obstacles.

1 **Initialize:**
2 last_mode = path_following;
3 $\lambda = -1$;
4 **while** *True* **do**
5 a=in_T_C($\sigma_{\mathrm{a}}, U_{\mathrm{des}} \cos\left(\phi - \psi_{\mathrm{pf}} - \beta_{\mathrm{des}}\right) - V_{\mathrm{o}}, R_{\mathrm{m}}, \infty$);
6 **if** *a is True* **then**
7 $\psi_{\mathrm{des}} = \psi_{\mathrm{pf}}$; (28)
8 mode = path_following;
9 **else**
10 **if** *last_mode is path_following* **then**
11 choose λ based on agent regulations;
12 $\psi_{\mathrm{des}} = \psi_{\mathrm{oa}}(\lambda)$; (30)
13 mode = obstacle_avoidance;
14 **end**
15 last_mode = mode
16 **end**

path following would lead to the agent moving closer to the obstacle, i.e. $\dot{\sigma}_{\mathrm{a}} < 0$ given path following. This behavior is captured by the following implementation of set-based theory. For more details, the interested reader is referred to [27].

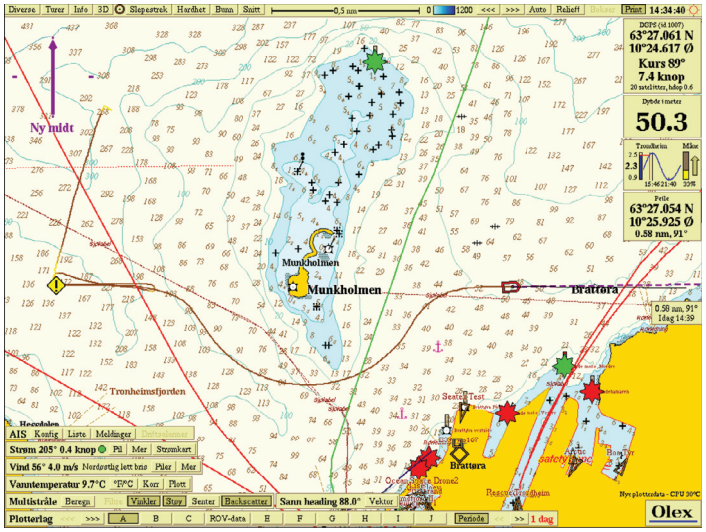
In (30), the parameter $\lambda = \pm 1$ determines the agent direction of motion about an obstacle, i.e. clockwise or counterclockwise, respectively. Note that the parameter *last_mode* in Algorithm 3 ensures that λ does not change, and thus does not create inconsequent behavior during the maneuver when the USV circumvents an obstacle.

Fig. 10. Munkholmen island in Trondheimsfjorden.

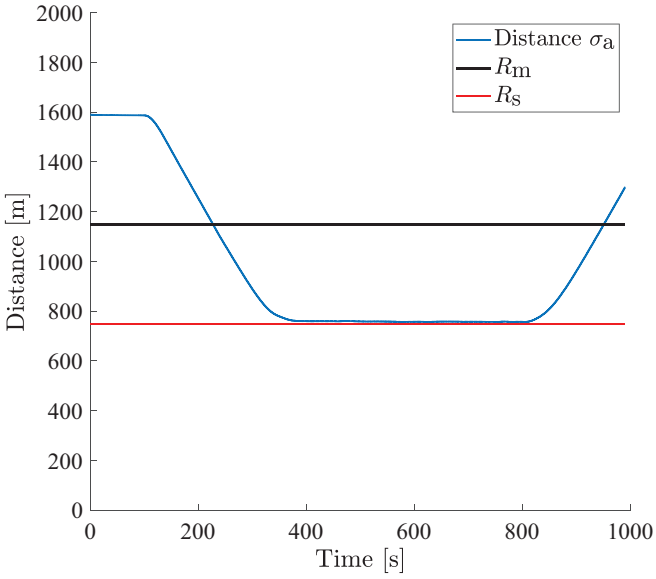
6.2. COLREGs

For USVs, the parameter λ must be set according to COLREGs in line 11 in Algorithm 3. COLREGs define the proper direction to the various collision avoidance scenarios (see Fig. 9a): overtaking, crossing from right, head-on and crossing from left [22].

In this paper, λ is chosen according to [27], where the COLREGs situation is determined based on the angle ω between the fore of the obstacle and the USV. In summary, $\lambda = -1$ for head-on, crossing from left and right, which corresponds to counter-clockwise motion. For overtaking and stationary obstacles, passing on both sides is allowed and the direction which is closest to the USV



(a) Screenshot from the onboard map Olex showing the trajectory of R/V Gunnerus starting from the marked exclamation point towards and around Munkholmen island and back to the straight line path heading east.



(b) Distance between R/V Gunnerus and the center point of Munkholmen island along with the mode change radius R_m and the safe distance R_s.

Fig. 11. Experimental results for straight line path following where the desired path goes through Munkholmen island. R/V Gunnerus activates collision avoidance as it enters the mode change radius and converges to the safe radius as it circumvents the obstacle. On the other side of the island, path following naturally leads R/V Gunnerus further away from the obstacle and is reactivated by the guidance system.

current heading and thereby will result in a less sharp turn is chosen (see Fig. 9b).

7. Experimental results: R/V Gunnerus

This section presents the on-board control system of R/V Gunnerus and experimental results for four different COLREGs scenarios. In all experiments, the set-based guidance system in Algorithm 3 with λ chosen as described in Section 6.2 was implemented. Note that this is also the first full-scale experimental verification of the path following guidance law (28).

Experiments have been conducted for four different scenarios: A stationary land obstacle, and the three COLREGs situations requiring the controlled vessel to act; head-on, overtaking and crossing from right. Note that in these experiments, the look-ahead distance Δ is different for path following guidance (28) and collision avoidance (30). A smaller Δ results in a more aggressive guidance law which is inclined to make sharper turns, but might also result in oscillatory behavior and overshooting close to the path. Therefore, a larger Δ is used for path following mode, where a smooth approach to the path is desirable, and a smaller for collision avoidance, where higher reactivity is desirable. In these particular experiments,

$$\Delta_{pf} = 150 \text{ m},$$
$$\Delta_{oa} = 100 \text{ m}. \tag{43}$$

For the case of a stationary obstacle, Munkholmen island (Fig. 10) is considered. For the remainder of the experiments,

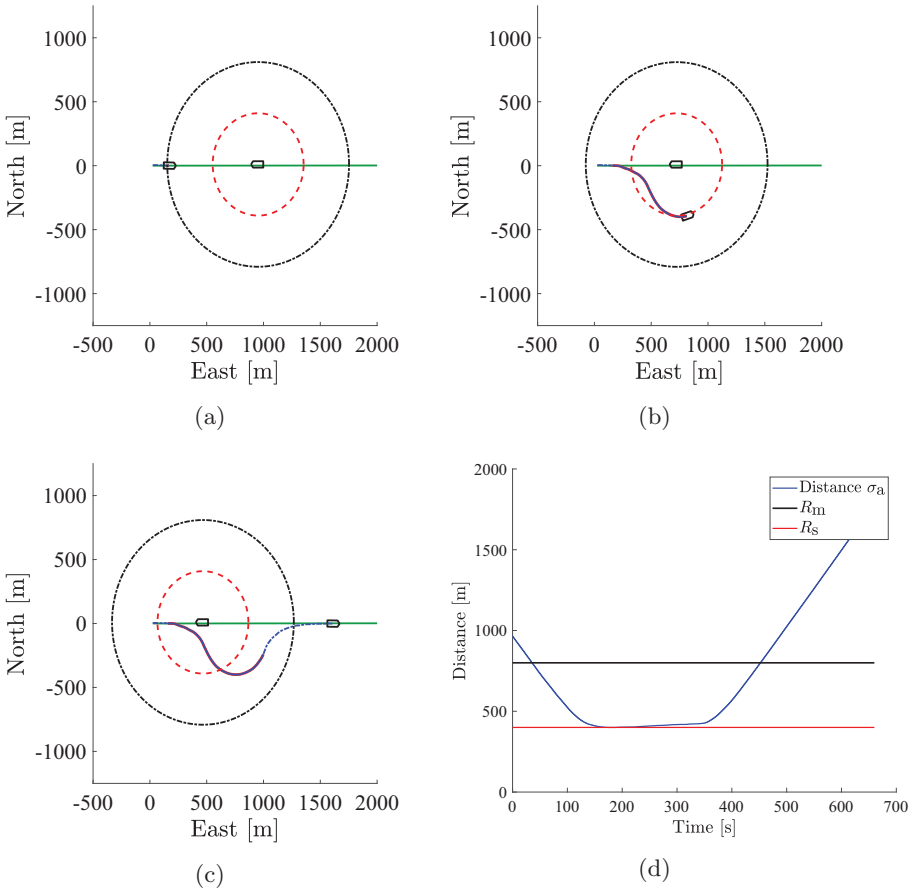


Fig. 12. Experimental results for straight line path following where another ship is on the path moving directly towards R/V Gunnerus. Collision avoidance is activated (a) as it enters the mode change radius and converges to the safe radius without overshoot ((b) and (d)) as it circumvents the moving obstacle. Eventually path following naturally leads R/V Gunnerus further away from the obstacle and is reactivated by the guidance system (c).

virtual obstacles are implemented. The virtual obstacles represent USVs which do not respond in any way to the threat of a collision. In these experiments, the following numeric values are used for the safe and mode change radius of the virtual obstacles in the head-on, crossing from right and overtaking situation.

$$R_s = 400 \text{ m}$$
$$R_m = 800 \text{ m} \tag{44}$$

These numeric values were found to be sufficiently large that R/V Gunnerus is able to converge to and track the safe radius without overshooting when activating collision avoidance mode.

Furthermore, the desired path is defined as

$$x_p(\theta) = \cos(\alpha)\theta - A\sin(\alpha)\sin(\omega\theta) + x_0, \tag{45}$$

$$y_p(\theta) = \sin(\alpha)\theta + A\cos(\alpha)\sin(\omega\theta) + y_0, \tag{46}$$

where α, A, ω, x_0 and y_0 are constant path variables. Different paths have been chosen for the different COLREGs scenarios. Numeric values for the path parameters are therefore presented in each of the following subsections.

7.1. R/V Gunnerus

R/V Gunnerus is a research vessel which is owned and operated by NTNU. The ship is steered by two azimuth thrusters and is equipped with the Kongsberg Maritime K-Pos DP-11 control system. For the purpose of these experiments, the set-based guidance Algorithm 3 thus provides heading and surge velocity references to an underlying controller in the control system. This controller is developed by Kongsberg Maritime and the details are as such unavaliable for publication. The system provides control allocation, smoothing and a mapping between the provided (u_{des}, ψ_{des}) from the guidance system to thruster force and rudder angles. Further details on R/V Gunnerus can be found in [34].

7.2. Stationary obstacle: Munkholmen island

Munkholmen island is used as a stationary obstacle. The position and size of the obstacle was determined using sea maps, and the following numeric values are chosen based on sea maps:

$$R_s = 750 \text{ m}$$
$$R_m = 1150 \text{ m} \tag{47}$$

For this experiment, the desired path is a straight line going from west to east directly through the island, just to the south of the defined obstacle center. Thus, the path variables for the path (46) are

$$\alpha = 1.57, \quad A = 0, \quad \omega = 0.0, \quad x_0 = 0, \quad y_0 = 0. \tag{48}$$

The initial position of R/V Gunnerus is on the straight line path. The results are shown in Fig. 11. R/V Gunnerus follows the desired path until it reaches the mode change radius. Clearly, continued path following would decrease the distance to the island further, so collision avoidance is activated. Stationary obstacles are treated as an overtaking situation, i.e. both a clockwise and counter-clockwise

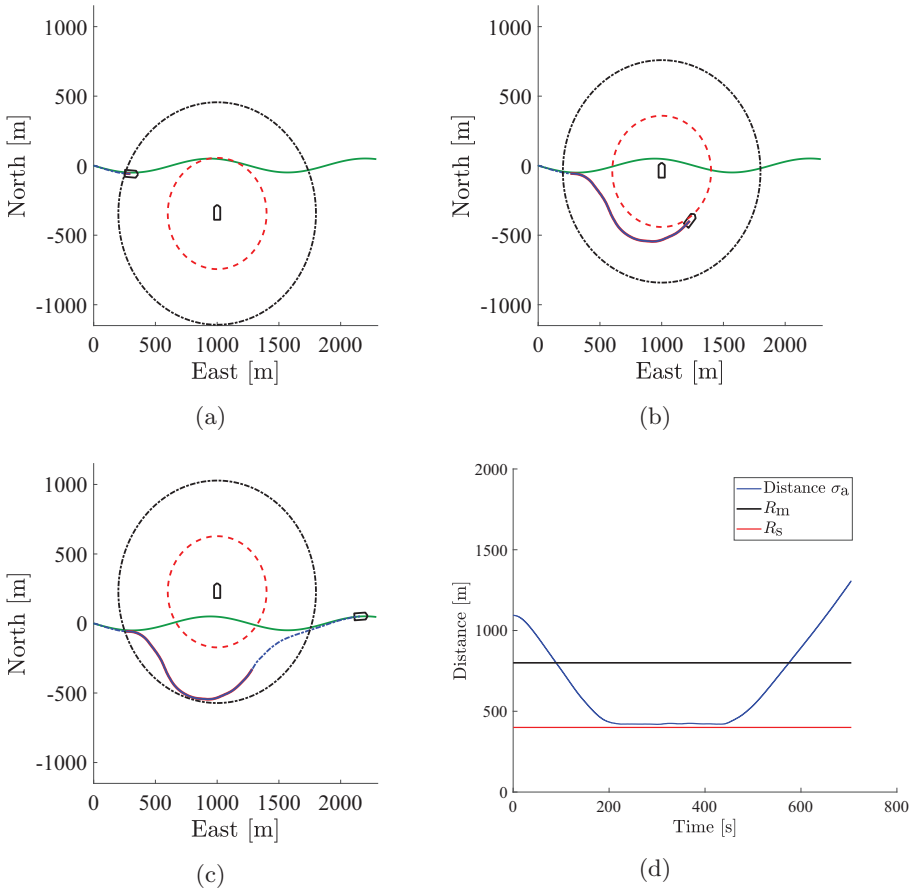


Fig. 13. Experimental results for straight line path following where another ship crosses the path from the right. Collision avoidance is activated (a) as R/V Gunnerus enters the mode change radius and converges to the safe radius without overshoot ((b) and (d)) as it circumvents the moving obstacle. Eventually path following naturally leads R/V Gunnerus further away from the obstacle and is reactivated by the guidance system (c).

motion is considered. In this case, a counterclockwise motion requires R/V Gunnerus to perform a less sharp turn as the path is slightly to the south of the obstacle center. As the island is circumvented, the path following guidance law (28) eventually leads R/V Gunnerus away from the island and towards the path. Path following is then reactivated and the ship converges back to the eastbound path. The path following guidance law (28) ensures that the ship converges to and follows the desired path.

7.3. Head-on

The chosen head-on situation is similar to the previous example. R/V Gunnerus starts on the desired straight line path heading from west to east. However, on the path is an obstacle moving directly west with a surge velocity of 1 m/s. The results are shown in Fig. 12. Similarly to the stationary obstacle, R/V Gunnerus circumvents the obstacle by converging to the safe radius R_s around the obstacle and track this until the path following guidance law (28) will result in the distance σ_a increasing, i.e. the tangent cone function in Algorithm 1 returns `True`. For head-on situations, it is especially important that the mode change radius R_m is sufficiently large since the USV must be able to make a turning maneuver and converge to R_s also when the other ship is moving towards it. This must be tuned based on the maneuverability properties of the USV in question and the speed of the obstacles.

7.4. Crossing from right

In the crossing from right situation, R/V Gunnerus is following a sinewave path (46) defined by the following path variables.

$$\alpha = 1.57, \quad A = 50, \quad \omega = 0.005, \quad x_0 = 0, \quad y_0 = 0. \tag{49}$$

Furthermore, the obstacle is heading due North with a surge velocity of 1 m/s from an initial position of $(-450, 1000)$ m relative to the initial position of R/V Gunnerus. The results are shown in Fig. 13. R/V Gunnerus follows the desired path until collision avoidance is activated. The set-based guidance system 3 correctly identifies the COLREGs situation and hence chooses a counterclockwise motion around the obstacle. The safe radius limit is never violated. In theory, R/V Gunnerus should converge to and track this radius perfectly. However, the heading reference provided by the set-based guidance system ψ_{des} is not tracked perfectly by the built-in controller due to filtering and smoothing effects, which results in a small deviation as seen in Fig. 13d.

7.5. Overtaking

With the exception of having an orientation of $\alpha = 5.75$ rad, the path variables for overtaking are the same as for the crossing from right scenario. Furthermore, the obstacle has a surge velocity of 1 m/s and a heading parallel to the path. The initial position of the obstacle is $(-650, 1100)$ m relative to R/V Gunnerus. The

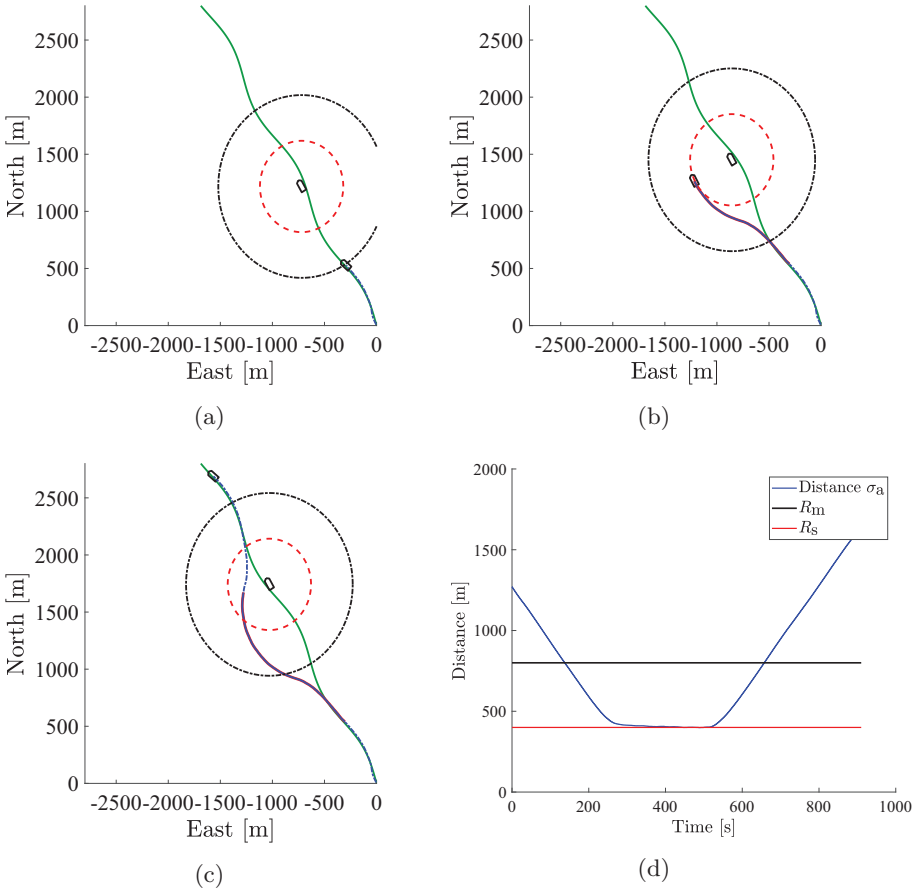


Fig. 14. Experimental results for straight line path following where another ship is moving in the same direction as R/V Gunnerus. Collision avoidance is activated (a) as R/V Gunnerus enters the mode change radius and converges to the safe radius without overshoot ((b) and (d)) as it circumvents the moving obstacle. Eventually path following naturally leads R/V Gunnerus further away from the obstacle and is reactivated by the guidance system (c).

results are shown in Fig. 14. In this case, both a clockwise and a counter-clockwise motion around the obstacle is considered as R/V Gunnerus enters the mode change radius. In this particular case, a clockwise motion requires a less sharp turn and is therefore chosen.

8. Set-based collision avoidance for mobile robots: multi-agent system

This section presents a set-based guidance system for a multi-agent system for path following and collision avoidance. The suggested algorithm has been implemented and validated in experiments on unicycle robots from the Robotarium (see Section 9). In this section, overlapping obstacles are considered, i.e. the guidance system is able to handle situations where an agent is within the mode change radius of multiple obstacles. In the experiments, all agents are controlled and have the same guidance system implemented. However, the guidance algorithm is applicable also for a single controlled agent facing multiple independent obstacles simultaneously, e.g. for a USV in a port.

8.1. Overlapping obstacles

As described above, in the nonholonomic and underactuated case of unicycles and USVs, the set-based system must activate either path following or collision avoidance. Hence, in case of multiple overlapping obstacles, it is necessary to choose not only if collision avoidance mode should be active, but also how to handle the overlapping obstacles. One conservative option is to merge overlapping obstacles into a new, larger, virtual obstacle and use this obstacle in the guidance system for collision avoidance. However, if the controlled agent is close to one or more of the actual obstacles as this merging occurs and the new, virtual obstacle is constructed, it may be within the unsafe area defined by the safe radius of the virtual obstacle which will lead to an immediate activation of collision avoidance mode. The obstacle avoidance guidance law (30) might then guide the agent directly through one of the smaller, actual obstacles on its way out of the large, virtual one. This is illustrated by the simulation in Fig. 15.

To avoid situations such as these and a highly conservative obstacle avoidance control, this paper suggests a cost function which considers all obstacles closer to the agent than their respective mode change radii and choose one of them to actively avoid. In this paper the following notation is used. Obstacles with a distance to the agent larger than the mode change radius R_m are *inactive* since according to the tangent cone they will not activate collision avoidance mode. *Engaged* obstacles are within the mode change radius and it might be necessary to activate obstacle avoidance to prevent a collision. Finally, the *active* obstacle is one of the engaged obstacles which is used to calculate ψ_{oa} as in (30) when obstacle avoidance is the active mode. This is captured by Algorithm 4 .

Note that the use of a cost function implies that we can no longer guarantee that the set-based obstacle avoidance task σ_a will be satisfied for all obstacles as in [27]. The design of the cost function determines the overall behavior of the system. For instance, COLREGs restrictions must be included for USVs. In this particular

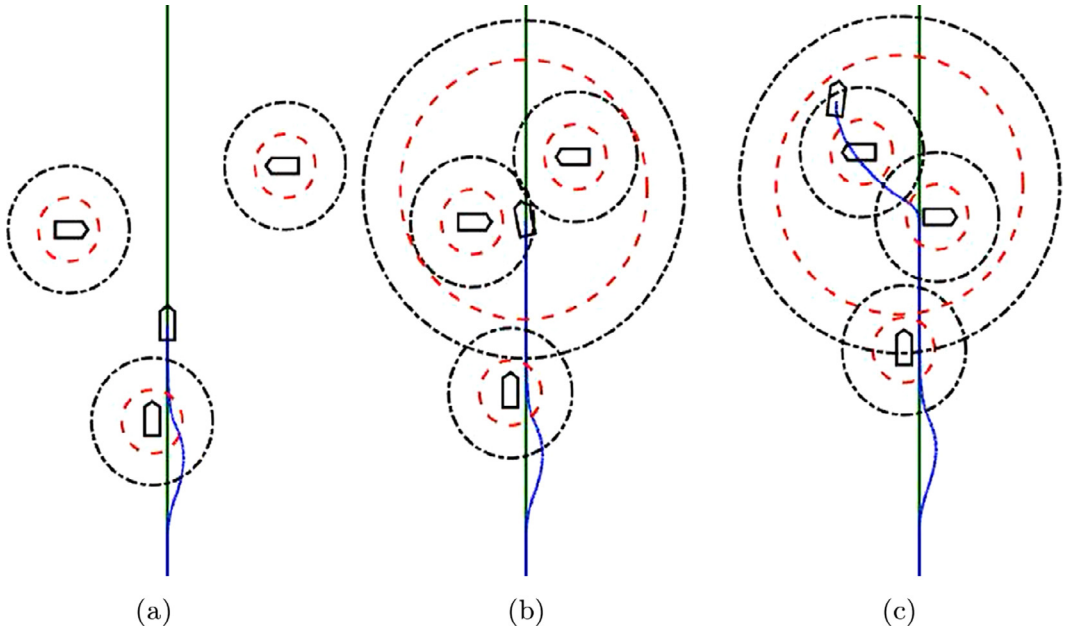


Fig. 15. Simulation: USV following a straight line path due north (green line) with three obstacles in the vicinity. Obstacle safe radius and mode change radius shown in red and black dashed circles, respectively. In this case, overlapping obstacles are handled by merging them into a new, larger virtual obstacle. This occurs in (b). However, the USV is inside the unsafe area defined by the safe radius R_s of the new, virtual obstacle at the time of merging and collision avoidance is therefore immediately activated. The desired heading of the USV is then defined by (30), which guides the USV out of the center of the new, virtual obstacle to track its safe radius R_s. However, (30) only considers the state of the virtual obstacle and not the two *actual* obstacles and might therefore guide the USV directly towards one of them. In this paper we therefore choose to use a cost function which chooses one of the engaged obstacles to actively avoid. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Algorithm 4: Handling multiple, possibly overlapping obstacles for nonholonomic mobile robots. All engaged obstacles are evaluated using the tangent cone. If collision avoidance is necessary, the active obstacle is chosen based on a cost function.

1 **Initialize agent current_agent:**
2 all_obstacles = all_obstacles < current_agent;
3 last_mode = path_following;
4 $\lambda = -1$;
5 **while** *True* **do**
6 engaged_obstacles = [];
7 **forall the** *obst* $\in$ *all_obstacles* **do**
8 **if** $\sigma_a(obst) \leq R_m$ **then**
9 engaged_obstacles.append(obst);
10 **end**
11 a = True;
12 **forall the** *obst* $\in$ *engaged_obstacles* **do**
13 a = a *and* in_T_C(σ_a(obst),$U_{des}\cos(\phi(\text{obst}) - \psi_{pf} - \beta_{des}) - V_o$(obst),$R_m, \infty$);
14 **end**
15 **if** *a is True* **then**
16 $\psi_{des} = \psi_{pf}$; (28)
17 mode = path_following
18 **else**
19 active_obstacle, λ_{cost} = minimize_cost_function(engaged_obstacles);
20 mode = active_obstacle;
21 **if** *mode is not last_mode* **then**
22 $\lambda = \lambda_{cost}$
23 $\psi_{des} = \psi_{oa}(active\_obstacle, \lambda)$; (30)
24 **end**
25 last_mode = mode;
26 **end**

paper, experiments have been performed on unicycles which are not restricted by COLREGs.

8.2. Set-based guidance system

In this paper, the N agents of the system are numbered from 1 to N. Note that line 2 in Algorithm 4 ensures that only obstacles with a lower number than the current agent, i.e. agents with higher priority, are considered for collision avoidance mode. For instance, if agent 1 is within the mode change radius of agent 2 and 3, path following is activated for agent 1. Agent 2, however, will activate collision avoidance to circumvent agent 1 if path following will decrease the distance between them further. Based on the planned motion of agent 1 and 2, agent 3 will activate path following if this will take it further away from both 1 and 2 or choose one of the higher priority agents to actively circumvent otherwise. Which one is chosen depends on the cost function. In case of a single agent in the face of many independent obstacles, such as a USV in a port, Algorithm 4 is still applicable by removing the condition of higher priority (line 2) and designing a suitable cost function.

8.3. Cost function

This section describes the defined desired behavior of a multi-agent system and the corresponding implementation of the cost function used in the experiments presented in Section 9.

If a given agent has only one engaged obstacle which must be avoided, λ will be chosen such that the agent will move behind the obstacle to avoid large detours. However, if the obstacle is stationary, both a clockwise and counter-clockwise circumvention is considered and the direction leading to the least sharp turn is chosen (corresponding to overtaking in COLREGs as described in Section 6.2).

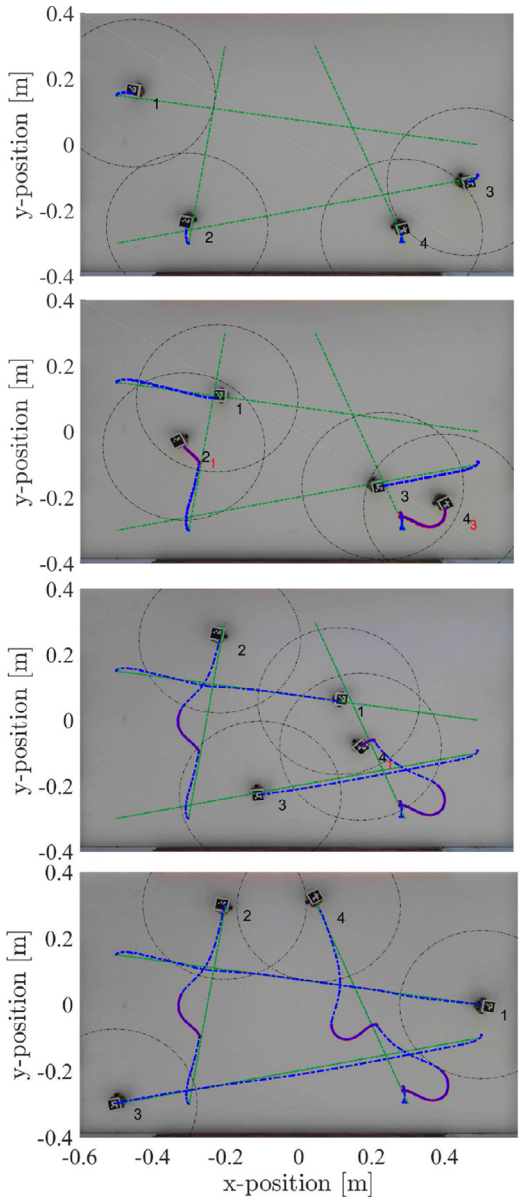


Fig. 16. Experimental results from the Robotarium with $N = 4$ agents following a straight line path. Algorithm 4 is implemented on all agents, who start on their respective paths drawn in green. Each agent is marked with its index in black and, in case of obstacle avoidance, the active obstacle in red. The mode change radius R_m is drawn in black around each agent. The agent trajectories are drawn in a blue dashed line where path following is active and in a solid line otherwise. Agent 1 is always in path following mode, since it has the right of way. Agent 2 and 4 both activate obstacle avoidance during the experiment to avoid agent 1 and agent 3 and 1, respectively. It is clear that the agents avoid collisions when necessary and converge to and follow the desired path otherwise. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

In the case of multiple engaged obstacles, the implemented cost function considers each engaged obstacle as the active obstacle, both with a clockwise and counter-clockwise motion. Each option is weighed using a cost function which considers the change in distance between the agent and the remaining engaged obstacles (should increase as much as possible, i.e. maximize $\dot{\sigma}_a$) and how close the considered motion is relative to the agent's current heading (should be as small as possible to avoid sharp turns). This is summarized in Algorithm 5 :

Algorithm 5: The implementation of the function minimize_cost_function used in the experiments presented in this paper.

Input: current_agent, engaged_obstacles
obst = engaged_obstacles[0];
if *engaged_obstacles.length() is 1* ***AND*** $U_o(obst)$ *is not 0* **then**
  Find angle ω (Fig. 9b);
  if $\omega > 0$ **then**
    $\lambda = 1$;
  else
    $\lambda = -1$;
  end
  return *obst*, λ;
else
  cost_function = [];
  forall the *obst* $\in$ *engaged_obstacles* **do**
    $\psi_{cc} = \psi_{oa}(-1)$;
    $\psi_c = \psi_{oa}(1)$;
    cost_function.append$\left(k_1 \begin{bmatrix} |\psi - \psi_{cc}| \\ |\psi - \psi_c| \end{bmatrix} - k_2 \sum_{i \in (\text{engaged_obstacles} \setminus \{\text{obst}\})} \begin{bmatrix} U_{des}\cos(\phi(i) - \psi_{cc} - \beta_{des}) - V_o(i) \\ U_{des}\cos(\phi(i) - \psi_c - \beta_{des}) - V_o(i) \end{bmatrix}\right)$;
  end
  row, column = minimum(cost_function);
  obst = engaged_obstacles[column];
  if *row is 0* **then**
    $\lambda = -1$;
  else
    $\lambda = 1$;
  end
  return *obst*, λ;
end

9. Experimental results: robotarium unicycles

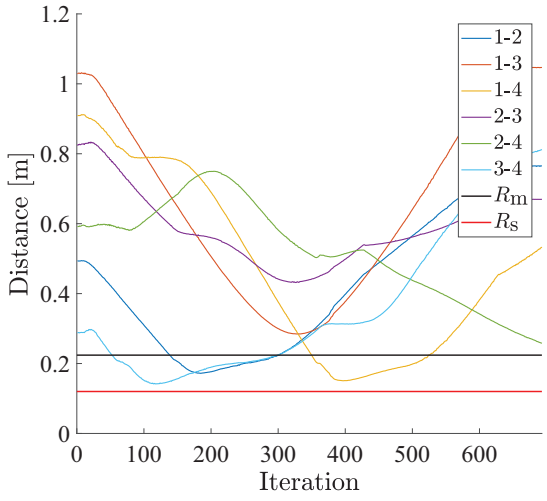
9.1. Robotarium unicycles

The Robotarium is a swarm-robotic research testbed that is accessible through a public web interface and gives users the flexibility to test a variety of multi-robot algorithms [29]. The GRITSBots miniature differential drive robots are equipped with a WiFi-enabled chip operating at 160 MHz. The position and orientation of the robots are tracked using an overhead camera system and ArUco tags. Furthermore, simulators in Matlab and Python are available for download. The simulators enable users to prototype and test their algorithms before submitting them for execution. To ensure safe operation, the Robotarium robots have a built-in collision avoidance algorithm based on barrier certificates. However, this can be bypassed if the submitted experiments achieve a sufficiently high safety score in a simulation-based verification step. The set-based collision avoidance method passed these tests, and thus all experiments presented in this paper are based only on the set-based collision avoidance methods and do not rely on other methods.

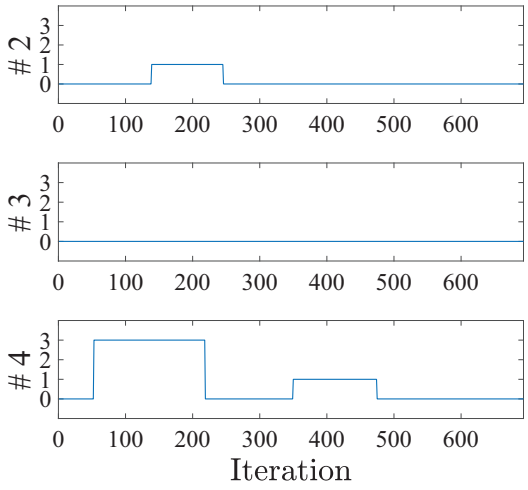
The Robotarium provides some controllers which are utilized in the experiments to follow the references given by the set-based collision avoidance framework. In particular, the set-based collision avoidance framework calculates a desired heading ψ_{des} as described in Algorithm 3. In addition, the agents should drive with a constant, positive surge velocity u_{des}. By applying (27),

$$\dot{x}_{des} = \cos(\psi_{des})u_{des}, \tag{50}$$

$$\dot{y}_{des} = \sin(\psi_{des})u_{des}. \tag{51}$$



(a) Distance between agents during experiment. The distance clearly remain above the minimum safe distance R_s.



(b) The active mode of the agents during the experiment. By default, agent 1 is always in path following mode (corresponding to zero in the plots above).

Fig. 17. Experimental results from the Robotarium with $N = 4$ agents following a straight line path. Clearly the activation of collision avoidance is concurrent with the agent distance moving below the mode change radius R_m.

Given the actual, measured pose (x, y, ψ) in addition to the calculated $(\dot{x}_{des}, \dot{y}_{des})$, the Robotarium provides a mapping to unicycle dynamics (u, r) which is imposed on the agents.

In these experiments, the numeric values are chosen as follows:

$$R_s = 0.08 \text{ m} \tag{52}$$

$$R_m = 0.22 \text{ m} \tag{53}$$

$$u_{des} = 0.08 \text{ m/s} \tag{54}$$

$$\Delta = 0.10 \text{ m} \tag{55}$$

The above numeric values were found to give the best performance in simulations and were therefore also chosen for the experiments. The cost function constants in line 15 of Algorithm 5 were chosen as $k_1 = 0.03$ and $k_2 = 1.0$. Ideally, the active obstacle and the desired heading for circumventing it should be chosen based only on how it affects the distance to the other engaged obstacles, which should increase as much as possible to avoid collisions. However, physical limitations in angular velocity might result in collisions due to slow tracking of the desired heading if this is too far from the current heading of an agent. Therefore, k_1 is given a non-zero value to consider this limitation. Finally, due to the limited size of the Robotarium board, in all experiments the agents start on their respective desired paths. Videos of the experimental results may be seen here[1].

9.2. Straight line path following

For this experiment, the $N = 4$ agents are to follow straight line paths defined by a start position $\boldsymbol{p}_{0,n} = [x_{0,n}, y_{0,n}]^T$ and goal position $\boldsymbol{p}_{x,n} = [x_{x,n}, y_{x,n}]^T$. Thus, for agent $n \in N$, the path heading γ_p is constant and defined as

$$\gamma_{p,n} = \arctan\left(\frac{y_{x,n} - y_{0,n}}{x_{x,n} - x_{0,n}}\right). \tag{56}$$

Furthermore, the cross-track error $y_{e,n}$ is defined as

$$y_{e,n} = \begin{bmatrix}\sin(\gamma_{p,n}) & \cos(\gamma_{p,n})\end{bmatrix}\begin{bmatrix}x_n - x_{0,n} \\ y_n - y_{0,n}\end{bmatrix}. \tag{57}$$

Thus, the guidance law for path following (28) simplifies into

$$\psi_{pf,n} = \gamma_{p,n} - \arctan\left(\frac{y_{e,n}}{\Delta}\right). \tag{58}$$

The experimental results are plotted in Figs. 16 and 17. As seen in Fig. 16, the agents converge to and follow their respective paths in path following mode, and are able to safely avoid collisions. Agent 1 is by default always in path following mode, and in this particular experiment, collision avoidance is not activated for agent 3. Agents 2 and 4, however, activate collision avoidance to avoid higher-priority agents and deviate from their paths. Fig. 17 confirms that the distance limit R_s is never violated.

[1] www.dropbox.com/sh/u8zt3hs64swt3b1/AADWx90A4pGI5WVtxRqdPxp7a?dl=0

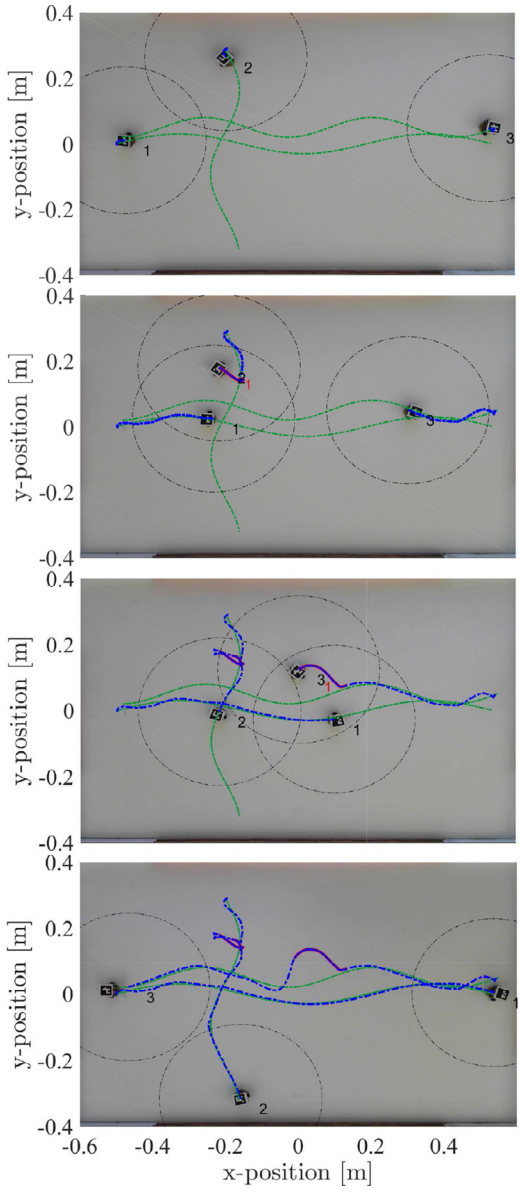


Fig. 18. Experimental results from the Robotarium with $N = 3$ agents following a curved path. This figure uses the same markup as Fig. 16.

9.3. Curved path following

This experiment considers $N = 3$ agents which should all follow a sine wave path. The path is defined as

$$x_{p,n}(\theta) = A_{x,n} \sin(\omega_{x,n}\theta) + x_{0,n} + a_n\theta, \tag{59}$$

$$y_{p,n}(\theta) = A_{y,n} \sin(\omega_{y,n}\theta) + y_{0,n} + b_n\theta, \tag{60}$$

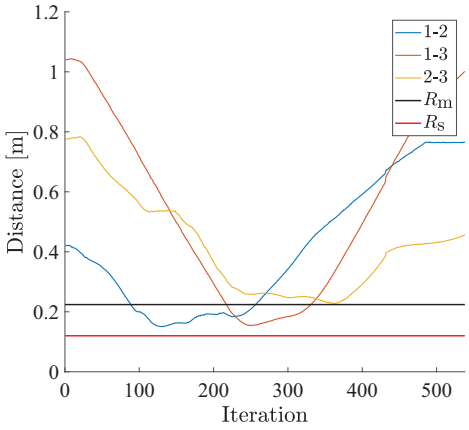
where $A_{x,n}$, $A_{y,n}$, $\omega_{x,n}$, $\omega_{y,n}$, a_n and b_n are path constants for agent n. For these experiments, the following numeric values were chosen:

$$\begin{array}{cccc} A_{x,1} = 0.0 & \omega_{x,1} = 0.0 & x_{0,1} = -0.5 & a_1 = 1.1 \\ A_{y,1} = 0.03 & \omega_{y,1} = 10.0 & y_{0,1} = 0.0 & b_1 = 0.0 \\ A_{x,2} = 0.04 & \omega_{x,2} = 8.0 & x_{0,2} = -0.2 & a_2 = 0.0 \\ A_{y,2} = 0.0 & \omega_{y,2} = 0.0 & y_{0,2} = 0.29 & b_2 = -0.65 \\ A_{x,3} = 0.0 & \omega_{x,3} = 0.0 & x_{0,3} = 0.55 & a_3 = -1.1 \\ A_{y,3} = -0.03 & \omega_{y,3} = 15.0 & y_{0,3} = 0.05 & b_3 = 0.0 \end{array} \tag{61}$$

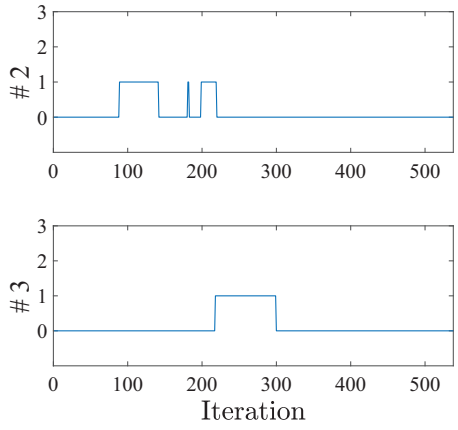
This corresponds to agent 1 moving from left to right, agent 2 moving from top to bottom and agent 3 moving from right to left in a sine-wave with different amplitudes and frequency. As seen in Fig. 18, the agents are able to follow their desired path with reasonable accuracy. Deviations are mostly due to the calculated heading reference ψ_{des} not being perfectly tracked by the controller. As seen in the straight line experiment, the agents avoid collisions by circumventing higher priority agents if necessary. Fig. 19 shows that agent 2 activates collision avoidance to avoid agent 1 several times during the experiment. This is to be expected since both agents' paths have curves so the distance between these agents may naturally increase and decrease several times. When the agents are closer to each other than the mode change radius R_{m} and this occurs, this will lead to several mode changes. However, the control system is able to handle this well and the agents exhibit good and safe behavior.

10. Conclusions

Set-based collision avoidance is a highly generic approach which can be adapted and applied to numerous different robotic systems. In this paper, we consider both a fully actuated robotic manipulator, a USV and multi-agent systems consisting of unicycles. For the robotic manipulator, we propose a method for defining column-shaped obstacles as an alternative to the previously suggested spherical obstacles [18]. The method is implemented and experimentally verified using a UR5 manipulator which is given an additional task of reaching a goal position while pointing in a specified direction. A method for set-based collision avoidance for a USV is also presented. In this case, the USV is underactuated and should follow a predefined path. The control system activates collision avoidance when necessary to avoid collisions with both stationary and dynamic obstacles in a COLREGs compliant manner, and ensures convergence to and following of the desired path otherwise. This method has been experimentally tested and verified full-scale on R/V Gunnerus in Trondheimsfjorden. Finally, a similar approach is suggested for a multi-agent system of mobile base robots which also considers the possibility of overlapping obstacles. It is suggested to define a hierarchy amongst the agents where the lower-priority agents are responsible for avoiding collisions with higher-priority agents. To handle the overlapping obstacles, a cost function is suggested. The method is experimentally verified using the Robotarium unicycles.



(a) Distance between agents during experiment. The distance clearly remain above the minimum safe distance R_m.



(b) The active mode of the agents during experiment. By default, agent 1 is always in path following mode (corresponding to zero in the plot above).

Fig. 19. Experimental results from the Robotarium with $N = 3$ agents following a curved path. Clearly the activation of collision avoidance is concurrent with the agent distance moving below the mode change radius R_m.

Declaration of Competing Interest

The authors whose names are listed immediately below certify that they have NO affiliations with or involvement in any organization or entity with any financial interest or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

CRediT authorship contribution statement

Signe Moe: Conceptualization, Methodology, Software, Investigation, Writing - original draft. **Kristin Y. Pettersen:** Conceptualization, Methodology, Writing - review & editing, Supervision, Funding acquisition. **Jan Tommy Gravdahl:** Conceptualization, Methodology, Writing - review & editing, Supervision, Funding acquisition.

Acknowledgments

The authors would like to thank Martin Syre Wiig for a great collaboration in preparing and conducting the R/V Gunnerus trials, in addition to Øystein Lurås, Inge Spangelo and Bjoern Ole Vinje at Kongsberg Maritime for providing access to their control system API and for their invaluable support before and during the experiments, and the crew aboard the R/V Gunnerus. Furthermore, the Robotarium provided excellent assistance and wonderful execution of the mobile robot experiments. This work was supported by the Research Council of Norway in part through the Center of Excellence funding scheme, project number 223254, and in part through the Center for research based innovation SFI Manufacturing, project number 237900.

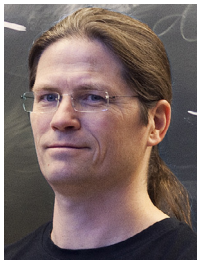
References

[1] Antonelli G. Stability analysis for prioritized closed-Loop inverse kinematic algorithms for redundant robotic systems. IEEE Trans Rob 2009;25(5):985–94. doi:10.1109/TRO.2009.2017135.
[2] Caccavale F, Siciliano B. Kinematic control of redundant free-floating robotic systems. Adv Rob 2001;15(4):429–48. doi:10.1163/156855301750398347.
[3] Egeland O, Pettersen KY. Free-floating robotic systems. In: Control problems in robotics and automation, 230. Springer Berlin Heidelberg; 1998. p. 119–134.
[4] Nenchev D, Umetani Y, Kazuya Y. Analysis of a redundant free-flying spacecraft/manipulator system. IEEE Trans Robot Autom 1992;8(1):1–6. doi:10.1109/70.127234.
[5] Buss SR. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods; 2009. https://www.math.ucsd.edu/sbuss/ResearchWeb/ikmethods/iksurvey.pdf
[6] Klein CA, Huang C. Review of pseudoinverse control for use with kinematically redundant manipulators. IEEE Trans Syst Man Cybern 1983;SMC-13(2):245–50. doi:10.1109/TSMC.1983.6313123.
[7] . Springer handbook of robotics. Siciliano B, Khatib O, editors. Springer Berlin Heidelberg; 2008. ISBN 978-3-540-23957-4. doi:10.1007/978-3-540-30301-5.
[8] Siciliano B, Sciavicco L, Villani L, Oriolo G. Robotics: modelling, planning and control. Springer Verlag; 2009.
[9] Antonelli G, Arrichiello F, Chiaverini S. The null-space-based behavioral control for autonomous robotic systems. Intell Serv Robot 2008;1(1):27–39. doi:10.1007/s11370-007-0002-3.
[10] Arrichiello F, Chiaverini S, Indiveri G, Pedone P. The null-space-based behavioral control for mobile robots with velocity actuator saturations. Int J Rob Res 2010;29(10):1317–37.
[11] Shiyou D, Xiaoping Z, Guoqing L. The Null-Space-Based Behavioral Control for Swarm Unmanned Aerial Vehicles. In: Proceedings 2011 first international conference on instrumentation, measurement, computer, communication and control. Beijing, China: IEEE; 2011. p. 1003–6. ISBN 978-0-7695-4519-6. doi:10.1109/IMCCC.2011.253.
[12] Herrera D, Monllor M, Santiago D, Roberti F, Carelli R. Null-space based control for human following and social field avoidance. In: Proceedings of the 2017 XVII workshop on information processing and control (RPIC); 2017. p. 1–6.
[13] Marchand E, Chaumette F, Rizzo A. Using the task function approach to avoid robot joint limits and kinematic singularities in visual servoing. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, 3; 1996. p. 1083–90. ISBN 0-7803-3213-X. doi:10.1109/IROS.1996.568954.
[14] Hanafusa H, Yoshikawa T, Nakamura Y. Analysis and control of articulated robot arms with redundancy. In: Proceedings of the 8th IFAC world congress; 1981.
[15] Kanoun O, Lamiraux F, Wieber PB. Kinematic control of redundant manipulators: generalizing the task-priority framework to inequality task. IEEE Trans Robot 2011;27(4):785–92.
[16] Faverjon B, Tournassoud P. A local based approach for path planning of manipulators with a high number of degrees of freedom. In: Proceedings of the EEE international conference on robotics and automation, 4; 1987. p. 1152–9.
[17] Khatib O. The potential field approach and operational space formulation in robot control. Adaptive and learning systems: theory and applications. Narendra KS, editor. Springer US; 1986.
[18] Moe S, Antonelli G, Teel AR, Pettersen KY, Schrimpf J. Set-Based tasks within the singularity-Robust multiple task-Priority inverse kinematics framework: general formulation, stability analysis, and experimental results. Front Robot AI 2016;3(April):1–18. doi:10.3389/frobt.2016.00016.
[19] Fossen TI. Handbook of marine craft hydrodynamics and motion control. Wiley; 2011. ISBN 9781119991496.
[20] Khatib O. Real-time obstacle avoidance for manipulators and mobile robots. In: Proceedings of the IEEE international conference on robotics and automation, 2; 1985. p. 500–5. doi:10.1109/ROBOT.1985.1087247.

[21] Fox D, Burgard W, Thrun S. The dynamic window approach to collision avoidance. Robot Autom Mag, IEEE 1997;4(1):23–33. doi:10.1109/100.580977.
[22] Kuwata Y, Wolf MT, Zarzhitsky D, Huntsberger TL. Safe maritime autonomous navigation with COLREGS, using velocity obstacles. IEEE J Oceanic Eng 2014;39(1):110–19.
[23] Koren Y, Borenstein J. Potential field methods and their inherent limitations for mobile robot navigation. In: Proceedings of the IEEE international conference on robotics and automation; 1991. p. 1398–404. ISBN 0-8186-2163-X. doi:10.1109/ROBOT.1991.131810.
[24] Wiig MS, Pettersen KY, Krogstad TR. Collision avoidance for underactuated marine vehicles using the constant avoidance angle algorithm. Accepted to IEEE Trans Control Syst Technol 2018. www.dropbox.com/sh/qglvjuhvrhqxqd0/AABeukMnCQhVYRllTix2jsBMa
[25] Wiig MS, Pettersen KY, Krogstad TR. A 3d reactive collision avoidance algorithm for underactuated underwater vehicles. Submitted to J Field Robot 2018. www.dropbox.com/s/d3u5wjh7d4vclii
[26] Moe S, Antonelli G, Pettersen KY, Schrimpf J. Experimental results for set-based control within the singularity-robust multiple task-priority inverse kinematics framework. In: Proceedings of the IEEE international conference on robotics and biomimetics; 2015. Zhuhai, China
[27] Moe S, Pettersen KY. Set-based line-of-sight (LOS) path following with collision avoidance for underactuated unmanned surface vessel. In: Proceedings of the 24th mediterranean conference on control and automation; 2016. Athens, Greece
[28] Moe S, Pettersen KY. Set-based line-of-sight (LOS) path following with collision avoidance for underactuated unmanned surface vessels under the influence of ocean currents. In: Proceedings of the 2017 IEEE conference on control technology and applications (CCTA); 2017. Hawaii, USA
[29] Pickem D, Glotfelter P, Wang L, Mote M, Ames A, Feron E, et al. The robotarium: a remotely accessible swarm robotics research testbed. In: Proceedings of the 2017 IEEE international conference on robotics and automation (ICRA); 2017. p. 1699–706. doi:10.1109/ICRA.2017.7989200. Singapore
[30] Antonelli G. Underwater robots. Springer Tracts in Advanced Robotics, 96. Springer International Publishing; 2014. ISBN 978-3-319-02876-7. doi:10.1007/978-3-319-02877-4.
[31] Spong MW, Hutchinson S. Robot modeling and control. Wiley; 2005. ISBN 9780471649908.
[32] Bertails F, Hadap S, Cani M, Lin M, Marschner S, Kacic-Alesic Z, Ward K. Realistic hair simulation - Animation and rendering. ACM SIGGRAPH 2008 class notes; 2008. doi:10.1145/1401132.1401247.
[33] Fossen TI, Pettersen KY. On uniform semiglobal exponential stability (USGES) of proportional line-of-sight guidance laws. Automatica 2014;50(11):2912–17. doi:10.1016/j.automatica.2014.10.018.
[34] Skjetne R, Sorensen MEN, Breivik M, Værno SAT, Brodtkorb AH, Sorensen AJ, et al. AMOS DP research cruise 2016: academic full-scale testing of experimental dynamic positioning control algorithms onboard R/V gunnerus. In: Proceedings of the international conference on offshore mechanics and arctic engineering, Volume 1: offshore technology; 2017.

Signe Moe received the M.Sc. degree in engineering cybernetics, specializing in guidance, navigation, and control, from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 2013, and the Ph.D. degree from the Center for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, NTNU, with a thesis entitled Guidance and Control of Robot Manipulators and Autonomous Marine Robots. She is currently a Post-Doctoral Fellow associated with the Center for research-based innovation SFI Manufacturing, where her research primarily revolves around control of industrial robotic systems. In addition she holds a position as a researcher at SINTEF Digital, Dept. of Mathematics and Cybernetics, where she works in the group for Analytics and AI on industrial AI.

Kristin Y. Pettersen is a Professor in the Department of Engineering Cybernetics, NTNU where she has been a faculty member since 1996. She was Head of Department 2011–2013, Vice-Head of Department 2009–2011, and Director of the NTNU ICT Program of Robotics 2010–2013. She is Adjunct Professor at the Norwegian Defence Research Establishment (FFI). In the period 2013–2022 she is also Key Scientist at the CoE Centre for Autonomous Marine Operations and Systems (NTNU AMOS). She is a co-founder of the NTNU spin-off company Eelume AS, where she was CEO 2015–2016. She received the MSc and PhD degrees in Engineering Cybernetics at NTNU, Trondheim, Norway, in 1992 and 1996, respectively. She has published four books and more than 250 papers in international journals and refereed conferences. Her research interests focus on nonlinear control of mechanical systems with applications to robotics, with a special emphasis on marine robotics and snake robotics. She was awarded the IEEE Transactions on Control Systems Technology Outstanding Paper Award in 2006 and in 2017. She was a nominated and elected member of the Board of Governors of IEEE Control Systems Society 2012–2014 and is currently a member of the IFAC Council. She has also held and holds several board positions in industrial and research companies. She was Program Chair of the IEEE Conference on Control Technology and Applications in 2018, and has served as Associate Editor for several IEEE CSS conferences. She has served as Associate Editor of IEEE Transactions on Control Systems Technology and IEEE Control Systems Magazine, and is Senior Editor of Transactions on Control Systems Technology. She is IEEE CSS Distinguished Lecturer 2019–2021. She is an IEEE Fellow, member of the Norwegian Academy of Technological Sciences, and member of the Academy of the Royal Norwegian Society of Sciences and Letters.



Jan Tommy Gravdahl received the Siv.ing and Dr.ing degrees in Engineering Cybernetics from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 1994 and 1998, respectively. He is since 2005 professor at the Department of Engineering Cybernetics, NTNU, where he also served as Head of Department in 2008-09. He has supervised the graduation of 127 MSc and 14 PhD candidates. He has published five books and more than 250 papers in international conferences and journals. In 2000 and again in 2017, he was awarded the IEEE Transactions on Control Systems Technology Outstanding Paper. He is since 2017 senior editor of the IFAC journal Mechatronics. His current research interests include mathematical modeling and nonlinear control in general, in particular applied to turbomachinery, marine vehicles, spacecraft, robots, and high-precision mechatronic systems.