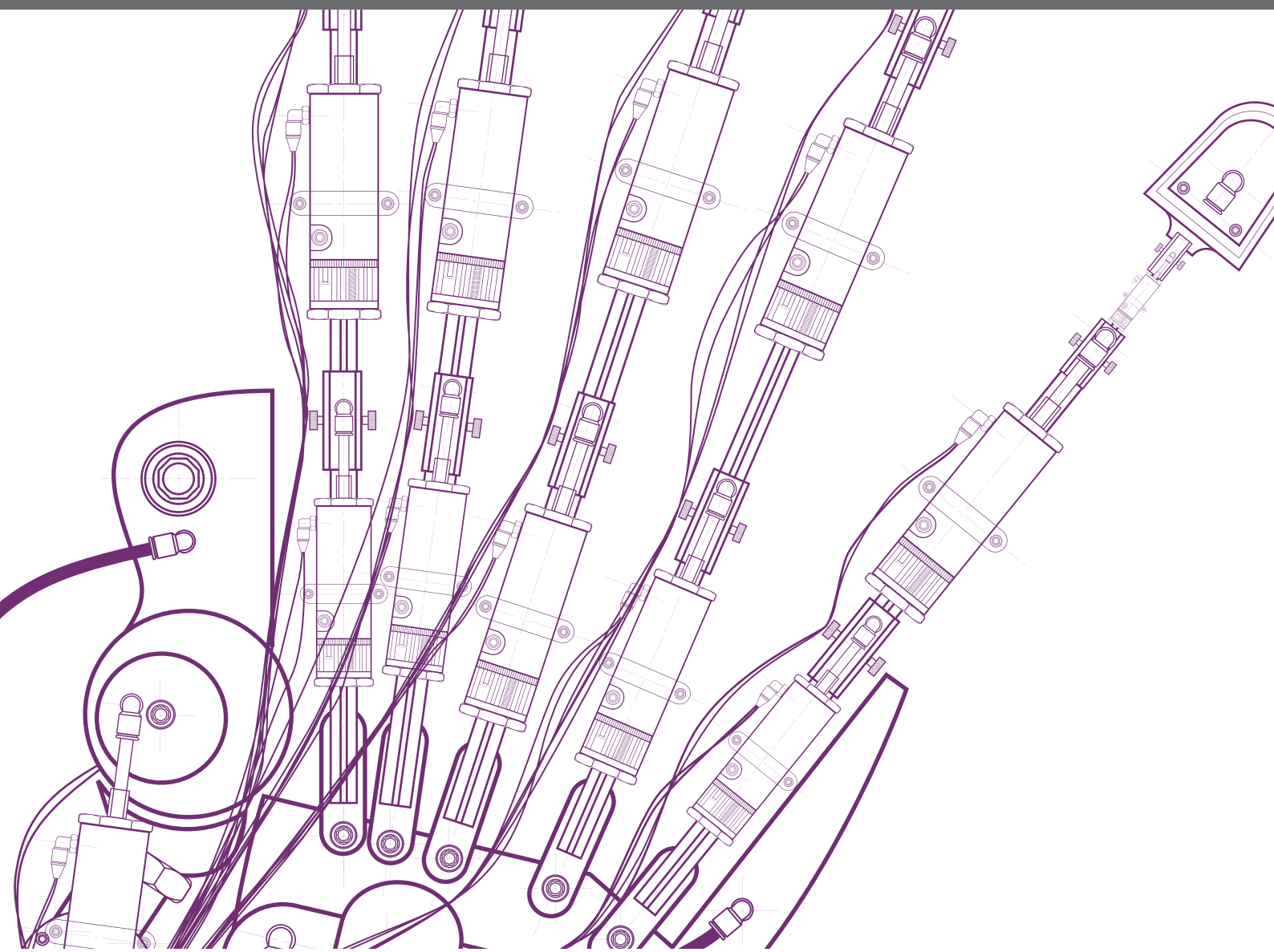
ADVANCED CONTROL METHODS IN MARINE ROBOTICS APPLICATIONS

EDITED BY: Fabio Bonsignorio, Enrica Zereik, Marco Bibuli, Kristin Ytterstad Pettersen and Oussama Khatib

PUBLISHED IN: Frontiers in Robotics and AI



frontiers Research Topics

frontiers

ISSN 1664-8714
ISBN 978-2-88966-872-4
DOI 10.3389/978-2-88966-872-4

About Frontiers

Frontiers is more than just an open-access publisher of scholarly articles: it is a pioneering approach to the world of academia, radically improving the way scholarly research is managed. The grand vision of Frontiers is a world where all people have an equal opportunity to seek, share and generate knowledge. Frontiers provides immediate and permanent online open access to all its publications, but this alone is not enough to realize our grand goals.

Frontiers Journal Series

The Frontiers Journal Series is a multi-tier and interdisciplinary set of open-access, online journals, promising a paradigm shift from the current review, selection and dissemination processes in academic publishing. All Frontiers journals are driven by researchers for researchers; therefore, they constitute a service to the scholarly community. At the same time, the Frontiers Journal Series operates on a revolutionary invention, the tiered publishing system, initially addressing specific communities of scholars, and gradually climbing up to broader public understanding, thus serving the interests of the lay society, too.

Dedication to Quality

Each Frontiers article is a landmark of the highest quality, thanks to genuinely collaborative interactions between authors and review editors, who include some of the world's best academicians. Research must be certified by peers before entering a stream of knowledge that may eventually reach the public - and shape society; therefore, Frontiers only applies the most rigorous and unbiased reviews.
Frontiers revolutionizes research publishing by freely delivering the most outstanding research, evaluated with no bias from both the academic and social point of view. By applying the most advanced information technologies, Frontiers is catapulting scholarly publishing into a new generation.

What are Frontiers Research Topics?

Frontiers Research Topics are very popular trademarks of the Frontiers Journals Series: they are collections of at least ten articles, all centered on a particular subject. With their unique mix of varied contributions from Original Research to Review Articles, Frontiers Research Topics unify the most influential researchers, the latest key findings and historical advances in a hot research area! Find out more on how to host your own Frontiers Research Topic or contribute to one as an author by contacting the Frontiers Editorial Office: frontiersin.org/about/contact

ADVANCED CONTROL METHODS IN MARINE ROBOTICS APPLICATIONS

Topic Editors:
Fabio Bonsignorio, Heron Robots, Italy
Enrica Zereik, National Research Council (CNR), Italy
Marco Bibuli, National Research Council (CNR), Italy
Kristin Ytterstad Pettersen, Norwegian University of Science and Technology, Norway
Oussama Khatib, Stanford University, United States

Citation: Bonsignorio, F., Zereik, E., Bibuli, M., Pettersen, K. Y., Khatib, O., eds. (2021). Advanced Control Methods in Marine Robotics Applications. Lausanne: Frontiers Media SA. doi: 10.3389/978-2-88966-872-4

Table of Contents

04 ***Editorial: Advanced Control Methods in Marine Robotics Applications***
Fabio Bonsignorio, Enrica Zereik, Marco Bibuli, Kristin Y. Pettersen and Oussama Khatib

06 ***Path Following, Obstacle Detection and Obstacle Avoidance for Thrusted Underwater Snake Robots***
Eleni Kelasidi, Signe Moe, Kristin. Y. Pettersen, Anna M. Kohl, Pål Liljebäck and Jan Tommy Gravdahl

21 ***Robust Cooperative Moving Path Following Control for Marine Robotic Vehicles***
Matheus F. Reis, R. Praveen Jain, A. Pedro Aguiar and Joao Borges de Sousa

34 ***Marine Applications of the Fast Marching Method***
Santiago Garrido, David Alvarez and Luis E. Moreno

46 ***Underwater Robotics Competitions: The European Robotics League Emergency Robots Experience With FeelHippo AUV***
Matteo Franchi, Francesco Fanelli, Matteo Bianchi, Alessandro Ridolfi and Benedetto Allotta

56 ***Hybrid Collision Avoidance for ASVs Compliant With COLREGs Rules 8 and 13–17***
Bjørn-Olav H. Eriksen, Glenn Bitar, Morten Breivik and Anastasios M. Lekkas

74 ***Reinforcement Learning-Based Tracking Control of USVs in Varying Operational Conditions***
Andreas B. Martinsen, Anastasios M. Lekkas, Sébastien Gros, Jon Arne Glomsrud and Tom Arne Pedersen

90 ***Trustable Environmental Monitoring by Means of Sensors Networks on Swarming Autonomous Marine Vessels and Distributed Ledger Technology***
Ivan Berman, Enrica Zereik, Aleksandr Kapitonov, Fabio Bonsignorio, Alisher Khassanov, Aziza Oripova, Sergei Lonshakov and Vitaly Bulatov

113 ***Interoperability Among Unmanned Maritime Vehicles: Review and First In-field Experimentation***
Riccardo Costanzi, Davide Fenucci, Vincenzo Manzari, Michele Micheli, Luca Morlando, Daniele Terracciano, Andrea Caiti, Mirko Stifani and Alessandra Tesei

128 ***Control Protocols for Range-Based Navigation of a Networked Group of Underwater Vehicles***
Daniela De Palma, Giovanni Indiveri and Gianfranco Parlangeli

frontiers in Robotics and AI

EDITORIAL
published: 15 April 2021
doi: 10.3389/frobt.2021.654581

Editorial: Advanced Control Methods in Marine Robotics Applications

Fabio Bonsignorio[1], Enrica Zereik[2], Marco Bibuli[2], Kristin Y. Pettersen[3] and Oussama Khatib[4]*

[1]Heron Robots, Genoa, Italy, [2]Institute of Marine Engineering, Italian National Research Council, Genoa, Italy, [3]Department of Engineering Cybernetics, Centre for Autonomous Marine Operations and Systems, Norwegian University of Science and Technology, Trondheim, Norway, [4]Robotics Laboratory, Department of Computer Science, Stanford University, Stanford, CA, United States

Keywords: advanced control, control theory, control engineering, marine robotics, underwater robotics

Editorial on the Research Topic

Advanced Control Methods in Marine Robotics Applications

1 INTRODUCTION

This article collection aims to provide a set of reports that could be used to better understand the state of the art and trigger a wide-range discussion among researchers and practitioners active in marine robotics and researchers active in robotics applications of control theory and engineering. This research topic originated from the workshop "Hand-Shaking Advanced Control in Marine Robotics Applications" held at IROS 2018. However, contributions have been collected through an open call.

As a result, we have a diverse set of contributions in which authors deal with current challenges in marine robotics; in particular issues and solutions related to control problems for systems interacting with the environment, advanced control of systems with uncertain dynamics working within uncertain environments, classical control with prioritized tasks for object avoidance, cooperative motion coordination, biomimetic approaches to formation control, etc. We did not receive submissions on more research-oriented topics like "embodiment," "morphological computation," "orchestration control," "self-organization of behaviors in soft robots," and "persistent autonomy," nor on "human–robot interaction"; this is an interesting fact by itself.

2 CONTRIBUTED ARTICLES

The contributed articles provide a reasonably accurate state of the art for control methods currently used in marine robotics applications. Kelasidi et al. presents and integrates control for an underwater snake robot (this shape has a number of advantages in inspection and maintenance, the main issue being its control difficulty). Franchi et al. reports about the real-world experience and the technologies exploited during the European Robotics League in 2017. It is interesting to notice how "simple" the control techniques used in a real-world competition are with respect to other control approaches reported in the literature, while at the same time the team performed comparatively well. Garrido et al. shows (in simulation) the potential benefits of the fast marching method for navigation also in the marine robotics domain. Reis et al. addresses the cooperative moving path following (CMPF) control problem. It could be interesting to compare the three-layered hybrid collision avoidance (COLAV) system for autonomous surface vehicles in Eriksen et al. with that proposed in Kelasidi et al. Costanzi et al. reviews the interoperability issues in

OPEN ACCESS

Edited by:
Yongping Pan, National University of Singapore, Singapore

Reviewed by:
Fumin Zhang, Georgia Institute of Technology, United States

****Correspondence:***
Fabio Bonsignorio fabio.bonsignorio@heronrobots.com

Specialty section:
This article was submitted to Robotic Control Systems, a section of the journal Frontiers in Robotics and AI

Received: *16 January 2021*
Accepted: *18 February 2021*
Published: *15 April 2021*

Citation:
Bonsignorio F, Zereik E, Bibuli M, Pettersen KY and Khatib O (2021) Editorial: Advanced Control Methods in Marine Robotics Applications. Front. Robot. AI 8:654581. doi: 10.3389/frobt.2021.654581

complex maritime activities performed by teams of heterogeneous robots. It is interesting here to compare with Berman et al., which merges a secure sampling method for environmental monitoring—based on distributed ledger technology—with a swarm management methodology inspired by Reynold's boids. Could block-chain technologies help? Martinsen et al. reports about a reinforcement learning-based (RL) control scheme to optimize a model-based feedforward controller. De Palma et al. deals with the formation reconstruction for a team of vehicles based on the range distance between agents of a subset of the participants. Another interesting comparison is about swarm management in Berman et al. and cooperative control in Reis et al..

3 CONCLUSION

The diverse set of results published by a "real world–oriented" community probably gives a reasonable idea of where we are with robotics and AI at large.

Remarkably, a number of new tools are becoming available and are being exploited by the community. These include integrated guidance and control systems to ensure path following for underwater snake robots, new navigation methods, fast marching, new cooperative path following, obstacle and collision avoidance methods, new control approaches for range based navigation, reinforcement learning, and block-chain technologies among others. In particular, reinforcement learning seems a promising technique to improve the way robots perform most required tasks, while in general, deep learning is not widely used by the authors of this article collection. Fast marching holds promise for navigation tasks. Distributed ledger technologies seem to be ready for real-world application, to manage trustable and secure cooperation of heterogeneous multi-robot networks or swarms for critical applications (monitoring, but in general wherever security and trust are important).

Also in this community, a transition to a reproducible research practice is much needed, as only one contribution (Berman et al.) shared most of the code and data sets. The evaluation of the state of the art will certainly require further effort. However, marine robotics research is making inroads in this direction and increased progress can be expected in the coming years. Even from the down-to-earth point of view of marine robotics, we can expect quick and deep progress in the coming years.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial and intellectual contribution to the work and approved it for publication.

Conflict of Interest: Author FB was employed by the company Heron Robots.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Robotics and AI

ORIGINAL RESEARCH
published: 23 July 2019
doi: 10.3389/frobt.2019.00057

Path Following, Obstacle Detection and Obstacle Avoidance for Thrusted Underwater Snake Robots

Eleni Kelasidi[1,2*†], *Signe Moe*[3,4*†], *Kristin. Y. Pettersen*[1], *Anna M. Kohl*[1], *Pål Liljebäck*[1] *and Jan Tommy Gravdahl*[1,3]

[1] Department of Engineering Cybernetics, Centre for Autonomous Marine Operations and Systems, Norwegian University of Science and Technology, Trondheim, Norway, [2] Department of Seafood Technology, SINTEF Ocean, Trondheim, Norway, [3] Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway, [4] Department of Mathematics and Cybernetics, SINTEF Digital, Oslo, Norway

The use of unmanned underwater vehicles is steadily increasing for a variety of applications such as mapping, monitoring, inspection and intervention within several research fields and industries, e.g., oceanography, marine biology, military, and oil and gas. Particularly interesting types of unmanned underwater vehicles are bio-inspired robots such as underwater snake robots (USRs). Due to their flexible and slender body, these versatile robots are highly maneuverable and have better access capabilities than more conventional remotely operated vehicles (ROVs). Moreover, the long and slender body allows for energy-efficient transit over long distances similar to torpedo-shaped autonomous underwater vehicles (AUVs). In addition, USRs are capable of performing light intervention tasks, thereby providing intervention capabilities which exceed those of AUVs and inspection class ROVs. USRs may also propel themselves using energy-efficient motion patterns inspired by their biological counterparts. They can thereby increase the propulsion efficiency during transit and maneuvering, which is among the great challenges for autonomous underwater vehicles. In this paper, a control system for path following, and algorithms for obstacle detection and avoidance, are presented for a USR with thrusters attached at the tail module. The position of the obstacles is detected using a single camera in the head module of the USR and a developed computer vision algorithm. For the proposed control concept the robot joints are used for directional control while the thrusters are used for forward propulsion. The USR circumvents obstacles by following a circular path around them before converging back to the main straight line path when this is safe. Experimental results that validate the proposed methods are also presented.

Keywords: underwater snake robots, energy efficiency, thrusted USR, path following, obstacle detection and avoidance

OPEN ACCESS

Edited by:
Yongping Pan,
National University of Singapore, Singapore

Reviewed by:
Matthew Jason Bays,
Naval Sea Systems Command (NAVSEA), United States
Dimitris Tsakiris,
Foundation for Research and Technology Hellas, Greece

***Correspondence:**
Eleni Kelasidi
eleni.kelasidi@sintef.no
Signe Moe
signe.moe@ntnu.no

†These authors have contributed equally to this work and significantly in writing this paper

Specialty section:
This article was submitted to Robotic Control Systems, a section of the journal Frontiers in Robotics and AI

Received: *13 December 2018*
Accepted: *27 June 2019*
Published: *23 July 2019*

Citation:
Kelasidi E, Moe S, Pettersen KY, Kohl AM, Liljebäck P and Gravdahl JT (2019) Path Following, Obstacle Detection and Obstacle Avoidance for Thrusted Underwater Snake Robots. Front. Robot. AI 6:57. doi: 10.3389/frobt.2019.00057

1. INTRODUCTION

Through millions of years of evolution, sea snakes, eels and fish have developed highly efficient motion for propulsion and locomotion. These creatures are able to rapidly change direction in a highly efficient manner (Lighthill, 1970, 1975). Many of them have superior acceleration capabilities, while simultaneously being able to access confined spaces using their flexible bodies.

Over the last decades, remotely operated vehicles (ROVs) have been extensively used for subsea inspection, maintenance, and repair operations in the oil and gas industry (Christ and Wernli, 2013). These vehicles rely on being operated by a highly trained human in the loop. In order to make such operations safer and more cost-efficient, there has been an increasing interest in developing intervention AUVs (I-AUVs) (Ridao et al., 2014), underwater snake robots (USRs) (Mclsaac and Ostrowski, 1999; McIsaac and Ostrowski, 2002; Takayama and Hirose, 2002; Wilbur et al., 2002; Crespi et al., 2005; Yamada et al., 2005; Crespi and Ijspeert, 2006; Li et al., 2011; Stefanini et al., 2012; Liljebäck et al., 2014; Kelasidi et al., 2016a,b) and underwater snake robots with thrusters (Sverdrup-Thygeson et al., 2016a,b) as a step toward improved autonomy, dexterity and precision for underwater manipulation tasks. Detailed discussions on different underwater robotic systems such as ROVs, AUVs and bio-inspired robotic systems can be found in Kelasidi et al. (2016a) and Kelasidi et al. (2017b).

Inspired by biological swimming creatures, a novel concept for bio-inspired multi-articulated robotic systems has been illustrated in **Figure 1**, which combines properties of aquatic animals with state of the art solutions from marine technology. Unlike conventional underwater robotic solutions, the USR is a slender and highly redundant robot, which is able to propel itself forward using body undulations combined with caudal, dorsal and pectoral fins and/or with stern propellers and tunnel thrusters along the body. This provides significant flexibility and increases the maneuverability of the robot for subsea applications, as illustrated in **Figure 1** (Kelasidi et al., 2015; Sverdrup-Thygeson et al., 2016b). The modular design of the robot makes it suitable for different applications by simply connecting various modules in different combinations to form various types of vehicles. As illustrated in **Figure 1**, the robotic system is a dexterous robotic arm which can operate tools and carry out light intervention tasks. In addition, by using either foils or thrusters, it can transit over long distances in a similar manner as a survey AUV, while its flexible and slender body provides the ability to access and operate in restricted areas of subsea structures. The modular design of the robot makes it applicable for different applications depending on the requirement of the task. For instance, a purely bio-inspired solution without using propellers can be considered for applications where limited acoustic noise is required, whereas equipping the robot with thrusters can provide improved maneuverability for inspection and intervention tasks. The use of USRs for such subsea operations introduces several interesting research challenges. **Figure 2** presents the first USR equipped with thrusters at the tail module, developed at the Norwegian University of Science and Technology (NTNU) (Liljebäck et al., 2014; Kelasidi et al., 2016b). This robotic platform using thrusters only at the tail module can be considered a special case of the general concept shown in **Figure 1**, and is a step toward developing the next generation of USRs with additional effectors. The modular snake robot Mamba (Liljebäck et al., 2014) can be equipped with thrusters when operated underwater (Kelasidi et al., 2016b). Mamba with thrusters is a new type of snake-like robot which combines biologically inspired undulatory locomotion with the use of thrusters, and is the test platform considered for all the experimental results presented in this paper.

Obstacle avoidance is a crucial task for numerous robotic systems. For fixed base systems, the robot must avoid self-collisions as well as any objects that might be within its workspace. For floating base robots, such as a USR or a surface ship, the main task is to avoid stationary and dynamic obstacles such as islands/pipelines/other structures and other ships or floating base systems. There exist several path planning algorithms for computing a safe path to avoid obstacles, such as A*, RRT and HBug (Hernandez et al., 2015). However, these global path planning methods are not suitable for unknown and dynamic environments, and must be complemented by a local guidance system that is able to make the mobile robotic system avoid small, unforeseen, and dynamic obstacles while following the global path. A variety of such local approaches have been proposed, both for the general and maritime case, such as potential fields (Khatib, 1985), dynamic window (Fox et al., 1997; Loe, 2008), velocity obstacles van den Berg et al. (2011), Kuwata et al. (2014), and Tangent/WedgeBug Laubach and Burdick (1999). However, these approaches have several drawbacks. Potential fields may suffer from oscillating behavior and convergence to local minima (Koren and Borenstein, 1991), and the dynamic window approach can be computationally heavy. The velocity obstacle (VO) approach has good mathematical qualities and is computationally simple, but is not straightforward to implement. However, the main drawback of these methods is the fact that it is not obvious how to combine these collision avoidance methods with existing, commonly used guidance methods for path following such as line-of-sight (LOS) (Fossen, 2011). The Wedgebug algorithm is applied to Mars Rovers and assumes that the rover is modeled as a point robot in a 2D binary environment (i.e., every point in the environment is either contained within an impassable obstacle, or lies in freespace) and that obstacle boundaries block sensing as well as motion. In the approach proposed in this paper, the obstacle boundaries have the possibility to be virtual, which prevents passage into identified unsafe areas without physical obstacles in the way.

In nautical navigation there exists several obstacle avoidance methods which all require some information about the obstacle itself, i.e., position, size and/or velocity. To detect underwater obstacles, one may use sensors such as sonars and cameras (Nicholson and Healey, 2008; Ridao et al., 2014; Mallios et al., 2016). Due to the properties of light propagation under water, acoustics-based navigations methods are often applied. Vision systems decrease the range, but also decrease space and cost and increase the resolution (Bonin-Font et al., 2008). Often, vision systems are based on two cameras, i.e., stereo vision. With such a setup, one can use matching and geometric triangulation to calculate the 3D-position of detected features (Goldberg et al., 2002). The USR Mamba is equipped with a single camera at the head module. However, obstacle avoidance still requires sensing of depth, i.e., the distance between the vision sensor and the obstacle. To achieve this using monocular vision, one must rely on assumptions concerning the scene geometry and vehicle motion (Bhatti, 2008; Lei et al., 2013). In this paper, we

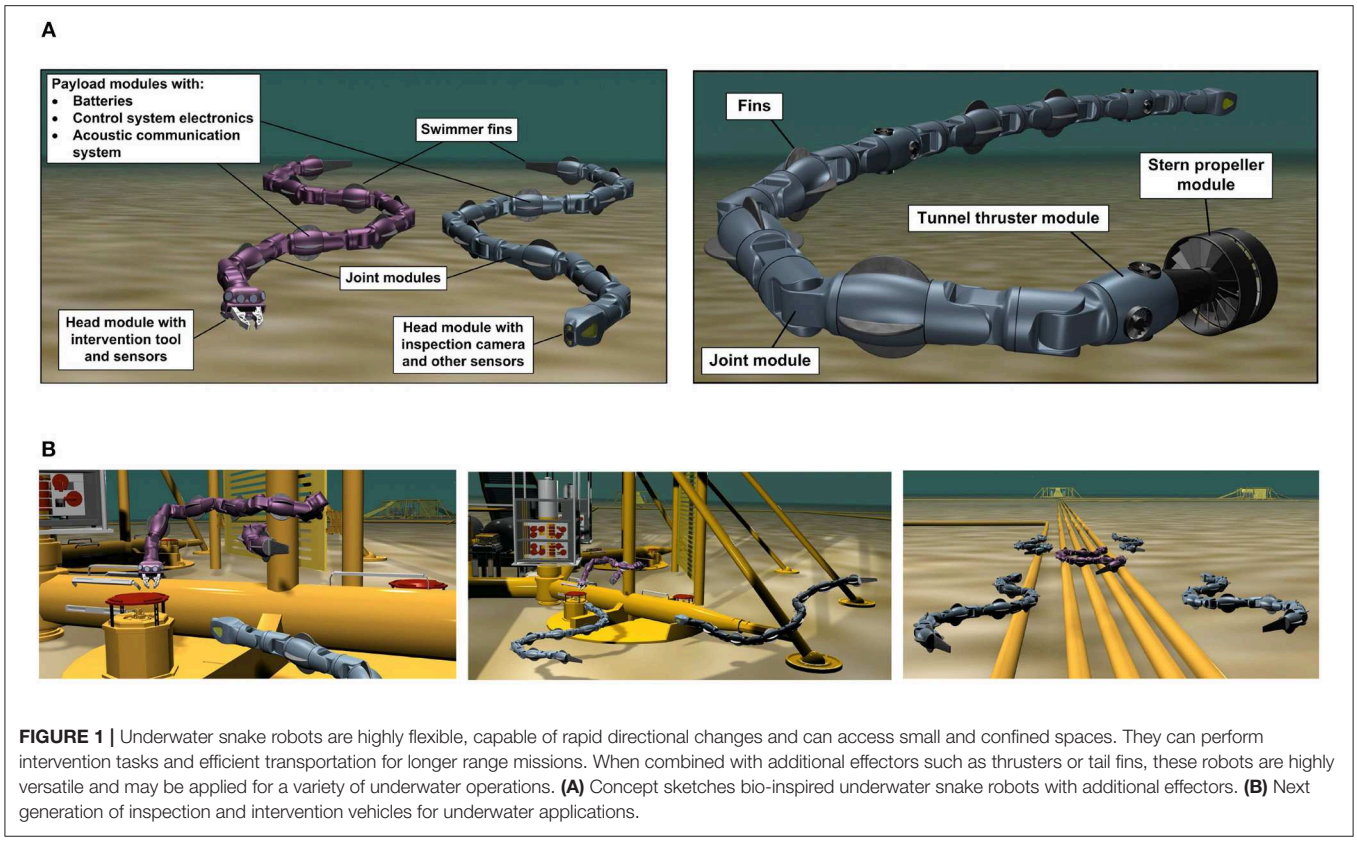


FIGURE 1 | Underwater snake robots are highly flexible, capable of rapid directional changes and can access small and confined spaces. They can perform intervention tasks and efficient transportation for longer range missions. When combined with additional effectors such as thrusters or tail fins, these robots are highly versatile and may be applied for a variety of underwater operations. **(A)** Concept sketches bio-inspired underwater snake robots with additional effectors. **(B)** Next generation of inspection and intervention vehicles for underwater applications.

FIGURE 2 | Different configurations of the underwater snake robot Mamba.

have developed a computer vision algorithm to detect potential obstacles along the path of the USR by using a single camera attached at the head module of the robot and reflective markers on the obstacles. The area of the markers is a priori knowledge and can be used to calculate the 3D distance based on the corresponding area in the image similar to Bousaid et al. (2016). Different geometric shapes (i.e., triangle, square and circle). This can be used to classify different types of obstacles. The shape is determined by analyzing the curvature of the shape and counting the number of peaks.

In this paper, we perform experiments to investigate both the path following and obstacle avoidance control problem using the USR Mamba with thrusters (Kelasidi et al., 2016b). The goal of the experiments is to detect potential obstacles along the path and design the USR motion to ensure that the robot can converge to and follow a predefined reference path while avoiding the detected obstacles.

In Kelasidi et al. (2016b), it is suggested that in order to ensure efficient transportation, a USR with thrusters at the tail module should mainly use the thrusters for locomotion, while the multi-articulated body should be used for directional control. Motivated by these results, we propose a motion control strategy for thrusted USRs with an overall goal of investigating its ability to follow a given reference path. Several previous works consider control schemes for locomotion of USRs without thrusters. A comparison of these approaches is presented in Kelasidi et al. (2017b) and Kelasidi et al. (2016a). In addition, a docking approach for thrusted USRs using the joint angles to control the direction of the robot has been presented in Sans-Muntadas et al. (2017).

This paper presents a path following control strategy that is able to make the thrusted USR follow the desired reference path. Furthermore, the developed obstacle detection scheme was successfully applied and combined with a set-based collision avoidance method (Moe and Pettersen, 2016; Kohl et al., 2017). This approach ensures obstacle avoidance when necessary and path following otherwise.

The path following control concept and obstacle avoidance for USRs without thrusters has been investigated in Kohl et al. (2017). Here, both the direction and propulsion are achieved through the undulatory motion of the joints. In this paper, these methods are adapted to USRs with thrusters. In addition, an obstacle detection strategy is presented and combined with the path following and obstacle avoidance methods. The proposed guidance and control strategy and obstacle detection and obstacle avoidance strategy are experimentally validated for USRs with thrusters. To the authors' best knowledge experimental results regarding obstacle detection and avoidance have not been presented in previous literature for thrusted USRs.

This paper is organized as follows: Section 2 presents the experimental setup as well as the guidance and control methods for path following and obstacle avoidance and the obstacle detection algorithm. In section 3, the experimental results are presented and discussed. Conclusions are given in section 4.

2. SETUP AND CONTROL SYSTEM

In this section, we give a brief description of the thrusted USR Mamba and the experimental setup. Furthermore, we discuss and present how the guidance and control approach proposed earlier for USRs without thrusters are adapted for the experiments with the thrusted Mamba. Finally, the obstacle detection technique adopted in this paper and the set-based obstacle avoidance approach proposed for the thrusted USRs are presented.

2.1. Experimental Setup

The underwater snake robot with thrusters at the tail module named Mamba (**Figure 2**) is basically a self-propelled robotic arm with a slender and flexible body able to access and carry out inspection tasks in confined spaces not accessible by conventional underwater vehicles. Mamba has a modular design and can operate at shallow water depths. For more information about the robot, see Kelasidi et al. (2016a,b). Note that for the thrusted robot it is important to know the amplitude of the applied thruster forces as a function of the particular control input. Initial experiments were performed to obtain the necessary mapping from the thruster inputs u_c to thruster forces F_t for the USR, and the results prove that the relationship is quite linear (Kelasidi et al., 2016b). Another purely bio-inspired configuration of the underwater snake robot Mamba with a passive caudal fin attached at the tail module of the robot (**Figure 2**) can be advantageous compared to the configuration with thrusters, since it does not produce significant acoustic noise. Moreover, a fin configuration will not perturb the surroundings as much as the thrusters, which is highly relevant for applications such as archaeological investigation of shipwrecks and non-invasive monitoring of marine life. A comparative study of the robot with and without a caudal fin was presented in Kelasidi et al. (2017a). In particular, it was shown that by attaching a passive caudal fin it is possible to double the forward velocity. This significant velocity increase requires a relatively low increase in power consumption, and is achieved with a minimum increase in the complexity of the mechanical design.

The robot considered in the current study consists of 18 joints mounted with a relative orientation of 90 degrees in an alternating fashion to achieve both yaw and pitch motion (Liljebäck et al., 2014). An external skin was used during the experiments in order to achieve an additional water barrier, in addition to making the robot's outer surface more smooth. The experimental setup is illustrated in **Figure 3**.

The experiments carried out in a basin at the Marine cybernetics laboratory (MC-lab), Trondheim, Norway (MCl, 2018). The basin is 1.5 m deep with a surface area of 40 m $\times$ 6.45 m. Six underwater cameras from Qualisys (QUA, 2018) were used to track and log the position and orientation θ of the robot, using a structure with reflective markers attached at the head or tail module. The center of mass (CM) p_x, p_y is then calculated using the kinematic equations of the robot (Kelasidi et al., 2016a). As illustrated in **Figure 3**, the obtained measurements were used to investigate different control challenges for the thrusted underwater snake robot Mamba. During these experiments the joint angles responsible for the vertical (pitch) motion were set to zero degrees to achieve purely horizontal motion. All the algorithms were implemented in an external computer using Labview, and the necessary signals were sent/received to/from the robot through a CAN bus connection through a tether. **Figure 3** illustrates three different case studies for USRs with thrusters: (1) Locomotion efficiency studies Kelasidi et al. (2016b), (2) Path following of USR with thrusters, and (3) Switching strategy between path following, and obstacle detection and obstacle avoidance modes developed and experimentally validated in this paper. In the following sections, each part of the case studies illustrated in **Figure 3** will be discussed in more detail.

2.2. Guidance and Control

The guidance and control system of the USR is illustrated in **Figure 3** and the definition of the mathematical symbols are described in **Table 1**. The guidance system provides a reference θ_{ref} for the orientation of the USR, which the controller attempts to follow by controlling the USR joints. The thrusters are controlled by the input u_c and each joint follows the output from the heading controller ϕ_0 according to (2). The proposed control strategy assumes that the robot joints are used for directional control, while the propulsion of the robot is given only by the thrusters. It is a decoupled system where the values of the control input uc is responsible for controlling the forward velocity of the robot, while the heading controller (2) is responsible for the turning motion of the robot.

Obstacle avoidance is by its very nature a safety feature which should be activated when necessary and otherwise not affect the behavior of the system. In this paper, the default mode of operation is straight line path following, although this objective may easily be replaced by another mode of operation to be combined with the proposed obstacle avoidance method. For

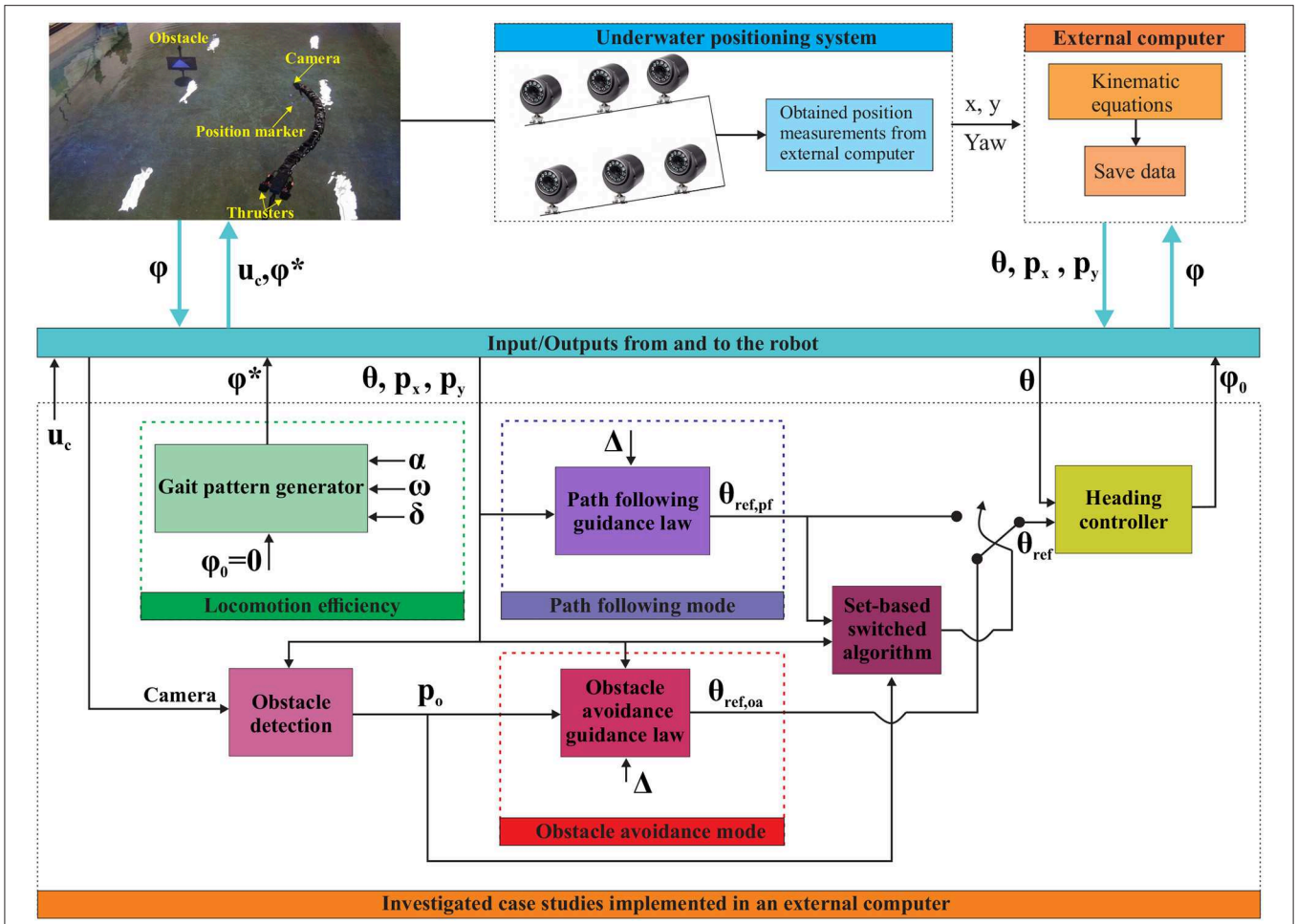


FIGURE 3 | Experimental setup to investigate path following control and obstacle detection and obstacle avoidance using the underwater snake robot Mamba.

more details, see section 2.3 and Moe and Pettersen (2016). To achieve a guidance system with a path following and an obstacle avoidance mode, we employ a guidance law from Kohl et al. (2017) which is suitable both for straight line and circular path following. The latter is applied for obstacle avoidance to encircle obstacles on the way.

Obstacles are avoided by ensuring that the USR always maintains a certain safe distance between itself and the obstacle. Thus, in our obstacle avoidance guidance system we propose to encircle an obstacle, whose center position is defined as $\boldsymbol{p}_o = [p_{ox}, p_{oy}]^T$, with a virtual circle of radius R_s. The circle center is anchored in the obstacle center, and the radius is chosen sufficiently large so that if the USR is outside or on R_s, a collision will not occur. Therefore, R_s is referred to as the safe radius, and the formalized control objective of the obstacle avoidance is to ensure that the USR is always outside or on R_s.

A variety of different path following control approaches for USRs without thrusters have been studied in previous literature (McIsaac and Ostrowski, 2003; Lapierre and Jouvencel, 2005; Alamir et al., 2007; Kelasidi et al., 2016a, 2017b). An introductory discussion comparing the different control approaches studied for underwater swimming robots can be found in (Kelasidi et al., 2016a, 2017b). In this paper, we present experimental results for the underwater snake robot Mamba with thrusters at the tail module, using the path following control approach described below.

The control approach consists of a path following guidance law responsible for producing the reference orientation $\theta_{\text{ref,pf}}$, the heading controller responsible for making the actual orientation θ track the reference orientation, and the control input u_c to the thrusters responsible to propel the robot forward. The reference orientation $\theta_{\text{ref,pf}}$ of the robot is calculated using the guidance law presented in (1), which for the straight line path following reduces to the well-known LOS guidance law. The LOS approach is based on a term guiding the vehicle in question along the desired path and another toward the path. The latter is reduced to zero when the vehicle is on the desired path and is commonly used both for marine vehicles (Børhaug and Pettersen, 2006; Breivik and Fossen, 2008) and USRs (Kelasidi et al., 2016a, 2017b).

In this paper, the reference path is aligned with the world x-axis. Therefore, the y-position of the USR p_y is defined as the path cross-track error for path following. The orientation of the robot was measured using the underwater camera positioning

TABLE 1 | Definition of mathematical terms.

Symbol	Description
u_c	Thruster inputs
θ	Orientation of the robot
θ_{ref}	Reference orientation of the robot
(x, y)	Vector of global coordinates of the CM of links
(p_x, p_y)	Global coordinates of the CM of the robot
α	Amplitude of sinusoidal motion pattern
ω	Frequency of sinusoidal motion pattern
δ	Phase shift between the joints with a sinusoidal motion pattern
ϕ_o	Joint offset coordinate used for directional control
$\boldsymbol{\phi}$	Vector of joint angles ϕ_i
$\boldsymbol{\phi^*}$	Vector of reference joint angles ϕ_i^*
$\boldsymbol{\Delta}$	Look-ahead distance
$\boldsymbol{p}_o$	Coordinates of the CM of the obstacle
$\theta_{\text{ref,pf}}$	Orientation of the robot during path following mode
$\theta_{\text{ref,oa}}$	Orientation of the robot during obstacle avoidance mode

system as shown in **Figure 3** by attaching reflective markers at the tail module of the robot. The heading controller (3) is used to generate the joint angle offset, ϕ_0, which is sent to the robot via the CAN.

There are multiple possible definitions of the orientation of an USR (Kelasidi et al., 2016a; Kohl et al., 2016). In this paper, the orientation θ of the robot is defined as the orientation of the head angle $\theta := \theta_N$. In the experimental setup, the USR position and orientation is measured using the underwater positioning system in the lab (see **Figure 3**). The reference orientation is defined by the following guidance law (Kohl et al., 2017):

$$\begin{aligned} \theta_{\text{ref}} &= \arctan(\tfrac{\mu_y}{\mu_x}), \\ \boldsymbol{\mu}(\boldsymbol{p}) &= -\frac{dh_p^T}{\|dh_p\|^2}\left(k_{\text{tran}} h(\boldsymbol{p})\right) + \nu \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} dh_p^T \frac{k_{\text{along}}}{\|dh_p\|}, \\ \nu &= \begin{cases} -1, & \text{circle counterclockwise} \\ +1, & \text{circle clockwise} \end{cases} \end{aligned} \tag{1}$$

Here, $h(\boldsymbol{p})$ is a cost function that implicitly defines the reference path, $dh_p^T = \nabla h(\boldsymbol{p})$ is a vector that is normal to the level sets of h, k_{tran} is the transversal gain, and k_{along} the along-path gain. This reference angle is referred to as $\theta_{\text{ref,pf}}$ and is utilized as a reference for path following (see **Figure 3**).

Since $dh_p^T = \nabla h(\boldsymbol{p})$ is perpendicular to the level sets of $h(\cdot)$, the control law (1) can be intuitively described as follows. The reference velocity $\boldsymbol{\mu}(\boldsymbol{p})$ is composed of two components: The first component is perpendicular to the level sets of $h(\cdot)$ and decreases the distance of the center of mass to the curve $\gamma = h^{-1}(0)$. The second component is tangent to the level sets of $h(\cdot)$ and regulates the velocity of the center of mass on the curve $\gamma = h^{-1}(0)$. The choice of ν enables us to choose the direction which the robot should follow around the obstacle.

Analogously, the angle $\theta_{\text{ref,oa}}$ obtained from (1) by using $h_{oa}(\boldsymbol{p}) = (p_x - p_{ox})^2 + (p_y - p_{oy})^2 - R_s^2$ is used as a reference for obstacle avoidance. In this case, the parameter ν controls the USR direction of motion, and is chosen such that the USR circumvents an obstacle by deviating as little as possible from the reference straight line path. Note that for this guidance scheme it is sufficient to know the position of the obstacles relative to the USR. However, in this paper we have calculated the obstacle world position because the obstacles are detected relative to the camera frame (attached to the USR head link), whereas the position of the USR is given as the CM.

When applied to a straight line, the guidance law (1) ensures that the USR converges to the reference path. However, for a circular path, the guidance law (1) ensures that the robot approaches the path and thereafter remains close to it with a constant offset outside the radius R_s.

Remark 1. *Note that the offset can be made small by increasing k_{tran} or eliminated completely by adding integral action to the guidance law. However, in this paper we deliberately choose to employ a rather small k_{tran} and thus always keep the USR safely outside the circle, rather than ensure that it converges closer to the safe radius R_s and possibly overshoots.*

The final part of the guidance system is an algorithm which determines if path following or obstacle avoidance is the active mode. This is described in more detail in section 2.3.

In Sans-Muntadas et al. (2017) it is proposed to set the reference for each joint as

$$\phi_i^*(t) = \phi_0, \tag{2}$$

i.e., to make each joint have the same value, providing an even curvature along the whole robot. This is different from Kohl et al. (2017) where undulations are used for propulsion, and the joint references include an additional sinusoidal term with a phase shift between the joints. Instead (2) ensures that the joints are used only for directional control, while the thrusters are used to propel the robot forward. In particular, Sans-Muntadas et al. (2017) has shown that by using (2) the robot managed to converge nicely toward and move along the desired path. Hence, in order to steer the thrusted USR to the reference orientation, the parameter ϕ_0 is used to control the direction of the robot. To steer the orientation θ according to the guidance law (1), the following PD controller is used to define the joint angle offset (Kohl et al., 2017):

$$\phi_0 = k_p \left(\theta_{\text{ref}} - \theta\right) + k_d \left(\dot{\theta}_{\text{ref}} - \dot{\theta}\right) \tag{3}$$

In the above equation, the control gains k_p and k_d are constant and positive. In addition, to ensure that the joint angle ϕ_i tracks the reference signal $\phi_i^* = \phi_0$, a low level P-controller is implemented in the microcontrollers inside each module of Mamba. Similarly, to assign a rotational speed to the thrusters, a corresponding low level controller is implemented to ensure that the two tail thrusters track the reference u_c.

2.3. Set-Based Obstacle Avoidance

It is clear that tasks such as path following and obstacle avoidance are not necessarily compatible. If an obstacle is somewhere along

the path, the USR either has to deviate from the path or collide. We therefore propose a switched control system with a path following and an obstacle avoidance mode. The default mode of operation, which is active as long as it will not lead to a collision, is path following. When the USR is close to an obstacle and path following will further decrease that distance, the system switches to collision avoidance mode.

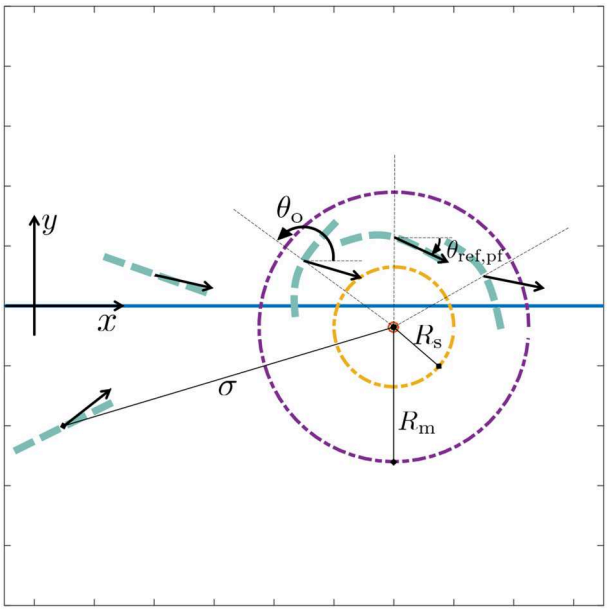


FIGURE 4 | Obstacle avoidance parameters: the set-based task σ is defined as the distance between the obstacle center and the USR. Outside the mode change radius R_m, the system is always in path following mode. The desired straight line path lies along the x-axis. The desired heading for path following is defined as $\theta_{ref,pf}$ and indicated by the black arrows for several USR positions and orientations. Inside R_m, the system is in path following mode if it will lead to an increase in σ, i.e., when the angle between $\theta_{ref,pf}$ and θ_o is smaller than or equal to $\pi/2$. Otherwise, obstacle avoidance mode is active, in which case the desired heading is defined by $\theta_{ref,oa}$ and the USR should converge the and track the safe radius R_s.

The switched guidance system is based on recent results in set-based control (Moe et al., 2016). Here, a widely used kinematic control framework is extended to handle set-based tasks, which have a valid interval of values rather than en exact desired state. Obstacle avoidance may be described as such a task, where the distance between the USR and an obstacle should be kept within a certain interval. In particular, the valid interval is given by all positive numbers above the lower bound R_s. However, the approach proposed in Moe et al. (2016) is applicable to redundant systems to fulfill several, compatible tasks simultaneously. Since the two control objectives, i.e., path following and obstacle avoidance, are not compatible, we therefore alter the approach according to Moe and Pettersen (2016) and Kohl et al. (2017) to switch between the two tasks, i.e., the two guidance laws $\theta_{ref,pf}$ and $\theta_{ref,oa}$ described in section 2.2.

For the switched system we introduce an additional circle, which is also anchored in the obstacle center $\boldsymbol{p}_o$, with a radius $R_m > R_s$. The radius R_m is referred to as the *mode change radius*. Outside the mode change radius, the guidance system is always in path following mode. Inside R_m, either mode may be active. If path following mode will not lead to the distance between the USR and the obstacle decreasing further, it is active. Otherwise, obstacle avoidance is activated, and the USR should converge toward the safe radius R_s. The mode change radius must be chosen sufficiently large so that in case of a switch to obstacle avoidance mode, the USR converges to the safe radius without overshoot. This is partly achieved by tuning the obstacle avoidance guidance law such that the USR converges to an offset outside R_s rather than to the actual safe radius as described in the previous section. The desired switching behavior is captured by

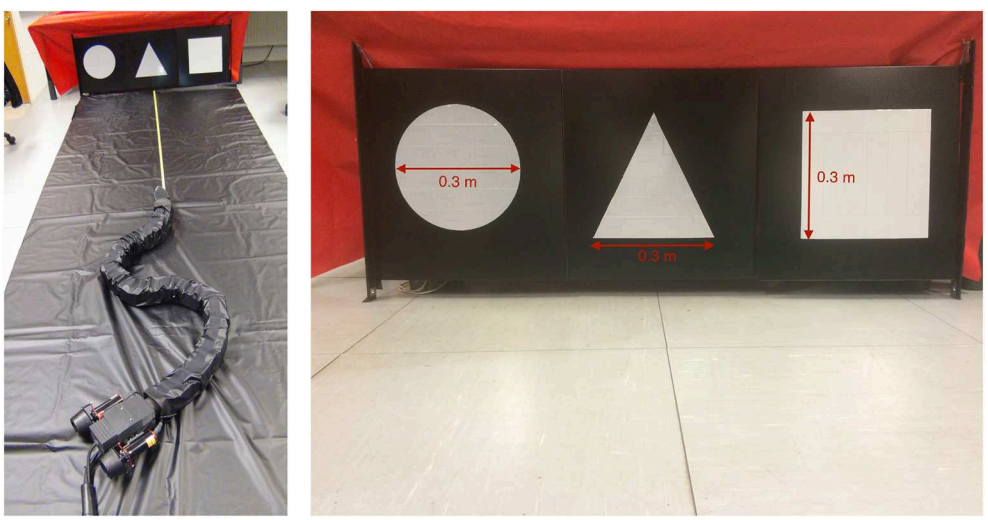


FIGURE 5 | The robot Mamba with thrusters and the reflective markers representing obstacles.

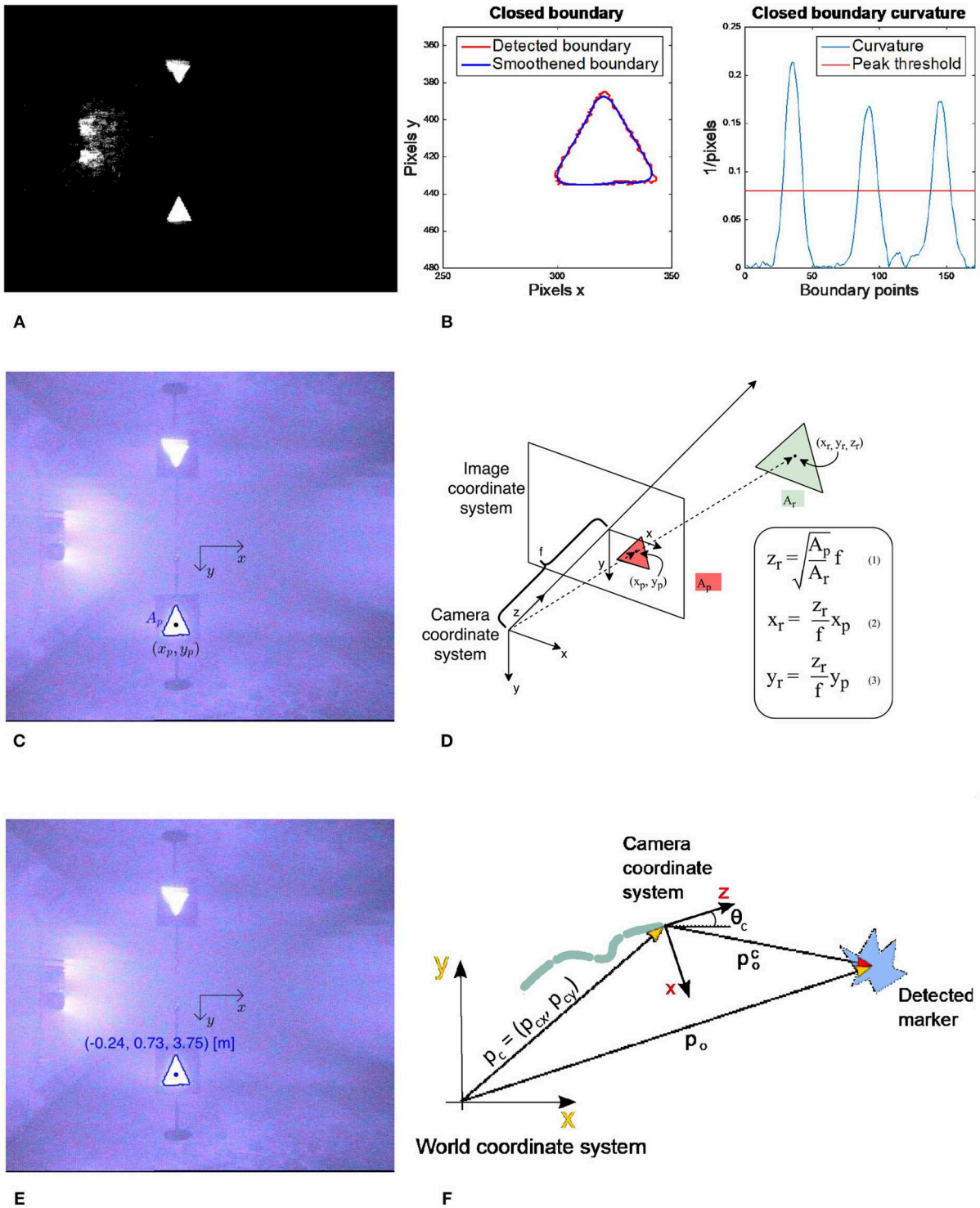


FIGURE 6 | The 3D-position of a marker relative to the camera coordinate system $\boldsymbol{p}_o^c = [x_r, y_r, z_r]^T$ can be calculated using the focal length f of the camera, the area of the marker in the image A_p and reality A_r and the position of the marker in the image coordinate system (x_p, y_p). In this paper, the USR is moving in the plane, so the vertical component y_r can be ignored. **(A)** Convert input image to black and white. **(B)** Find closed boundaries, smoothen them, calculate curvature and count number of peaks to classify geometric shape. **(C)** Find the position (x_p, y_p) and the area of the shape A_p in the image coordinate system. **(D)** Use the pinhole camera model, camera focal length f and the actual area of the shape A_r to calculate the 3D-position of the detected shape relative to the camera coordinate frame. **(E)** The calculated position is referred to as $\boldsymbol{p}_o^c = [x_r, y_r, z_r]$. **(F)** Use the USR camera position and orientation to calculate the detected position in the world coordinate frame $\boldsymbol{p}_o$.

Algorithm 1, which is based on set-based theory described in Moe et al. (2016), Moe and Pettersen (2016), and Kohl et al. (2017).

Remark 2. *Note that a similar approach is applied in Kohl et al. (2017) for obstacle avoidance of snake robots without thrusters that is able to propel forward only by using undulatory gaits. Due to the oscillatory behavior of the swimming snake robots, the set-based approach must be more conservative to ensure that no part of the robot collides with the obstacle. For thrusted USRs in this paper, we exploit the fact that the thrusters ensure forward propulsion and the joints control the direction of motion by letting the robot safely curve around the obstacle.*

Algorithm 1: The set-based switched guidance algorithm.

Input: σ, $\theta_{\text{ref,pf}}$, θ_{o}
if $\sigma \geq R_m$ **then**
  $\theta_{\text{ref}} = \theta_{\text{ref,pf}}$
else if $|\theta_{ref,pf} - \theta_o| \leq \frac{\pi}{2}$ **then**
  $\theta_{\text{ref}} = \theta_{\text{ref,pf}}$
else
  $\theta_{\text{ref}} = \theta_{\text{ref,oa}}$
end

For the obstacle avoidance scenario described above, the obstacle avoidance task σ is defined as the distance between the USR CM and an obstacle. It has a valid interval $D = [R_{\text{s}}, \infty)$, and the input parameters are illustrated in **Figure 4**, where $\theta_{\text{ref,pf}}$ is the desired heading for path following and θ_{o} is the angular coordinate of the obstacle. Thus, as illustrated in **Figure 4**, path following will result in the distance between the USR and the obstacle increasing when the angle between $\theta_{\text{ref,pf}}$ and θ_{o} is less than $\pi/2$. In this case, path following is active also within the radius R_{m}. Note that by using the CM of the USR when calculating σ, part of the USR is actually allowed within the safe radius R_{s}. This must be accounted for by choosing a sufficiently large R_{s}. Furthermore, the switching strategy in Algorithm 1 is completely general, and may be applied for any combination of guidance laws to achieve alternative desired behaviors such as target tracking, trajectory tracking or other path following schemes.

Remark 3. *Note that this method is valid for multiple obstacles given that said obstacles are not overlapping or moving. In these experiments, only one stationary obstacle was used due to the limited size of the test basin. Handling overlapping and moving obstacles is a topic for future work.*

2.4. Obstacle Detection

In this paper, we assume that the USR is to operate in some structured environment which we are free to influence, e.g., an underwater oil and gas structure. Hence, we presume that potential obstacles are marked with some sort of geometric shape that may be detected using a camera on the USR head and computer vision. Thus, obstacles of different sizes may be marked with different shapes. For unforeseen events such as debris another detection scheme must be applied. However, note that a set-based approach is still applicable for avoidance given estimation of obstacle position and velocity.

TABLE 2 | The average forward velocity and power consumption for the path following case studies using the underwater snake robot mamba with thrusters.

	u_c	F_t [N]	Δ [m]	k_p	$\theta(0)$ [deg]	$p_y(0)$ [m]	$\bar{v}$ [m/s]	P_{avg} [W]
Path 1	60	2.4362	1	0.18	-82.70	0.8905	0.2468	63.8400
Path 2	60	2.4362	1	0.09	-26.20	0.9544	0.2265	53.3855
Path 3	60	2.4362	1	0.09	21.4	1.7991	0.2167	58.4885
Path 4	60	2.4362	1	0.13	-2.70	1.1570	0.2240	45.8850

For these experiments, we used the pinhole camera model (Medioni and Kang, 2004) to derive the equations applied in the implemented detection algorithm. Three geometric shapes with a known area A_{r} have been constructed using reflective tape, and these represent the obstacles in the experiments: a circle, a triangle and a square (see **Figure 5**). To avoid an obstacle by circumventing it as described in sections 2.2 and 2.3, the obstacle position in the world coordinate frame $\boldsymbol{p}_{\text{o}}$ must be known. Hence, the goal of the obstacle detection algorithm is to calculate this position.

The obstacle detection algorithm is based on four main steps, which are illustrated in **Figure 6**: (1) Recognize and classify an obstacle marker as a triangle, square or circle, (2) find the position and area of the marker in the image x_{p}, y_{p} and A_{p}, (3) compare A_{p} to the actual area of the marker A_{r} and use the camera focal length f and the marker position in the image to calculate the 3D obstacle position relative to the camera $\boldsymbol{p}_{\text{o}}^{\text{c}}$, and (4) find the obstacle position relative to the world coordinate frame $\boldsymbol{p}_{\text{o}}$ by rotating and translating about the camera frame orientation and position. The detailed implementation can be found in Algorithm 2.

Remark 4. *Note that the equations in **Figure 6D** are based on the assumption that the obstacle marker is oriented parallel to the camera coordinate system xy-plane, i.e., that all the corners of the triangle have the same z-coordinate. This assumption is not satisfied if the camera is looking at a marker at an angle. However, due to the relatively small size of the markers, the potential difference in the z-coordinate of the corners is limited and small compared to the distance at which it is necessary to observe them to successfully avoid the obstacle. Thus, this assumption is a valid approximation and will result in a limited error in the calculated position.*

3. EXPERIMENTAL RESULTS

In this section, we discuss the obtained experimental results for the proposed path following control strategy (section 3.1) and the obstacle detection and avoidance concept (sections 3.2–3.3) described in previous section using the thrusted USR Mamba.

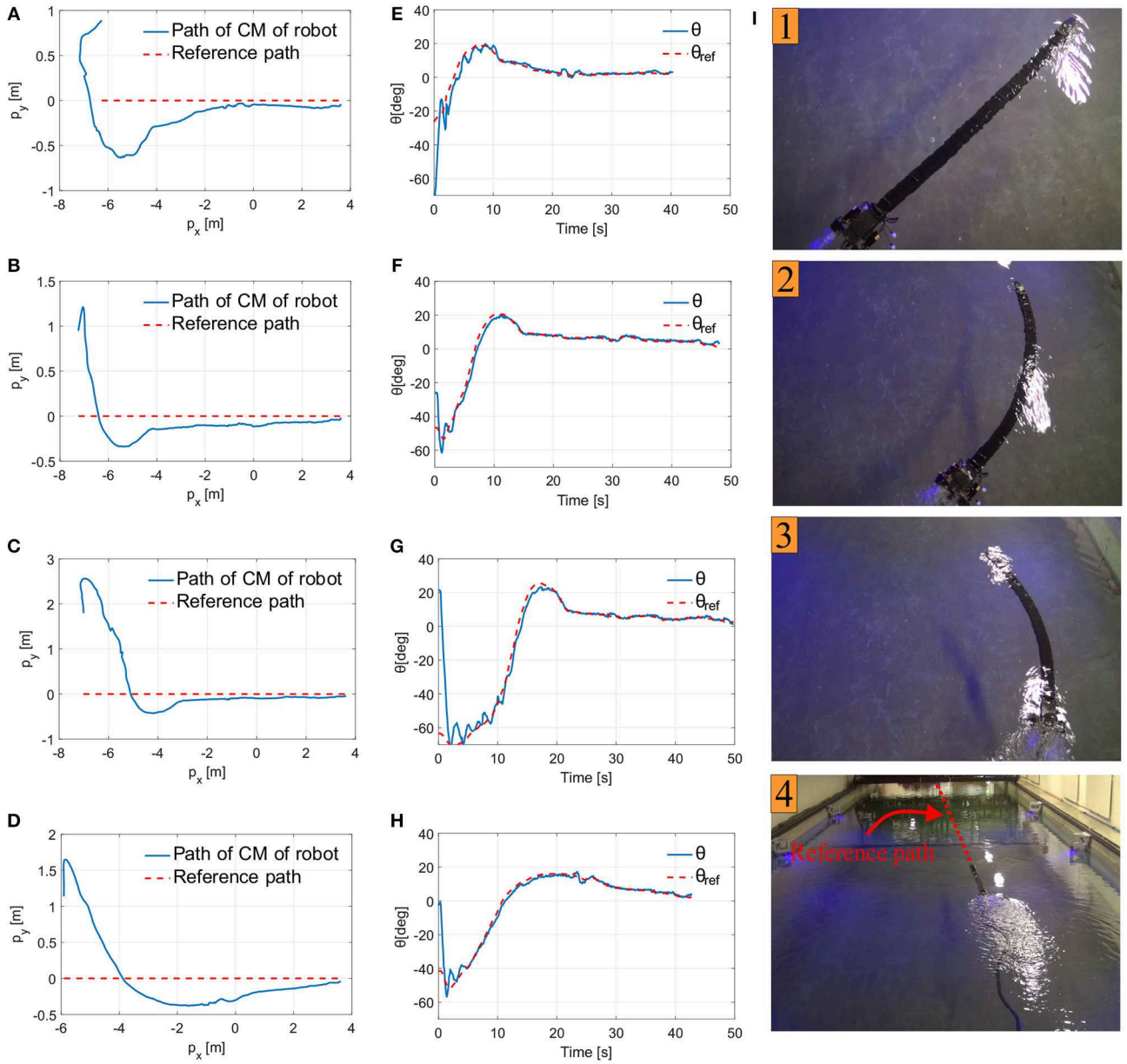


FIGURE 7 | Experimental results path following. **(A)** Center of mass position for $u_c = 60$, $k_p = 1$, $k_d = 0$, $\Delta = 180$cm and $\theta = -82.7^{\text{O}}$. **(B)** Center of mass position for $u_c = 60$, $k_p = 1$, $k_d = 0$, $\Delta = 90$cm and $\theta = -26.2^{\text{O}}$. **(C)** Center of mass position for $u_c = 60$, $k_p = 1$, $k_d = 0$, $\Delta = 90$cm and $\theta = 21.4^{\text{O}}$. **(D)** Center of mass position for $u_c = 60$, $k_p = 1$, $k_d = 0$, $\Delta = 130$cm and $\theta = 2.7^{\text{O}}$. **(E)** The orientation of the robot for case **(A)**. **(F)** The orientation of the robot for case **(B)**. **(G)** The orientation of the robot for case **(C)**. **(H)** The orientation of the robot for case **(D)**. **(I)** The motion of the underwater snake robot Mamba with thrusters during the path following for the experimental results presented in **(C,G)**, where the *red line* indicates the reference path.

3.1. Straight Line Path Following

In all experiments the joint angles of the robot were set to zero, whereas the initial orientation, $\theta(0)$, the position of the CM of the robot along the y axis, p_y, the proportional control gain, k_p, the look-ahead distance, Δ, and the control input to the thrusters, u_c, are displayed in **Table 2** for each trial. The average power consumption is calculated by using the following expression

$$P_{\text{avg}} = VI_{\text{avg}}, \tag{4}$$

where $V = 35$ [V] and I_{avg} [A] is the average current that is measured using the high performance industrial logging multimeter FLUKE 289. In addition, the average forward velocity for each experimental trial was calculated as

$$\bar{v} = \left(\sqrt{(p_{\text{stop},x} - p_{\text{start},x})^2 + (p_{\text{stop},y} - p_{\text{start},y})^2}\right) / (t_{\text{stop}} - t_{\text{start}}), \tag{5}$$

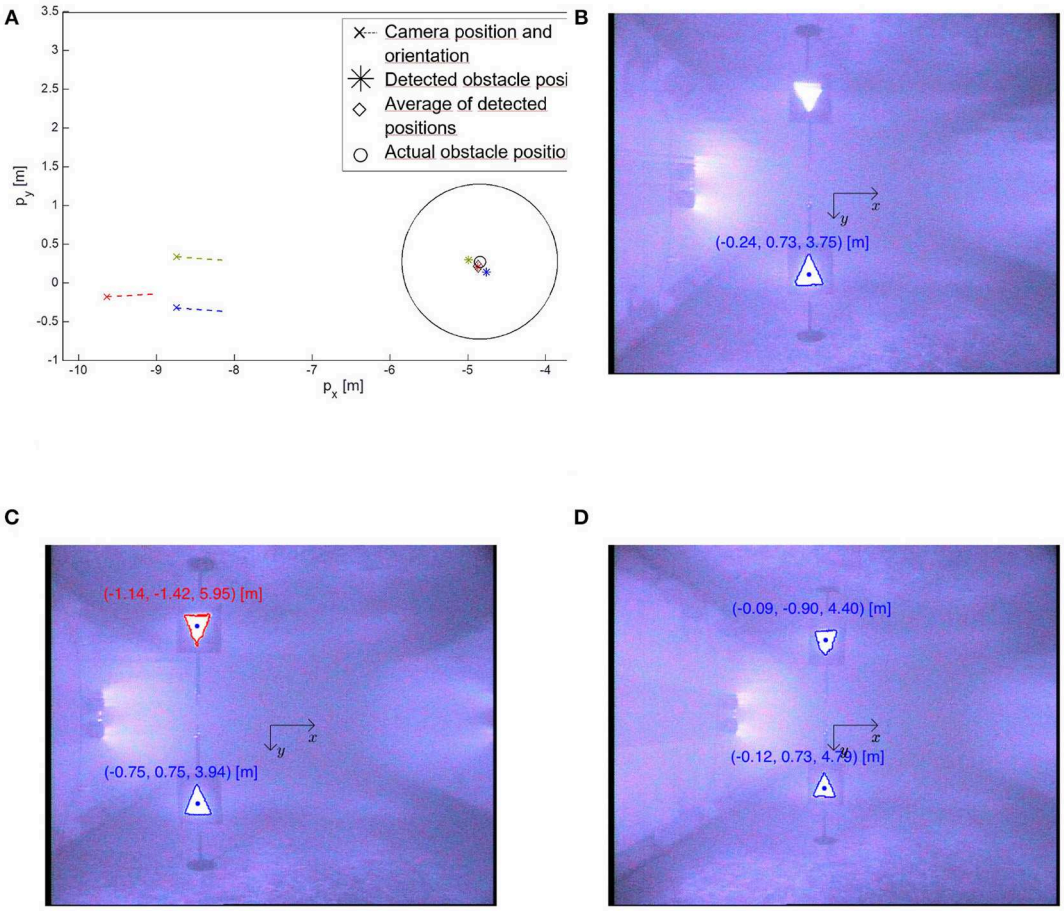


FIGURE 8 | The obstacle detection algorithm described in **Figure 6** was run three times for different USR positions and orientation shown as magenta, blue and red, respectively. For the two latter cases, the algorithm also detects the distorted surface reflection of the markers and classifies it as a square and a triangle, respectively. For a true underwater applications this phenomena will not occur, and these fake detections are easily disregarded by observing that their *y*-coordinate is negative, i.e., they are above the surface. **(A)** The average detected position ◇ was 7 cm from the actual obstacle position ◯. **(B)** The detected obstacle position relative to the camera $\boldsymbol{p}_O^C$ corresponding to magenta in **(A)**. **(C)** The detected obstacle position relative to the camera $\boldsymbol{p}_O^C$ corresponding to blue in **(A)**. **(D)** The detected obstacle position relative to the camera $\boldsymbol{p}_O^C$ corresponding to red in **(A)**.

where $\mathbf{p}_{\text{start}}$ and $\mathbf{p}_{\text{stop}}$ represent the initial and the final points of the distance traveled in the time interval $t_{\text{stop}} - t_{\text{start}}$. The control gain k_d was set to zero for the experimental results presented for the straight line path following control approach. In addition, the joint offset ϕ_0 has been saturated at $\pm 20^\circ$ to ensure that the physical limitation of the robot joint angles is not exceeded.

Previous experimental results for path following of underwater snake robots using the body undulation for both propulsion and directional control, showed that the robot was able to reach and follow the path using the LOS guidance law (Kelasidi et al., 2016a, 2017b). However, the use of an oscillatory gait pattern causes steady state oscillations about zero for the cross-track error and the orientation, which is expected since it is difficult to achieve a purely non-oscillating motion for the CM and the orientation of underwater swimming snake robots (Kelasidi et al., 2016a, 2017b). These oscillations can be restrictive for several applications in subsea environment, such as for instance docking (Sans-Muntadas et al., 2017).

Experimental results for four different path following trials of Mamba with thrusters are presented in **Figure 7**, see also the **Supplementary Videos**. As **Figure 7** illustrates, the robot manages to converge to and follow the desired path for all the investigated cases. Furthermore, the reference orientation is tracked without oscillations. The overshoot and the initial rapid change on the orientation shown in **Figure 7** as the snake robot converges to the path is a result of the tuning, and in particular the choice of the look-ahead distance Δ. The larger the choice of Δ is, the smaller the overshoot will be, and the slower the convergence rate will be. The choice of Δ is thus a trade-off between convergence and the overshoot (Kelasidi et al., 2017b). The small steady state error in cross-track error may be a result of several factors, such as the possible misalignment of the two thrusters used at the tail module of the robot, measurement errors from the different sensors used during

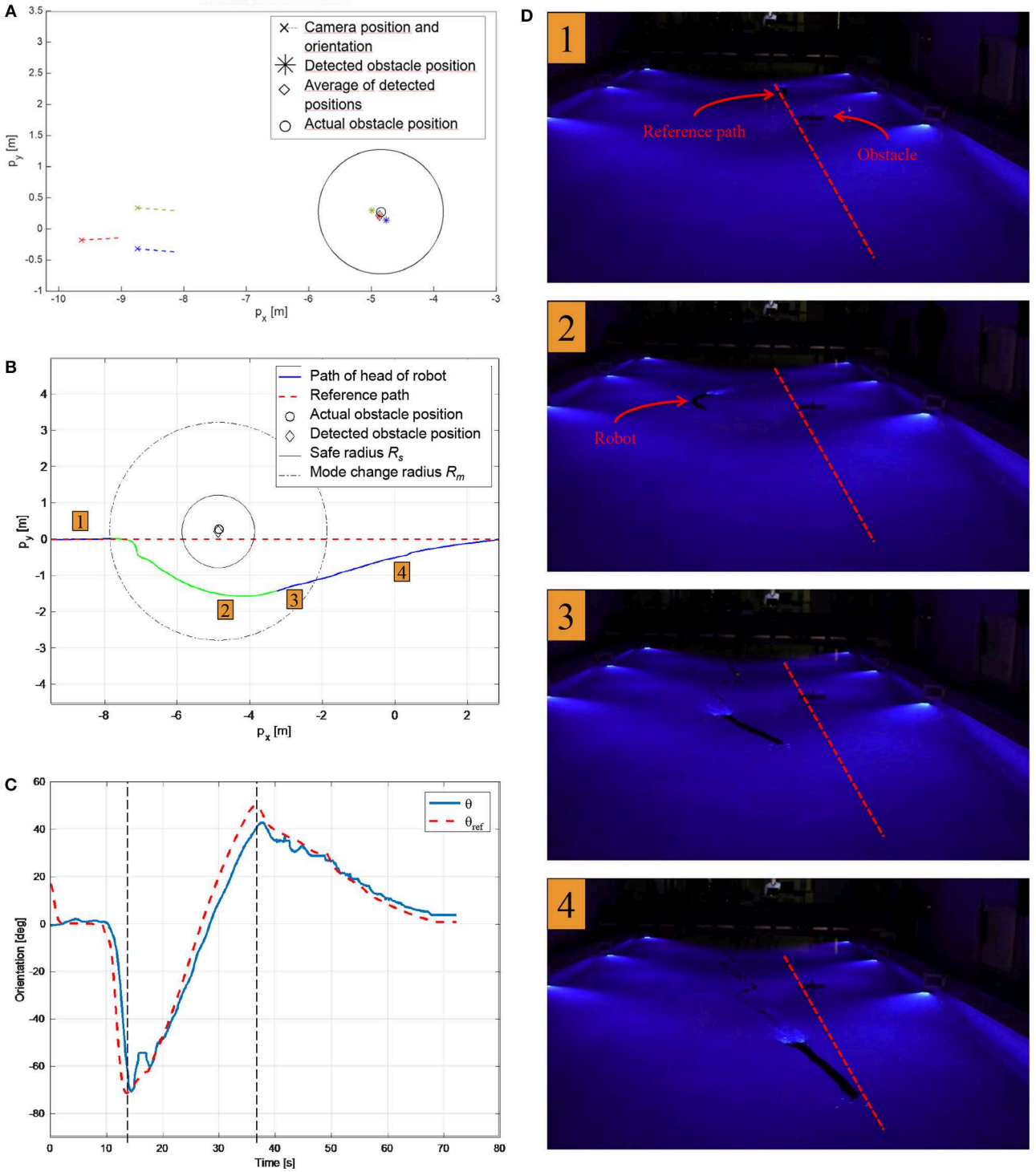


FIGURE 9 | Experimental results for path following with detected obstacle position and obstacle avoidance. **(A)** The obstacle detection algorithm described in **Figure 6** was run three times for different USR positions and orientation. The average detected position ◇ was approximately 0.07 m from the actual obstacle position ○. **(B)** The path of the USR as it follows the path (blue), switches to obstacle avoidance and circumvents the detected obstacle (green) and finally switches back to path following (blue). **(C)** The actual and desired orientation of the robot. The control system switches from path following to obstacle avoidance at $\approx t = 13$ s and back again at $\approx t = 37$ s. **(D)** The motion of the underwater snake Mamba with thrusters during the obstacle detection and avoidance experiments. Corresponding positions are indicated in **(B)**.

the experiments, and the forces on the robot due to the use of the tether. In the future, more advanced heading control approaches can be investigated to remove this small error, for instance by including integral action (Caharija et al., 2012). In addition to the convergence to the straight line path, we obtained results regarding the achieved forward velocity and the power consumption for all the investigated trials, and these are shown in **Table 2**. The achieved velocity is very similar in all trials, which is expected since the same control input value for the thrusters was used for all the investigated cases. However, the average power consumption varies for the different investigated path following case studies as shown in **Table 2**. This is reasonable since the power consumption is related not only to the trusters, but also to the joint modules. The actuation of the directional control and thereby the joint motion for each trial depends on the initial heading and distance from the path, which varies in the different trials.

3.2. Obstacle Detection

For the experiments presented in this paper, the obstacle detection algorithm was run off-line and the detected position was added manually to the switched guidance and control system described in section 2.3. However, the algorithm also has the potential to be fully autonomous as part of the online control system; the detection algorithm is sufficiently fast that runtime will not be a concern in an online implementation. The implementation of the necessary communication and control structure required to achieve this is a topic of future work. However, in the presented results the obstacle detection algorithm was run and the detected position added to the control system in one operation without removing the robot from the pool or turning it off.

To detect the obstacle position, the detection algorithm described in **Figure 6** was run three times for different camera positions and orientations, see **Figures 8**, **9A**. Note that when testing the obstacle detection scheme, the available USR positions and orientations where the obstacle marker was in the camera frame were limited by the pool size and the Qualisys tracking system. The average of the three detected positions $\boldsymbol{p}_{\mathrm{o}}$ was then inserted into the control system and used for the remainder of the experiments. To quantify the accuracy of the algorithm, the actual position of the obstacle was measured using the Qualisys tracking system, and the final detected position $\boldsymbol{p}_{\mathrm{o}}$ was approximately 0.07 m from the actual obstacle position, which corresponds to 7% of the safe radius $R_{\mathrm{s}} = 1$ m and 3.9% of the total length of the robot. This result is sufficiently accurate to safely use for the obstacle avoidance scheme, thereby confirming that the proposed detection approach is highly applicable. Note that to achieve a sufficiently good visual to detect and classify the reflective markers and to simulate a subsea environment as closely as possible, all lights were turned off during the experiments with the exception of the lights on the camera of the USR and the Qualisys tracking system.

3.3. Obstacle Avoidance

The experimental setup for path following and obstacle avoidance is identical to the one described in the previous section, and experimental results are shown in **Figures 9B–D**. A recording of the experiment can be seen in the **Supplementary Videos**. The USR initial position is on the reference path, and the initial mode is path following. Once the USR enters the mode change radius $R_{\mathrm{m}} = 3$ m, it is evident that continued path following will result in the USR getting closer to the obstacle. Hence, obstacle avoidance is activated, and the robot circumvents the obstacle by turning and attempting to stay outside the safe radius $R_{\mathrm{s}} = 1$ m. According to the theory described in section 2.2, the USR converges to a constant offset of the safe radius, which could be made smaller by a different choice of control gains. However, for this application it is crucial to avoid overshoot into R_{s}, and thus a larger offset is preferable. Furthermore, the position of the USR is defined by the CM, which also requires a more conservative approach since part of the USR will in fact be allowed to enter the safe radius R_{s} and must be able to do so safely. Finally, the physical obstacle in the pool partly blocked the camera tracking system, making it infeasible with the available experimental setup to attempt less conservative approaches which would exploit the flexibility of the USR better. Note that

Algorithm 2: Obstacle detection algorithm.

Input: Camera image, USR position and orientation p_x, p_y and θ

Convert image to black and white, search for closed boundaries of a certain size;

for *each closed boundary* **do**

Smoothen boundary, calculate curvature, count number of peaks in curvature;

if *number of peaks = 0* **then**

shape = circle;

A_r = area_circle_marker;

else if *number of peaks = 3* **then**

shape = triangle;

A_r = area_triangle_marker;

else if *number of peaks = 4* **then**

shape = square;

A_r = area_square_marker;

Find area and position in image A_{p}, x_{p} and y_{p};

Calculate position of shape relative to camera $\boldsymbol{p}_{\mathrm{o}}^{\mathrm{c}} = [x_{\mathrm{r}}, z_{\mathrm{r}}]^T$:

$$z_{\mathrm{r}} = \sqrt{\frac{A_{\mathrm{p}}}{A_{\mathrm{r}}}} f$$
$$x_{\mathrm{r}} = \frac{z_{\mathrm{r}}}{f} x_{\mathrm{p}}$$

Calculate position of shape in world frame $\boldsymbol{p}_{\mathrm{o}}$ by translation and rotation of the USR camera position and orientation:

$$\boldsymbol{p}_{\mathrm{o}} = \begin{bmatrix} p_{\mathrm{cx}} \\ p_{\mathrm{cy}} \end{bmatrix} + \begin{bmatrix} \cos(\theta_{\mathrm{c}}) & -\sin(\theta_{\mathrm{c}}) \\ \sin(\theta_{\mathrm{c}}) & \cos(\theta_{\mathrm{c}}) \end{bmatrix} \begin{bmatrix} z_{\mathrm{r}} \\ -x_{\mathrm{r}} \end{bmatrix}$$

end

the robot circumvents the obstacle by choosing the direction along the circle that ensures the shortest path as discussed in section 2.2.

As the USR circumvents the obstacle, path following will once again ensure that the robot moves away from the obstacle. Path following is then reactivated and the robot converges back to the path. This can be seen in **Figure 9B**. **Figure 9C** displays the reference orientation provided by the switched guidance system and the actual orientation of the USR. The implemented PD-controller ensures that the reference is tracked in a sufficiently accurate manner. Note that the switched guidance system described in Algorithm 1 results in abrupt changes in the reference orientation when the system switches between path following and obstacle avoidance. To provide the control system with a feasible reference signal, a hyperbolic smoothing function is implemented to ensure a continuous reference signal after a switch (Kohl et al., 2017). In addition, the commanded joint offset, ϕ_0, is filtered with a first-order low-pass filter before it enters the low-level controller.

Figure 9D displays images from the experiment. The USR clearly circumvents the obstacle on a circular path before converging back to the reference path. The chosen control parameters for the implementation are as follows:

1. Path following guidance law $\theta_{\text{ref,pf}}$: $k_{\text{tran}} = 0.1$, $k_{\text{along}} = 0.15$ (corresponding to a look-ahead distance $\Delta = 1.5$ m)
2. Obstacle avoidance guidance law $\theta_{\text{ref,oa}}$: $k_{\text{tran}} = 0.02$, $k_{\text{along}} = 0.15$
3. Controller ϕ_o: $k_p = 0.42$, $k_d = 0.03$
4. Smoothing function:

$$\alpha(t, t_{\text{last switch}}) = \frac{1}{2}\left(\tanh(\alpha_1(t - t_{\text{last switch}} - \alpha_2) + 1)\right),$$
$$\alpha_1 = 1.2, \alpha_2 = 1.6$$

4. CONCLUSIONS

USRs have a multitude of essential qualities for autonomous underwater operations, such as efficient locomotion, flexible bodies and the possibility to perform intervention tasks. These highly versatile robots may be equipped with different modules such as thrusters or fins, and are applicable for a variety of tasks within several fields of research.

In this paper, we present a guidance and control system to ensure path following and obstacle avoidance of a USR with thrusters, in addition to a computer vision algorithm to detect and calculate the position of potential obstacles. Based on preliminary results to ensure energy efficient motion and high velocity, the USR motion relies on thrusters for forward propulsion, whereas directional control is achieved through the joints of the body. The proposed methods are all experimentally verified for the first time, using the thrusted USR Mamba for the first time. Future work includes extending the proposed guidance and control approach to 3D in order to be able to investigate path following and obstacle avoidance of USR with thrusters in 3D.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct and intellectual contribution to the work, and approved it for publication.

ACKNOWLEDGMENTS

This work was funded by the Research Council of Norway through its Centres of Excellence funding scheme, project no. 223254-NTNU AMOS, and by VISTA - a basic research program in collaboration between The Norwegian Academy of Science and Letters, and Statoil.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/frobt.2019.00057/full#supplementary-material

Supplementary Video 1 | Obstacle avoidance.

Supplementary Video 2 | Path following thrusted Mamba.

REFERENCES

Alamir, M., El Rafei, M., Hafidi, G., Marchand, N., Porez, M., and Boyer, F. (2007). "Feedback design for 3D movement of an EEL-like robot," in *Proceedings of the IEEE International Conference on Robotics and Automation* (Roma), 256–261.

Bhatti, A., (ed.). (2008). *Stereo Vision*. Vienna: InTech Open.

Bonin-Font, F., Ortiz, A., and Oliver, G. (2008). Visual navigation for mobile robots: a survey. *J. Intell. Robot. Syst.* 53:263. doi: 10.1007/s10846-008-9235-4

Børhaug, E., and Pettersen, K. Y. (2006). "LOS path following for underactuated underwater vehicle," in *Proceedings of the 7th IFAC Conference on Manoeuvring and Control of Marine Craft* (Lisbon).

Bousaid, A., Theodoridis, T., and Nefti-Meziani, S. (2016). "Introducing a novel marker-based geometry model in monocular vision," in *2016 13th Workshop on Positioning, Navigation and Communications (WPNC)* (Bremen), 1–6.

Breivik, M., and Fossen, T. I. (2008). "Guidance laws for planar motion control," in *Proceedings of the 47th IEEE Conference on Decision and Control* (Mexico), 570–577.

Caharija, W., Pettersen, K. Y., Gravdahl, J. T., and Børhaug, E. (2012). "Integral LOS guidance for horizontal path following of underactuated autonomous underwater vehicles in the presence of vertical ocean currents," in *Proceedings of the American Control Conference* (Montreal, QC), 5427–5434.

Christ, R. D., and Wernli, R. L. (2013). *The ROV Manual: A User Guide for Remotely Operated Vehicles*. Waltham, MA: Elsevier Science.

Crespi, A., Badertscher, A., Guignard, A., and Ijspeert, A. (2005). "Swimming and crawling with an amphibious snake robot," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Barcelona), 3024–3028.

Crespi, A., and Ijspeert, A. J. (2006). "AmphiBot II: an amphibious snake robot that crawls and swims using a central pattern generator," in *Proceedings of the 9th International Conference on Climbing and Walking Robots (CLAWAR)* (Brussels), 19–27.

Fossen, T. I. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley.

Fox, D., Burgard, W., and Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robot. Automat. Mag.* 4, 23–33. doi: 10.1109/100.580977

Goldberg, S. B., Maimone, M. W., and Matthies, L. (2002). "Stereo vision and rover navigation software for planetary exploration," in *Proceedings of the IEEE Aerospace Conference Proceedings* (Big Sky, MT), 5.

Hernandez, E., Carreras, M., and Ridao, P. (2015). A comparison of homotopic path planning algorithms for robotic applications. *Robot. Auton. Syst.* 64, 44–58. doi: 10.1016/j.robot.2014.10.021

Kelasidi, E., Kohl, A. M., Pettersen, K. Y., and Gravdahl, J. T. (2017a). "An experimental investigation of path following for an underwater snake robot with a caudal fin," in *Proceedings of the 20th World Congress of the International Federation of Automatic Control* (Toulouse), 11182–11190.

Kelasidi, E., Liljebäck, P., Pettersen, K. Y., and Gravdahl, J. T. (2016a). Innovation in underwater robots: biologically inspired swimming snake robots. *IEEE Robot. Automat. Mag.* 23, 44–62. doi: 10.1109/MRA.2015.2506121

Kelasidi, E., Liljebäck, P., Pettersen, K. Y., and Gravdahl, J. T. (2017b). Integral line-of-sight guidance for path following control of underwater snake robots: theory and experiments. *IEEE Trans. Robot.* 33, 1–19. doi: 10.1109/TRO.2017.2651119

Kelasidi, E., Pettersen, K. Y., and Gravdahl, J. T. (2015). "Energy efficiency of underwater robots," in *Proceedings of the 10th IFAC Conference on Manoeuvring and Control of Marine Craft* (Copenhagen), 152–159.

Kelasidi, E., Pettersen, K. Y., Liljebäck, P., and Gravdahl, J. T. (2016b). "Locomotion efficiency of underwater snake robots with thrusters," in *Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics* (Lausanne), 174–181.

Khatib, O. (1985). "Real-time obstacle avoidance for manipulators and mobile robots," in *Proceedings of the IEEE International Conference on Robotics and Automation* (St. Louis, MO), 500–505.

Kohl, A., Moe, S., Kelasidi, E., Pettersen, K. Y., and Gravdahl, J. T. (2017). "Set-based path following and obstacle avoidance for underwater snake robots," in *IEEE International Conference on Robotics and Biomimetics* (Macau), 1206–1213.

Kohl, A. M., Kelasidi, E., Mohammadi, A., Maggiore, M., and Pettersen, K. Y. (2016). Planar maneuvering control of underwater snake robots using virtual holonomic constraints. *Bioinspir. Biomimet.* 11:065005. doi: 10.1088/1748-3190/11/6/065005

Koren, Y., and Borenstein, J. (1991). "Potential field methods and their inherent limitations for mobile robot navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation* (Sacramento, CA), 1398–1404.

Kuwata, Y., Wolf, M. T., Zarzhitsky, D., and Huntsberger, T. L. (2014). Safe maritime autonomous navigation with COLREGS, using velocity obstacles. *IEEE J. Ocean. Eng.* 39, 110–119. doi: 10.1109/JOE.2013.2254214

Lapierre, L., and Jouvencel, B. (2005). "Path following control for an eel-like robot," in *Proceedings of the MTS/IEEE International Conference Oceans* (Brest), 460–465.

Laubach, S. L., and Burdick, J. W. (1999). "An autonomous sensor-based path-planner for planetary microrovers," in *Proceedings of the 1999 IEEE International Conference on Robotics and Automation* (Detroit, MI), Vol. 1, 347–354.

Lei, G., Zhitao, X., Jun, W., Fang, Z., and Jinghong, M. (2013). Monocular vision distance measurement method based on dynamic error compensation. *Int. J. Digit. Content Technol. Appl.* 7, 230–237. doi: 10.4156/jdcta.vol7.issue5.28

Li, B., Yu, S., Ma, S., and Wang, Y. (2011). "An amphibious snake-like robot with novel gaits on ground and in water," in *Proceedings of the IASTED International Conference Intelligent Systems and Control (ISC 2011)* (Calgary, AB), 100–105.

Lighthill, M. J. (1970). Aquatic animal propulsion of high hydro-mechanical efficiency. *J. Fluid Mech.* 44, 265–301. doi: 10.1017/S0022112070001830

Lighthill, M. J. (1975). *Mathematical Biofluiddynamics*. Society for Industrial and Applied Mathematics. Available online at: https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/mathematical-biofluiddynamics-by-m-j-lighthill-society-for-industrial-and-applied-mathematics-1975-281-pp-2150/16A1602AAA4ED4BF89C7258E9365FFDA

Liljebäck, P., Stavdahl, Ø., Pettersen, K., and Gravdahl, J. (2014). "Mamba - a waterproof snake robot with tactile sensing," in *Proceedings of the International Conference on Intelligent Robots and Systems* (Chicago, IL), 294–301.

Loe, Ø. A. G. (2008). *Collision avoidance for unmanned surface vehicles*. (Master's thesis). Trondheim: Norwegian University of Science and Technology.

Mallios, A., Ridao, P., Ribas, D., Carreras, M., and Camilli, R. (2016). Toward autonomous exploration in confined underwater environments. *J. Field Robot.* 33, 994–1012. doi: 10.1002/rob.21640

McIsaac, K., and Ostrowski, J. (2002). "Experiments in closed-loop control for an underwater eel-like robot," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Washington, DC), 750–755.

McIsaac, K., and Ostrowski, J. (2003). Motion planning for anguilliform locomotion. *IEEE Trans. Robot. Automat.* 19, 637–625. doi: 10.1109/TRA.2003.814495

MCl (2018). *Marine Cybernetics Laboratory (MC-Lab)*. Available online at: https://www.ntnu.edu/imt/lab/cybernetics (accessed September 25, 2018).

Mclsaac, K., and Ostrowski, J. (1999). "A geometric approach to anguilliform locomotion: modelling of an underwater eel robot," in *Proceedings of the International Conference on Robotics and Automation (ICRA)* (Detroit, MI), 2843–2848.

Medioni, G., and Kang, S. B. (2004). *Emerging Topics in Computer Vision*. Upper Saddle River, NJ: Prentice Hall PTR.

Moe, S., Antonelli, G., Teel, A. R., Pettersen, K. Y., and Schrimpf, J. (2016). Set-based tasks within the singularity-robust multiple task-priority inverse kinematics framework: general formulation, stability analysis, and experimental results. *Front. Robot. AI* 3:16. doi: 10.3389/frobt.2016.00016

Moe, S., and Pettersen, K. Y. (2016). "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* (Athens), 402–409.

Nicholson, J. W., and Healey, A. J. (2008). The present state of autonomous underwater vehicle (AUV) applications and technologies. *Mar. Technol. Soc. J.* 42, 44–51. doi: 10.4031/002533208786861272

QUA (2018). *Qualisys–Motion Capture Systems*. Available online at: http://www.qualisys.com/ (accessed September 25, 2018).

Ridao, P., Carreras, M., Ribas, D., Sanz, P., and Oliver, G. (2014). "Intervention AUVs: the next challenge," in *Proceedings of the 19th IFAC World Congress* (Cape Town), 12146–12159.

Sans-Muntadas, A., Kelasidi, E., Pettersen, K. Y., and Brekke, E. (2017). "Spiral path planning for docking of underactuated vehicles with limited fov," in *Proceedings of the 1st IEEE Conference on Control Technology and Applications* (Hawaii), 732–739.

Stefanini, C., Orofino, S., Manfredi, L., Mintchev, S., Marrazza, S., Assaf, T., et al. (2012). A novel autonomous, bioinspired swimming robot developed by neuroscientists and bioengineers. *Bioinspir. Biomimet.* 7:025001. doi: 10.1088/1748-3182/7/2/025001

Sverdrup-Thygeson, J., Kelasidi, E., Pettersen, K. Y., and Gravdahl, J. T. (2016a). "A control framework for biologically inspired underwater swimming manipulators equipped with thrusters," in *Proceedings of the 10th IFAC Conference on Control Applications in Marine Systems* (Trondheim), 89–96.

Sverdrup-Thygeson, J., Kelasidi, E., Pettersen, K. Y., and Gravdahl, J. T. (2016b). "The underwater swimming manipulator - a bio-inspired auv," in *Proceedings of the IEEE/OES Autonomous Underwater Vehicles* (Tokyo), 387–395.

Takayama, T., and Hirose, S. (2002). "Amphibious 3d active cord mechanism "helix" with helical swimming motion," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Lausanne), 775–780.

van den Berg, J., Guy, S. J., Lin, M., and Manocha, D. (2011). "Reciprocal n-body collision avoidance," in *Robotics Research: The 14th International Symposium ISRR* (Berlin; Heidelberg: Springer), 3–19.

Wilbur, C., Vorus, W., Cao, Y., and Currie, S. (2002). "A lamprey-based undulatory vehicle," in *Neurotechnology for Biomimetic Robots*, eds J. Ayers, J. L. Davis, and A. Rudolph (Cambridge: Bradford/MIT Press), 285–296. Available online at: https://cbns.ucr.edu/faculty/currie_robolamprey2002.pdf

Yamada, H., Chigisaki, S., Mori, M., Takita, K., Ogami, K., and Hirose, S. (2005). "Development of amphibious snake-like robot ACM-R5," in *Proceedings of the 36th International Symposium on Robotics* (Tokyo).

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Robotics and AI

ORIGINAL RESEARCH
published: 21 November 2019
doi: 10.3389/frobt.2019.00121

Robust Cooperative Moving Path Following Control for Marine Robotic Vehicles

Matheus F. Reis, R. Praveen Jain, A. Pedro Aguiar and Joao Borges de Sousa*

Department of Electrical and Computer Engineering, Faculty of Engineering, University of Porto, Porto, Portugal

This paper presents results on recent developments pertaining to the coordinated motion control of a fleet of marine robotic vehicles. Specifically, we address the Cooperative Moving Path Following (CMPF) motion control problem, that consists of steering the robotic vehicles along a priori specified geometric paths that jointly move according to a target frame, while achieving a pre-defined coordination objective. To this end, each vehicle will need to communicate with their neighbors in order to cooperatively solve the CMPF task. Two distinct robust Moving Path Following motion control strategies for achieving robustness on the moving path following tasks are proposed. Experimental results demonstrating the application of CMPF to marine vehicles in the context of source localization and tracking of underwater targets are presented backed with stability and convergence guarantees.

Keywords: marine robotics, underactuated robotics, path following, robust control, cooperative control

OPEN ACCESS

Edited by:
Fabio Bonsignorio, The BioRobotics Institute Scuola Superiore Sant'Anna, Italy

Reviewed by:
Charalampos P. Bechlioulis, National Technical University of Athens, Greece
Zhouhua Peng, Dalian Maritime University, China

****Correspondence:***
Matheus F. Reis
matheus.ferreira.reis@gmail.com

Specialty section:
This article was submitted to Robotic Control Systems, a section of the journal Frontiers in Robotics and AI

Received: *16 May 2019*
Accepted: *04 November 2019*
Published: *21 November 2019*

Citation:
Reis MF, Jain RP, Aguiar AP and de Sousa JB (2019) Robust Cooperative Moving Path Following Control for Marine Robotic Vehicles. Front. Robot. AI 6:121. doi: 10.3389/frobt.2019.00121

1. INTRODUCTION

The motion control problem for underactuated robotic vehicles is a relatively mature area of research, with important works addressing trajectory tracking and path following schemes. In the path following (PF) problem, the vehicle is tasked to follow a *fixed* geometric path without the need of satisfying explicit time constraints, in contrast to trajectory tracking. A series of results addressing the PF motion control problem were published, starting with the pioneering work in Samson (1992), Micaelli and Samson (1993), and Aguiar and Pascoal (2007) for the case of wheeled mobile robots, Encarnação et al. (2000), Belleter et al. (2016) and references therein for marine vehicles and Cichella et al. (2011), Xargay et al. (2013) for UAVs.

A generalization of the path following problem is termed the Moving Path Following (MPF) motion control problem, which consists of steering the robotic vehicle along an a priori specified geometric path expressed with respect to a *moving* target frame. This problem finds applications in source seeking, convoy protection, target tracking, surveillance and monitoring and also autonomous landing. For example, in target tracking applications in the maritime environment, it is desirable for the vehicles to perform different types of maneuvers. These maneuvers can be framed as specific paths to be followed around the tracked target, and often allow the vehicle to have the necessary flexibility to operate in a highly complex environment, which is constantly inducing disturbances into its body due to the presence of maritime currents, waves and hydrodynamic effects. Further, it is observed that the MPF problem retains the advantages of the classical path following schemes (Aguiar et al., 2005) such as faster convergence of the robot to the moving path, while allowing the target reference frame to move freely.

The works in Oliveira and Encarnação (2013) and Oliveira et al. (2016) introduced the MPF control problem for tracking of ground targets using Unmanned Aerial Vehicles (UAVs) and later on, Oliveira et al. (2017) extended the solution to the 3D case. The proposed approach was suitable for robotic vehicles requiring a minimum positive forward speed, such as certain types of AUVs. In Jain et al. (2018b), a Lyapunov-based MPF control approach was presented for robotic vehicles without this restriction. Other control methods such as vector field method (Kapitanyuk et al., 2017) and nonlinear model predictive control (Jain et al., 2018c) have been proposed to solve the MPF problem. In contrast to the contributions of this paper, the salient features of the above reviewed literature are that they do not consider the external disturbances that depend on the operational environment, such as maritime currents, wind or rough terrain, that can affect the performance of the MPF controller. Further, they assume that the velocity of the target frame is known.

In path following literature, the problem of robustness has been addressed for example, in Dagci et al. (2003), where a cascade sliding mode controller for both kinematics and dynamics of a robotic vehicle was designed. In Aguiar and Pascoal (2002), a disturbance observer for constant unknown ocean currents was designed to solve the problem of dynamic positioning and way-point tracking of an underactuated AUV. Later on, the problem of robustness against parametric uncertainty in trajectory tracking and path following was also addressed in Aguiar and Hespanha (2007). More recently, in Zhang et al. (2014), a sliding mode technique combined with a predictive control strategy was developed to compensate for the impact of the hydrodynamic damping coupling on a 3D path following task for an Autonomous Underwater Vehicle (AUV). In Wang et al. (2016), a H_∞ robust controller for ground vehicles is proposed to achieve path following in the presence of disturbances caused by delays and data packet dropouts. All of the above schemes consider robustness for the path following problem. From the best of the authors knowledge, the only work concerning the problem of robustness in MPF literature is Reis et al. (2019), where sliding mode based controllers and a disturbance observer were designed to compensate external disturbances acting on the robotic vehicle.

A further extension of the MPF framework for multi-robot applications and formation control is the Cooperative Moving Path Following (CMPF) control problem, which consists in steering N vehicles along N paths defined with respect to a moving target while achieving some coordination objective. A special case of CMPF control, where the paths are fixed with respect to a given reference frame is the framework of Cooperative Path Following (CPF). As a recent example, the robustness problem in CPF literature was addressed in Gu et al. (2019), where two cooperative path following controllers using an Extended State Observer to estimate and compensate external disturbances in the kinetic level were proposed and validated experimentally using Autonomous Surface Vehicles (ASVs). In (Jain et al., 2018a), an event-based controller was explicitly designed to reduce the frequency of communication between the robotic vehicles. The control strategy effectively decomposes the control structure into two distinct layers. The first is responsible for the motion control of each individual vehicle, termed the PF controller. The second, termed the cooperative controller, is responsible for achieving coordination between the robots by using a consensus law. However, (Jain et al., 2018a) does not consider uncertainties and disturbances acting on the robotic vehicles. By decoupling the motion control layer from the cooperative control layer, one could use robust MPF controllers in the first layer to deal with the presence of certain types of disturbances acting on the vehicles, without affecting the formation control.

This paper extends the results obtained for the MPF controllers proposed by Reis et al. (2019) to the Cooperative MPF framework. Two MPF control strategies are proposed for the motion control layer, both using a known target pose and estimates of the target velocities. The first strategy employs a First Order Sliding Mode (FOSM) term to achieve robustness against bounded disturbances. The second strategy seeks to directly compensate the disturbance by computing an estimate of the disturbance using a disturbance observer. The cooperative layer consists of the consensus law proposed by Aguiar (2017). The stability of the proposed control laws is analyzed and it is shown that the origin of the path error is stable and converges to a small neighborhood around zero, even in the presence of bounded estimation errors on the target velocities and environmental disturbances. The design and theoretical results for the two variants of the proposed robust controllers are experimentally validated in a CMPF scenario using Autonomous Underwater Vehicles.

2. PROBLEM FORMULATION

2.1. Kinematic Model for an Underactuated Vehicle

Consider an inertial frame of reference $\{I\}$ and N robotic vehicles, each with its body frame $\{B_i\}$ attached to its center of mass. Define the set of N robotic vehicles as $\mathcal{I} = \{1, 2, ..., N\}$. The kinematic model of the i-th vehicle moving in $\mathbb{R}^n$ with $n = 2, 3$ can be expressed by

$$\begin{aligned} \dot{\boldsymbol{p}}_i(t) &= R_i(t)\, \boldsymbol{v}_i + \boldsymbol{d}_{v,i} \\ \dot{R}_i(t) &= R_i(t)\, S(\boldsymbol{\omega}_i + \boldsymbol{d}_{\omega,i}) \end{aligned} \tag{1}$$

where $\boldsymbol{p}_i \in \mathbb{R}^n$ denotes the position of the i-th robot with respect to the frame $\{I\}$, $R_i \in \mathcal{SO}(n)$ denotes the rotation matrix from the frame $\{B_i\}$ to an inertial frame $\{I\}$, $\boldsymbol{v}_i \in \mathbb{R}^n$ and $\boldsymbol{\omega}_i \in \mathbb{R}^{n(n-1)/2}$ are the linear and angular velocities of the i-th vehicle with respect to its own body frame, $S(\boldsymbol{\omega}_i) \in \mathfrak{so}(n)$ is the skew-symmetric matrix associated to the angular velocity $\boldsymbol{\omega}_i$.

Finally, $\boldsymbol{d}_{v,i} \in \mathbb{R}^n$ and $\boldsymbol{d}_{\omega,i} \in \mathbb{R}^{n(n-1)/2}$ are kinematic disturbances acting on each robot. Many different factors can be the source of these disturbances, depending on the type of vehicle and the operational environment. Marine vehicles such as AUVs are affected by unknown sea conditions that can induce unwanted external velocities due to maritime currents,

waves and wind. In the case of aerial vehicles, wind and internal dynamics can induce some unwanted disturbances in the kinematic model. In this work, we consider the problem of controlling an underactuated vehicle at the kinematic level, with the control signal defined as

$$\boldsymbol{u}_i = \begin{bmatrix} v_{f,i} \\ \boldsymbol{\omega}_i \end{bmatrix} \tag{2}$$

where the body linear velocity $\boldsymbol{v}_i$ is defined as $\boldsymbol{v}_i = [\ v_{f,i}\ 0\ ]^\mathsf{T}$ $(n = 2)$ or $v_i = [\ v_{f,i}\ 0\ 0\ ]^\mathsf{T}$ $(n = 3)$. This is the case for vehicles where only the longitudinal velocity $v_{f,i} \in \mathbb{R}$ and the body angular velocity $\boldsymbol{\omega}_i \in \mathbb{R}^{n(n-1)/2}$ can be controlled, such as some types of AUVs. We assume that the vehicle has an inner-loop autopilot controller that is responsible to track the linear and angular velocity commands generated by the controller based on the kinematic model of the robotic vehicle. Imperfect tracking by the inner-loop autopilot controller can further contribute to the velocity disturbances acting on the vehicle, that can be lumped into the terms $\boldsymbol{d}_{v,i}$ and $\boldsymbol{d}_{\omega,i}$.

2.2. Cooperative Moving Path Following Problem

In the CMPF control problem, the vehicles must follow a priori specified paths expressed with respect to a moving target whose position can be accurately estimated, while also maintaining some coordination objective. Define the target frame $\{T\}$ with its origin attached to the target center of mass. Then, the cooperative MPF problem can be divided in the following two sub-problems.

2.2.1. Moving Path Following Problem

Let $\boldsymbol{p}_t(t) \in \mathbb{R}^n$ denote the position of the target with respect to the frame $\{I\}$, and $\boldsymbol{p}^t_{d,i}(\gamma_i) \in \mathbb{R}^n$ be the desired path for vehicle i, specified with respect to the frame $\{T\}$ and parameterized by the path variable $\gamma_i \in \mathbb{R}$. As illustrated by **Figure 1**, for a given γ_i and time t, $\boldsymbol{p}_{d,i}(\gamma_i, t)$ and $\dot{\boldsymbol{p}}_{d,i}(\gamma_i, t)$ denote the position and velocity of the virtual reference point that must be followed by the i-th vehicle, with respect to the inertial frame $\{I\}$:

$$\boldsymbol{p}_{d,i}(\gamma_i, t) = \boldsymbol{p}_t + R_t\, \boldsymbol{p}^t_{d,i} \tag{3}$$

$$\dot{\boldsymbol{p}}_{d,i}(\gamma_i, t) = \boldsymbol{v}_t + R_t \left(\nabla \boldsymbol{p}^t_{d,i}\, \dot{\gamma}_i + S(\boldsymbol{\omega}_t)\, \boldsymbol{p}^t_{d,i} \right) \tag{4}$$

where $R_t(t) \in \mathcal{SO}(n)$ is the rotation matrix of frame $\{T\}$ with respect to $\{I\}$, $\boldsymbol{v}_t(t) \in \mathbb{R}^n$, $\boldsymbol{\omega}_t(t) \in \mathbb{R}^{n(n-1)/2}$ are the linear and angular target velocities and $\nabla \equiv \partial/\partial\gamma_i$ is the derivative with respect to γ_i.

Assumption 2.1. *The geometric path $\boldsymbol{p}^t_{d,i}(\gamma_i)$ is a differentiable function.*

Note that Assumption 2.1 is already needed in order to compute (4) from (3). Suppose we wish to control the position of the nose of the i-th vehicle, or more generically, a point $\bar{\boldsymbol{p}}_i = \boldsymbol{p}_i + R_i\, \boldsymbol{\epsilon}$ placed at a constant position $\boldsymbol{\epsilon} = [\epsilon_1\ \epsilon_2\ ]^\mathsf{T}$ $(n = 2)$ or $\boldsymbol{\epsilon} = [\epsilon_1\ \epsilon_2\ \epsilon_3\ ]^\mathsf{T}$ $(n = 3)$ from the origin of $\{B_i\}$. Then, define the MPF error associated to the i-th vehicle as the vector

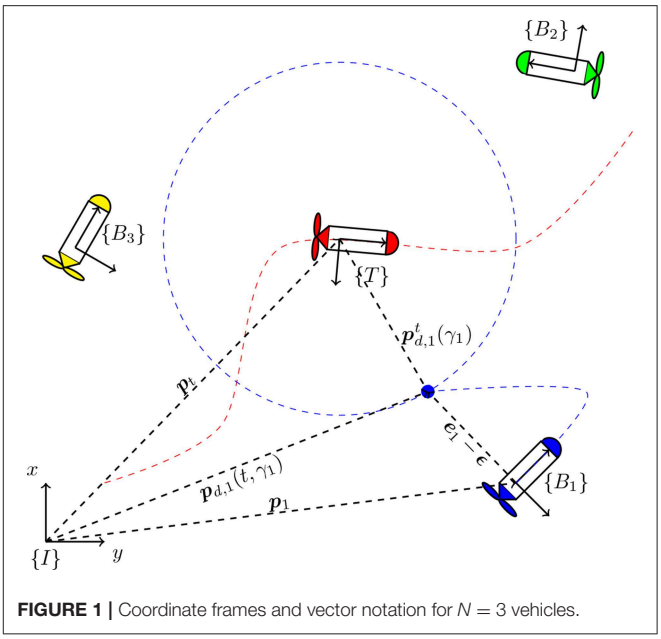


FIGURE 1 | Coordinate frames and vector notation for $N = 3$ vehicles.

$$\boldsymbol{e}_i = R_i^\mathsf{T} (\bar{\boldsymbol{p}}_i - \boldsymbol{p}_{d,i}), \quad i \in \mathcal{I} \tag{5}$$

The objective of the MPF control problem is to design a control law $\boldsymbol{u}_i$ such that the origin $\boldsymbol{e}_i \equiv 0$ is stable and $\boldsymbol{e}_i \to 0$ as $t \to \infty$, $\forall i \in \mathcal{I}$. That is, it is desired to steer the vehicles toward their moving geometric paths, such that $\bar{\boldsymbol{p}}_i$ stabilizes around $\boldsymbol{p}_{d,i}(\gamma_i, t)$, $\forall i \in \mathcal{I}$.

In order to control the progression of the virtual points $\boldsymbol{p}_{d,i}(t, \gamma_i)$ along the moving paths, the dynamics of the path variable $\dot{\gamma}_i$ should be explicitly controlled. This can be achieved by imposing the dynamics for γ_i as

$$\dot{\gamma}_i = v_d + \vartheta_i, \quad i \in \mathcal{I} \tag{6}$$

where the scalar v_d is the desired nominal speed of the path variable and ϑ_i is a bounded control signal, designed to achieve CMPF objectives such as: (i) consensus over the path variables of the robotic vehicles to achieve a desired formation along the moving path and (ii) faster convergence to the moving path. To move along the geometric paths with the desired velocity, the vehicles must satisfy the desired speed assignments $|\dot{\gamma}_i - v_d| \to 0$ as $t \to \infty$, $\forall i \in \mathcal{I}$.

2.2.2. Cooperative Motion Control Problem

Assume that the i-th vehicle communicates with a fixed set $\mathcal{N}_i \subset \mathbb{N}$ of neighbor vehicles. Given the path variables γ_i $(i \in \mathcal{I})$ for the N vehicles and a given undirected, fixed communication topology among them, the objective of the cooperative motion control problem is to design a decentralized control law $v_{r,i}(t)$ such that the positions of the virtual points are synchronized,

that is, $|\gamma_i - \gamma_j|$ converges to zero $\forall i,j \in \mathcal{I}$ as $t \to \infty$. To simultaneously achieve the speed assignment, coordination objective and also two other secondary objectives, function ϑ_i in (6) is decomposed as

$$\vartheta_i = v_{r,i}(t) + g_{e,i}(t) + g_{\omega,i}(t) \tag{7}$$

where $v_{r,i}(t)$ is the cooperative control signal (to be designed) that is responsible for achieving consensus between the vehicles, while $g_{e,i}(t)$ and $g_{\omega,i}(t)$ represent secondary objectives, where $g_{e,i}(t)$ is an error correction term, responsible for delaying the evolution of the path variable in case of momentary vehicle failure, and $g_{\omega,i}(t)$ is a rotation correction term, responsible for canceling the rotational motion induced on the path by the rotation of the target. These functions are to be properly defined in section 3.3.

3. ROBUST COOPERATIVE MOVING PATH FOLLOWING CONTROL

3.1. Robust MPF Controller Design

In this section, we consider the kinematic controller proposed in Jain et al. (2018a) with a modification designed to ensure robustness against disturbances. For the i-th vehicle, the error dynamics is given by

$$\dot{\boldsymbol{e}}_i = \dot{R}_i^{\mathsf{T}}(\bar{\boldsymbol{p}}_i - \boldsymbol{p}_{d,i}) + R_i^{\mathsf{T}}(\dot{\bar{\boldsymbol{p}}}_i - \dot{\boldsymbol{p}}_{d,i})\,.$$

Using model (1) with control signal (2) and MPF error (5), the error dynamics can be rewritten as

$$\begin{aligned}\dot{\boldsymbol{e}}_i \;=\; & -S(\boldsymbol{\omega}_i + \boldsymbol{d}_{\omega,i})\,\boldsymbol{e}_i \;+\; \Delta\boldsymbol{u}_i + \boldsymbol{d}_i - R_i^{\mathsf{T}}\,\boldsymbol{v}_t \\ & -R_i^{\mathsf{T}} R_t \left(\nabla \boldsymbol{p}_{d,i}^t\,\dot{\gamma}_i + S(\boldsymbol{\omega}_t)\,\boldsymbol{p}_{d,i}^t(\gamma_i)\right)\end{aligned} \tag{8}$$

where Δ is a constant matrix that can take the forms

$$\Delta = \begin{bmatrix} 1 & -\epsilon_2 \\ 0 & \epsilon_1 \end{bmatrix} \quad \text{or} \quad \Delta = \begin{bmatrix} 1 & 0 & \epsilon_3 & -\epsilon_2 \\ 0 & -\epsilon_3 & 0 & \epsilon_1 \\ 0 & \epsilon_2 & -\epsilon_1 & 0 \end{bmatrix}$$

for the planar ($n = 2$) and 3D ($n = 3$) cases, respectively. Note that it is always possible to choose ϵ such that Δ is full rank. Vector $\boldsymbol{d}_i \in \mathbf{R}^n$ is the total external disturbance acting on the vehicle. In the planar case, it is given by

$$\begin{aligned}\boldsymbol{d}_i &= \begin{bmatrix} R_i^{\mathsf{T}} & \boldsymbol{s}_\epsilon \end{bmatrix} \begin{bmatrix} \boldsymbol{d}_{v,i} \\ \boldsymbol{d}_{\omega,i} \end{bmatrix} \\ \boldsymbol{s}_\epsilon &= \begin{bmatrix} -\epsilon_2 & \epsilon_1 \end{bmatrix}^{\mathsf{T}}\end{aligned} \tag{9}$$

Remark 3.1. *Notice that, by the triangle inequality, the total external disturbance $\boldsymbol{d}_i$ is bounded by $\|\boldsymbol{d}_i\| \le \|\boldsymbol{d}_{v,i}\| + \|\boldsymbol{d}_{\omega,i}\|\,\|\boldsymbol{\epsilon}\|$.*

Assumption 3.1. *The total external disturbances $\boldsymbol{d}_i$ are bounded vector quantities.*

Theorem 1 (**Robust MPF**). *Consider an underactuated robotic vehicle described by (1) with control signal given by (2). Let the MPF error kinematics be described by (8), and consider that the pose of the i-th vehicle $\{\boldsymbol{p}_i, R_i\} \in \mathbb{R}^n \times \mathcal{SO}(n)$ and of the target frame $\{\boldsymbol{p}_t, R_t\} \in \mathbb{R}^n \times \mathcal{SO}(n)$ are known. Under Assumptions 2.1 and 3.1, the control law*

$$\begin{aligned}\boldsymbol{u}_i \;=\; & \Delta^{\dagger}\Big(-K_{p,i}\,\boldsymbol{e}_i + R_i^{\mathsf{T}}\left(\widehat{\boldsymbol{v}}_t + R_t\,S(\widehat{\boldsymbol{\omega}}_t)\,\boldsymbol{p}_{d,i}^t\right) \\ & + R_i^{\mathsf{T}} R_t \nabla \boldsymbol{p}_{d,i}^t \dot{\gamma}_i - \boldsymbol{w}_i\Big)\end{aligned} \tag{10}$$

$$\boldsymbol{w}_i \;=\; \begin{cases} \rho_i\,\frac{\boldsymbol{e}_i}{\|\boldsymbol{e}_i\|}, & \|\boldsymbol{e}_i\| \ge \epsilon_w \\ \rho_i\,\frac{\boldsymbol{e}_i}{\epsilon_w}, & \|\boldsymbol{e}_i\| < \epsilon_w \end{cases} \tag{11}$$

ensures that all trajectories of the MPF error are globally uniformly ultimately bounded and converge to a ball around the origin $\boldsymbol{e}_i = 0$ that can be made arbitrarily small. In (10), the matrix $\Delta^{\dagger}$ is the Moore-Penrose pseudo-inverse of Δ, $K_p \in \boldsymbol{R}^{n\times n}$ is a positive-definite gain matrix and $\widehat{\boldsymbol{v}}_t \in \mathbb{R}^n$, $\widehat{\boldsymbol{\omega}}_t \in \mathbb{R}^{n(n-1)/2}$ are estimates of the target velocities. In (11), ρ_i is a scalar such that

$$\rho_i \ge \|\boldsymbol{d}_{v,i}\| + \|\boldsymbol{d}_{\omega,i}\|\|\boldsymbol{\epsilon}\| + \|\widetilde{\boldsymbol{v}}_t\| + \|\widetilde{\boldsymbol{\omega}}_t \times \boldsymbol{p}_{d,i}^t(\gamma_i)\| \tag{12}$$

where $\widetilde{\boldsymbol{v}}_t = \boldsymbol{v}_t - \widehat{\boldsymbol{v}}_t$, $\widetilde{\boldsymbol{\omega}}_t = \boldsymbol{\omega}_t - \widehat{\boldsymbol{\omega}}_t$ are bounded estimation errors on the target velocities.

Proof: Define the Lyapunov candidate $V(\boldsymbol{e}_i) = \frac{1}{2}\boldsymbol{e}_i^{\mathsf{T}}\boldsymbol{e}_i$. Using the error dynamics in (8), its time derivative along the system trajectories is

$$\dot{V}(\boldsymbol{e}_i) = \boldsymbol{e}_i^{\mathsf{T}}\Big(\Delta\boldsymbol{u}_i + \boldsymbol{d}_i - R_i^{\mathsf{T}}\boldsymbol{v}_t - R_i^{\mathsf{T}}R_t\left(\nabla\boldsymbol{p}_{d,i}^t\,\dot{\gamma}_i + S(\boldsymbol{\omega}_t)\,\boldsymbol{p}_{d,i}^t\right)\Big) \tag{13}$$

where we have used the fact that $\boldsymbol{e}_i^{\mathsf{T}}\,S(\boldsymbol{\omega}_i + \boldsymbol{d}_{\omega,i})\,\boldsymbol{e}_i = 0$, since $S(\boldsymbol{\omega}_i + \boldsymbol{d}_{\omega,i})$ is skew-symmetric. Substituting control law (10) in (13) yields

$$\dot{V}(\boldsymbol{e}_i) = -\boldsymbol{e}_i^{\mathsf{T}}\,K_p\,\boldsymbol{e}_i + \boldsymbol{e}_i^{\mathsf{T}}\,(\boldsymbol{D}_i - \boldsymbol{w}_i)\;, \tag{14}$$

where $\boldsymbol{D}_i = \boldsymbol{d}_i - R_i^{\mathsf{T}}\left(\widetilde{\boldsymbol{v}}_t + R_t\,S(\widetilde{\boldsymbol{\omega}}_t)\,\boldsymbol{p}_{d,i}^t\right)$. Since $K_p > 0$, the first term is negative definite and bounded by $-\lambda_{\min}(K_p)\|\boldsymbol{e}_i\|^2$. Next, we consider the two cases of (11), when $\|\boldsymbol{e}_i\| \ge \epsilon_w$ or $\|\boldsymbol{e}_i\| < \epsilon_w$.

- For $\|\boldsymbol{e}_i\| \ge \epsilon_w$ in (14), we have

$$\begin{aligned}\dot{V}(\boldsymbol{e}_i) &\le -\lambda_{\min}(K_p)\|\boldsymbol{e}_i\|^2 + \boldsymbol{e}_i^{\mathsf{T}}\boldsymbol{D}_i - \rho_i\frac{\boldsymbol{e}_i^{\mathsf{T}}\boldsymbol{e}_i}{\|\boldsymbol{e}_i\|} \\ &\le -\lambda_{\min}(K_p)\|\boldsymbol{e}_i\|^2 + \|\boldsymbol{e}_i\|\,(\|\boldsymbol{D}_i\| - \rho_i)\end{aligned}$$

  where the Cauchy-Schwarz inequality was employed on term $\boldsymbol{e}_i^{\mathsf{T}}\boldsymbol{D}_i$. By Assumption 3.1, it is always possible to design ρ_i such that (12) is satisfied. Therefore, by Remark 3.1, choosing

$\rho_i \geq \|\boldsymbol{D}_i\|$ renders the second term on the right-hand side negative definite, which estabilishes that the trajectory $\boldsymbol{e}_i(t)$ of the closed-loop system reaches the ball $\mathcal{B}(\epsilon_w) := \{\boldsymbol{e}_i \in \mathbb{R}^n : \|\boldsymbol{e}_i\| \leq \epsilon_w\}$ in finite time.

- When the trajectories are inside $\mathcal{B}(\epsilon_w)$, we have $\|\boldsymbol{e}_i\| < \epsilon_w$, and (14) gets

$$\begin{aligned}\dot{V}(\boldsymbol{e}_i) &\leq -\lambda_{\min}(K_p)\|\boldsymbol{e}_i\|^2 + \boldsymbol{e}_i^{\mathsf{T}}\boldsymbol{D}_i - \rho_i\frac{\boldsymbol{e}_i^{\mathsf{T}}\boldsymbol{e}_i}{\epsilon_w}\\ &\leq -(1-\theta)\lambda_{\min}(K_p)\|\boldsymbol{e}_i\|^2\\ &\quad -\left(\theta\lambda_{\min}(K_p) + \frac{\rho_i}{\epsilon_w}\right)\|\boldsymbol{e}_i\|^2 + \|\boldsymbol{e}_i\|\|\boldsymbol{D}_i\|\end{aligned}$$

where $0 < \theta < 1$. Then, using the inequality above, one can write:

$$\begin{aligned}\dot{V}(\boldsymbol{e}_i) &\leq -(1-\theta)\lambda_{\min}(K_p)\|\boldsymbol{e}_i\|^2 < 0 \quad \forall\|\boldsymbol{e}_i\| \geq \mu_i\,,\\ \mu_i &= \frac{\|\boldsymbol{D}_i\|\epsilon_w}{\lambda_{\min}(K_p)\,\theta\,\epsilon_w + \rho_i}\end{aligned}$$

Note that $\mu_i \leq \epsilon_w$ for all $0 < \theta < 1$, which means that the trajectory of the closed-loop system $\boldsymbol{e}_i(t)$ again reaches the ball $\mathcal{B}(\mu_i) \subseteq \mathcal{B}(\epsilon_w)$ in finite time.

This establishes that the trajectories are globally ultimately uniformly bounded, since $V = \frac{1}{2}\|\boldsymbol{e}_i\|^2$ is radially unbounded (Khalil, 2002). Moreover, $\boldsymbol{e}_i(t)$ converges to the ball $\mathcal{B}(\mu_i) \subseteq \mathcal{B}(\epsilon_w)$, which can be made arbitrarily small when $\epsilon_w \to 0$. □

3.2. Robust MPF Controller Design With Disturbance Observer

In the presence of large amplitude disturbances, it may be difficult to tune the parameters ρ_i and ϵ_w so as to satisfy (12). In these situations, an observer can be designed to provide an estimate of the disturbance. Furthermore, this estimate can be used in the control law to compensate the real disturbance directly.

Without loss of generality, consider the planar problem. Consider that the vehicle pose $\{\boldsymbol{p}_i, R_i\} \in \mathbb{R}^2 \times \mathcal{SO}(2)$ is known and that the vehicle orientation is parameterized by the planar angle $\psi_i \in \mathbb{R}$, such that $R_i = R_i(\psi_i) \in \mathcal{SO}(2)$.

Then, the disturbance observer for the translational disturbance is defined as

$$\begin{cases}\dot{\hat{\boldsymbol{p}}}_i = R_i\,\boldsymbol{v}_i + \hat{\boldsymbol{d}}_{v,i} + K_1\,\tilde{\boldsymbol{p}}_i\\ \dot{\hat{\boldsymbol{d}}}_{v,i} = K_2\,\tilde{\boldsymbol{p}}_i\,,\end{cases} \tag{15}$$

where the estimation errors are defined as $\tilde{\boldsymbol{p}}_i = \boldsymbol{p}_i - \hat{\boldsymbol{p}}_i$ and $\tilde{\boldsymbol{d}}_{v,i} = \boldsymbol{d}_{v,i} - \hat{\boldsymbol{d}}_{v,i}$, and the positions $\boldsymbol{p}_i$, $i \in \mathcal{I}$ are accurately measured. For positive-definite matrices $K_1, K_2 \in \mathbb{R}^{2\times 2}$, the dynamics of the estimation errors $\tilde{\boldsymbol{p}}_i$, $\tilde{\boldsymbol{d}}_i$ can be proven to be Input-to-State Stable (ISS) with respect to the first time-derivative of $\boldsymbol{d}_{v,i}$ (Aguiar and Pascoal, 2002).

Similarly, observers for the rotational disturbances $d_{\omega,i} \in \mathbb{R}$ can be designed as:

$$\begin{cases}\dot{\hat{\psi}}_i = \omega_i + \hat{d}_{\omega,i} + k_{\omega_1}\tilde{\psi}_i\\ \dot{\hat{d}}_{\omega,i} = k_{\omega_2}\tilde{\psi}_i\,,\end{cases} \tag{16}$$

where the estimation errors are defined as $\tilde{\psi}_i = \psi_i - \hat{\psi}_i$ and $\tilde{d}_{\omega,i} = d_{\omega,i} - \hat{d}_{\omega,i}$, and the planar angles ψ_i are measured. Again, for positive scalars $k_{\omega_1}, k_{\omega_2} \in \mathbb{R}_{>0}$, the dynamics of the estimation errors $\tilde{\psi}_i$, $\tilde{d}_{\omega,i}$ can be proven to be ISS with respect to the first time-derivative of $d_{\omega,i}$ (Aguiar and Pascoal, 2002).

Theorem 2 (**Robust MPF with Disturbance Observer).** *Consider an underactuated robotic vehicle described by (1) and control signal given by (2). Let the MPF error kinematics be described by (8), and consider that the pose of the vehicle $\{\boldsymbol{p}_i, R_i\} \in \mathbb{R}^n \times \mathcal{SO}(n)$ and of the target frame $\{\boldsymbol{p}_t, R_t\} \in \mathbb{R}^n \times \mathcal{SO}(n)$ are known. Under Assumptions 2.1 and 3.1, the control law*

$$\begin{aligned}\boldsymbol{u}_i = \Delta^{\dagger}\Big(&-K_{p,i}\,\boldsymbol{e}_i + R_i^{\mathsf{T}}\left(\widehat{\boldsymbol{v}}_t + R_t\,S(\widehat{\boldsymbol{\omega}}_t)\,\boldsymbol{p}_{d,i}^t\right)\\ &+R_i^{\mathsf{T}}R_t\nabla\boldsymbol{p}_{d,i}^t\dot{\gamma}_i - \boldsymbol{w}_i - \widehat{\boldsymbol{d}}_i\Big)\end{aligned} \tag{17}$$

ensures that all trajectories of the MPF error are globally uniformly ultimately bounded and converge to a ball around the origin $\boldsymbol{e}_i = 0$ that can be made arbitrarily small. In (17), matrix $\Delta^{\dagger}$ is the Moore-Penrose pseudo-inverse of Δ, $K_p \in \mathbb{R}^{n\times n}$ is a positive-definite gain matrix, $\widehat{\boldsymbol{v}}_t \in \mathbb{R}^n$ and $\widehat{\boldsymbol{\omega}}_t \in \mathbb{R}^{n(n-1)/2}$ are estimates of the target velocities and $\widehat{\boldsymbol{d}}_i$ is the total estimated external disturbance, which is a function of the states of the disturbance observers

$$\widehat{\boldsymbol{d}}_i = \begin{bmatrix}R_i^{\mathsf{T}} & \boldsymbol{s}_\epsilon\end{bmatrix}\begin{bmatrix}\widehat{\boldsymbol{d}}_{v,i}\\ \widehat{d}_{\omega,i}\end{bmatrix} \tag{18}$$

The term $\boldsymbol{w}_i$ is defined by (11)*, with scalars ρ_i satisfying*

$$\rho_i \geq \|\widetilde{\boldsymbol{d}}_{v,i}\| + |\widetilde{d}_{\omega,i}|\|\boldsymbol{\epsilon}\| + \|\widetilde{\boldsymbol{v}}_t\| + \|\widetilde{\boldsymbol{\omega}}_t \times \boldsymbol{p}_{d,i}^t(\gamma_i)\| \tag{19}$$

Proof: The proof is very similar to Theorem 1, and can be performed by proposing the same Lyapunov candidate $V = \frac{1}{2}\boldsymbol{e}_i^{\mathsf{T}}\boldsymbol{e}_i$. Differentiating it in time and applying the error dynamics (8) with control law (10) yields

$$\dot{V}(\boldsymbol{e}_i) = -\boldsymbol{e}_i^{\mathsf{T}}\,K_p\,\boldsymbol{e}_i + \boldsymbol{e}_i^{\mathsf{T}}\,\left(\widetilde{\boldsymbol{D}}_i - \boldsymbol{w}_i\right) \tag{20}$$

where $\widetilde{\boldsymbol{D}}_i = \widetilde{\boldsymbol{d}}_i - R_i^{\mathsf{T}}\left(\widetilde{\boldsymbol{v}}_t + R_t\,S(\widetilde{\boldsymbol{\omega}}_t)\,\boldsymbol{p}_{d,i}^t\right)$ and $\widetilde{\boldsymbol{d}}_i$ is the total estimation error defined by $\widetilde{\boldsymbol{d}}_i = \boldsymbol{d}_i - \hat{\boldsymbol{d}}_i$.

Note that (20) is similar to (14), but with disturbance $\widetilde{\boldsymbol{D}}_i$ instead of $\boldsymbol{D}_i$. Therefore, using the same arguments for the proof of Theorem 1 with Assumption 3.1 and condition (19), one can conclude that the trajectories of the MPF error are

globally uniformly ultimately bounded and $\boldsymbol{e}_i(t)$ converges to the ball $\mathcal{B}(\bar{\mu}_i) \subseteq \mathcal{B}(\epsilon_w)$, which can be made arbitrarily small when $\epsilon_w \to 0$. □

Remark 3.2. *Comparing conditions (12) and (19) for the choice of ρ_i, in (19) the gain ρ_i must overcome only the norm of the disturbance estimation errors instead of the norm of the disturbance. Therefore, if the disturbance observer is properly designed, this method can reduce the necessary amount of control effort when compared to the previous method.*

Remark 3.3. *Both proposed control laws (10, 17) employ estimates of the target velocities. Since the velocity estimation errors $\widetilde{\boldsymbol{v}}_t$ and $\widetilde{\boldsymbol{\omega}}_t$ appear as additional disturbances in $\boldsymbol{D}_i$ and $\widetilde{\boldsymbol{D}}_i$, they can be properly compensated by the proposed controllers as long as ρ_i satisfies (12) or (19). In this case, the velocity estimation errors are implicitly assumed to be bounded. Furthermore, notice that in the case where no velocity estimators are employed ($\widehat{\boldsymbol{v}}_t = \boldsymbol{0}$ and $\widehat{\boldsymbol{\omega}}_t = \boldsymbol{0}$), the velocity estimation errors are simply $\widetilde{\boldsymbol{v}}_t = \boldsymbol{v}_t$ and $\widetilde{\boldsymbol{\omega}}_t = \boldsymbol{\omega}_t$, which are also bounded. These observations imply that velocity estimators are not necessarily required for the implementation of the proposed control laws. However, large velocity estimation errors would increase the lower bounds for the design of ρ_i, increasing the amount of control effort, which could lead to loss of performance.*

3.3. Cooperative Moving Path Following

This section provides a proper design for function ϑ_i in (7). First, the design of the error correction term $g_{e,i}(t)$ and of the rotation correction term $g_{\omega,i}(t)$ are discussed, and finally we make use of the results from Olfati-Saber et al. (2007) to design a cooperative control law $v_{r,i}(t)$.

3.3.1. Error Correction Term

The term $g_{e,i}(t)$ is a bounded error correction term that acts as an external input to the path dynamics, enabling faster convergence of the robotic vehicle to the moving path. It can be designed to delay or to stop the motion of the virtual point if the vehicle is too far away from the path. This can be done by defining the gradient with respect to the path variable of the MPF error norm squared:

$$\eta_{e,i} = \nabla\left(\frac{1}{2}\boldsymbol{e}_i^{\mathsf{T}}\boldsymbol{e}_i\right) = -\boldsymbol{e}_i^{\mathsf{T}} R_i^{\mathsf{T}} R_t \nabla \boldsymbol{p}_{d,i}^t(\gamma_i) \tag{21}$$

and then choosing a gradient descent law $g_{e,i} = -k_{e,i}\, sat(\eta_{e,i})$ with $k_{e,i} > 0$. The saturation function guarantees the boundedness for the correction term. Its effect is to effectively delay the evolution of the virtual point along the path by explicitly avoiding the evolution of γ_i if the MPF error norm is too large.

3.3.2. Path Rotation Correction Term

The term $g_{\omega,i}(t)$ is designed to delay the evolution of the virtual point $\boldsymbol{p}_{d,i}$ in a such a way that minimizes the effect of the target rotational motion, which is evident from the term $S(\boldsymbol{\omega}_t)\,\boldsymbol{p}_{d,i}^t(\gamma_i)$ in (4). This effect is important since, for large target angular velocities $\boldsymbol{\omega}_t$, the virtual point could move faster than the i-th vehicle could reach. Therefore, substituting (7) into the error dynamics (8), we seek to design a scalar $g_{\omega,i}$ such that

$$g_{\omega,i}(t) = \arg\min_{g_i\in\mathbb{R}} \|\nabla \boldsymbol{p}_{d,i}^t(\gamma_i)\, g_i + S(\boldsymbol{\omega}_t)\,\boldsymbol{p}_{d,i}^t(\gamma_i)\|\,. \tag{22}$$

If the target angular velocity is known, the minimum can be achieved by the least squares solution

$$g_{\omega,i}(\boldsymbol{\omega}_t, \gamma_i) = -\frac{\nabla^{\mathsf{T}}\boldsymbol{p}_{d,i}^t\, S(\boldsymbol{\omega}_t)\,\boldsymbol{p}_{d,i}^t}{\|\nabla \boldsymbol{p}_{d,i}^t\|^2}\,, \tag{23}$$

with minimum given by

$$\min_{g_i\in\mathbb{R}} \|\nabla \boldsymbol{p}_{d,i}^t(\gamma_i)\, g_i + S(\boldsymbol{\omega}_t)\,\boldsymbol{p}_{d,i}^t(\gamma_i)\| = \frac{(\nabla \boldsymbol{p}_{d,i}^t)^{\mathsf{T}}\boldsymbol{p}_{d,i}^t}{\|\nabla \boldsymbol{p}_{d,i}^t\|^2} S(\boldsymbol{\omega}_t)\,\nabla \boldsymbol{p}_{d,i}^t\,.$$

Remark 3.4. *Note that the minimum is identically null regardless the rotational motion of the target only if and only if: (i) the path is perpendicular to its gradient everywhere, i.e., $(\nabla \boldsymbol{p}_{d,i}^t)^{\mathsf{T}}\boldsymbol{p}_{d,i}^t = 0\ \forall\gamma_i$, $i \in \mathcal{I}$ or (ii) the angular velocity of the target is collinear to the path gradient everywhere, i.e., $\boldsymbol{\omega}_t = c\,\nabla \boldsymbol{p}_{d,i}^t = 0\ \forall\gamma_i$, $i \in \mathcal{I}$ for some constant $c \in \mathbb{R}$. Clearly, condition (ii) never holds in the planar case ($n = 2$).*

Assumption 3.2. *The path gradients are non-vanishing everywhere, i.e., $\nabla \boldsymbol{p}_{d,i}^t(\gamma_i) \neq \boldsymbol{0}\,, \forall\gamma_i\,, i \in \mathcal{I}$.*

From (23) and Assumption 3.2, the error correction term is bounded by.

$$|g_{\omega,i}(t)| \leq \frac{\max_{\gamma_i}\|\boldsymbol{p}_{d,i}^t(\gamma_i)\|}{\min_{\gamma_i}\|\nabla \boldsymbol{p}_{d,i}^t(\gamma_i)\|}\,\|\boldsymbol{\omega}_t\|\,.$$

3.3.3. Cooperative Controller

Consider the distributed consensus law (Aguiar, 2017):

$$v_{r,i} = -k_{c,i}\sum_{j\in\mathcal{N}_i}(\gamma_i - \widehat{\gamma}_j^i)\,, \qquad \forall i \in \mathcal{I} \tag{24}$$

where $k_{c,i} > 0$ are consensus gains and $\widehat{\gamma}_j^i$ are estimates of the path variables of the neighbor vehicles ($\gamma_j\,, j \in \mathcal{N}_i$) running inside the i-th vehicle computer. Assuming that the frequency of communication is low, its reasonable to assume that $\widehat{\gamma}_j^i \neq \gamma_j\,, \forall t > 0$. Therefore, one can write $\widehat{\gamma}_j^i = \gamma_j - \widetilde{\gamma}_j^i$, where $\widetilde{\gamma}_j^i$ is a path variable estimation error.

Assumption 3.3. *Given a fixed, undirected communication topology between the vehicles, the i-th vehicle updates its path variable γ_i to its $j \in \mathcal{N}_i$ neighbors in a fixed frequency. Additionally, assume that no data package is lost during communication. Consequently, the path variable estimation errors $\widetilde{\gamma}_j^i\,, \forall i, j \in \mathcal{I}$ are always bounded.*

Define the vectors $\boldsymbol{\gamma} = \begin{bmatrix}\gamma_1 & \gamma_2 & \cdots & \gamma_N\end{bmatrix}^{\mathsf{T}}$, $\boldsymbol{g}_e = \begin{bmatrix}g_{e,1} & g_{e,2} & \cdots & g_{e,N}\end{bmatrix}^{\mathsf{T}}$, $\boldsymbol{g}_\omega = \begin{bmatrix}g_{\omega,1} & g_{\omega,2} & \cdots & g_{\omega,N}\end{bmatrix}^{\mathsf{T}}$ and $\boldsymbol{1}_N = \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^N$. Using (24) in (7) and stacking the dynamic equations, one can write

$$\dot{\boldsymbol{\gamma}} = v_d\boldsymbol{1}_N - K_c L\boldsymbol{\gamma} - K_c\widetilde{\boldsymbol{\gamma}} + \boldsymbol{g}_e + \boldsymbol{g}_\omega\,, \tag{25}$$

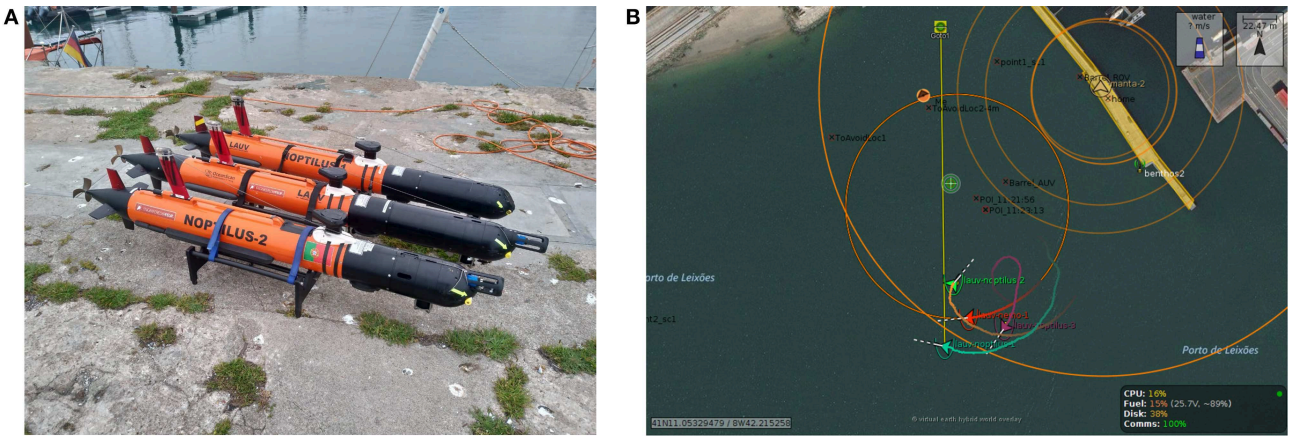


FIGURE 2 | (A) Autonomous underwater vehicles used on the experiments. **(B)** The Neptus console.

where $K_c = diag(k_{c,1}, k_{c,2}, \ldots, k_{c,N})$ is a positive definite matrix of consensus gains, $L = D - A \in \mathbb{R}^{N\times N}$ is the *Laplacian* of the network connection graph, defined by $D = diag(|\mathcal{N}_1|, |\mathcal{N}_2|, \ldots, |\mathcal{N}_N|)$ and the *adjacency matrix* $A = [a_{ij}]$, with $a_{ij} = 1$ if $j \in \mathcal{N}_i$ and $a_{ij} = 0$ otherwise. Vector $\widetilde{\boldsymbol{\gamma}}$ is defined as $[\widetilde{\boldsymbol{\gamma}}]_i = \sum_{j\in\mathcal{N}_i} \widetilde{\gamma}_j^i$, i.e., its i-th element is the sum of all path variable estimation errors for the i-th vehicle.

Theorem 3 (**Cooperative Controller).** *Consider a fleet of N underactuated robotic vehicles with dynamics described by (1) and control signal given by (2). Then, control laws (10) or (17) with robustness term (11) guarantee that the origin of the MPF error $\boldsymbol{e}_i \equiv 0$ is stable under the same conditions and assumptions of Theorems 1 and 2, respectively.*

Furthermore, under Assumption 3.3, the cooperative control law given by (24) ensures that $|\gamma_i - \gamma_j|, \forall i,j \in \mathcal{I}$ are Input-to-State Stable (ISS)[1] *with respect to the path variable estimation errors $[\widetilde{\boldsymbol{\gamma}}]_i$, error correction terms $g_{e,i}$ and rotation correction terms $g_{\omega,i}$, $\forall\, i \in \mathcal{I}$.*

Proof: The first part of the Theorem was already proved in Theorems 1 and 2. The part related to the cooperative control follows the same core ideas from (Jain et al., 2018a). First, define the *disagreement vector* (Olfati-Saber et al., 2007) as $\boldsymbol{\delta} := \boldsymbol{\gamma} - \alpha \mathbf{1}_N$, with $\alpha = (1/N)\mathbf{1}_N^\mathsf{T}\boldsymbol{\gamma}$.

Note that the consensus condition $|\gamma_i - \gamma_j| = 0, \forall i,j \in \mathcal{I}$ is achieved if and only if $\boldsymbol{\delta} = \mathbf{0}$. Additionally, the following two properties hold: (i) $L\boldsymbol{\gamma} = L\boldsymbol{\delta}$ and (ii) $\mathbf{1}_N^\mathsf{T}\boldsymbol{\delta} = 0$.

Next, define the ISS Lyapunov function candidate

$$V_{cc}(\boldsymbol{\delta}) = \boldsymbol{\delta}^\mathsf{T} L \boldsymbol{\delta} \geq 0$$

Taking its time-derivative and using (25), yields

$$\dot{V}_{cc} = -\boldsymbol{z}^\mathsf{T} K_c \boldsymbol{z} - \boldsymbol{z}^\mathsf{T} K_c \tilde{\boldsymbol{\gamma}} + \boldsymbol{z}^\mathsf{T} \boldsymbol{g}_e + \boldsymbol{z}^\mathsf{T} \boldsymbol{g}_\omega \tag{26}$$

with $\boldsymbol{z} = L\boldsymbol{\delta}$, where we used the properties (i) and (ii) introduced before. Using the Cauchy-Schwartz inequality, yields

$$\dot{V}_{cc} \leq -\lambda_{\min}(K_c)\|\boldsymbol{z}\|^2 + \lambda_{\max}(K_c)\|\boldsymbol{z}\|\|\widetilde{\boldsymbol{\gamma}}\| + \|\boldsymbol{z}\|\|\boldsymbol{g}_e\| + \|\boldsymbol{z}\|\|\boldsymbol{g}_\omega\| \tag{27}$$

Applying Young's inequality to the last three terms in (27), we have

$$\begin{aligned}\dot{V}_{cc} \leq & -\left(\lambda_{\min}(K_c) - \frac{\lambda_{\max}(K_c)}{2c} - \frac{1}{2c} - \frac{1}{2c}\right)\|\boldsymbol{z}\|^2 \\ & + \frac{c\lambda_{\max}(K_c)}{2}\|\widetilde{\boldsymbol{\gamma}}\|^2 + \frac{c}{2}\|\boldsymbol{g}_e\|^2 + \frac{c}{2}\|\boldsymbol{g}_\omega\|^2\end{aligned}$$

with a scalar $c \in \mathbb{R}_{>0}$. Choosing any $c > \frac{\lambda_{\max}(K_c)}{2\lambda_{\min}(K_c)} + \lambda_{\min}^{-1}(K_c) > 0$ leaves the first term of the right-hand side strictly negative, which by Assumption 3.3 and by the boundedness of $g_{e,i}, g_{\omega,i}$ establishes that the disagreement vector $\boldsymbol{\delta}$ is ISS with respect to the bounded disturbances $[\widetilde{\boldsymbol{\gamma}}]_i$, $g_{e,i}$ and $g_{\omega,i}$, for all $i \in \mathcal{I}$. □

[1] Khalil (2002) A nonlinear system $\dot{\boldsymbol{\delta}} = f(t, \boldsymbol{\delta}, \boldsymbol{\epsilon})$ is said to be Input-to-State Stable (ISS) if there exist a class $\mathcal{KL}$ function β and a class $\mathcal{K}$ function γ such that for any initial state $\boldsymbol{\delta}(t_0)$ and any bounded input $\boldsymbol{\epsilon}(t)$, the solution $\boldsymbol{\delta}(t)$ exists for all $t \geq t_0$ and satisfies

$$\|\boldsymbol{\delta}(t)\| \leq \beta(\|\boldsymbol{\delta}(t_0)\|, t - t_0) + \gamma\left(\sup_{t_0\leq\tau\leq t}\|\boldsymbol{\epsilon}(\tau)\|\right)$$

.

4. EXPERIMENTAL RESULTS

4.1. Experimental Setup

The experiments were performed on Porto de Leixões (Porto, Portugal) using three Light Autonomous Underwater Vehicles

(LAUVs) from the Underwater Systems and Technology Laboratory (LSTS) at the Faculty of Engineering of the University of Porto (FEUP) (**Figure 2A**). LAUVs are lightweight, portable vehicles that can be easily launched, operated and recovered with a minimal operational setup.

The vehicles operate under the DUNE/Neptus environments, which are part of a software toolchain (Pinto et al., 2013) developed and maintained by LSTS. DUNE is the on-board software running on the vehicles, comprising all the software needed for communications, navigation, control, maneuvering, plan execution and supervision of multiple types of robotic vehicles. The control algorithms were implemented on C++, using the available DUNE libraries. Neptus is a software used for command, control and monitoring, comprising many typical functions needed for a typical mission, such as planning, execution and post-mission analysis (**Figure 2B**).

A target vehicle was *simulated* and continuously sends its position and orientation (computed from GPS/IMU measurements using an extended Kalman filter (Braga et al., 2012) to the three follower vehicles through static UDP connections with a maximum frequency of 1 Hz. The control algorithm for the target vehicle is a vector field method (Nelson et al., 2007) that is responsible to steer the vehicle along a circumference with radius equal to 60 m in the clockwise direction at 0.5 m/s. The desired moving paths for the follower AUVs are planar circumference centered at the target vehicle with phase difference of $2\pi/3$ between them:

$$\boldsymbol{p}_{d,i}^{t}(\gamma_i) = R \begin{bmatrix} \cos(\gamma_i/R + \phi_i) \\ \sin(\gamma_i/R + \phi_i) \end{bmatrix}, \tag{28}$$

where $R = 25\,m$, $\phi_1 = 0\,rad$, $\phi_2 = 2\pi/3\,rad$ and $\phi_3 = -2\pi/3\,rad$. Each vehicle sends its path variable to the neighbor vehicles with a frequency of 1 Hz to maintain coordination, according to the consensus law (24) and Assumption 3.3. The consensus gains are $k_{c,i} = 0.1\,, \forall i \in \mathcal{I}$.

For the construction of the MPF errors $\boldsymbol{e}_i$, the value $\boldsymbol{\epsilon} = \begin{bmatrix} 1 & 0 \end{bmatrix}^{\mathsf{T}}$ was used. The controller gain matrices and error correction gains were chosen as $K_{p,i} = diag(0.2, 0.2)$ and $k_{e,i} = 2\,, \forall i \in \mathcal{I}$. The reference for the path variable velocity is $v_d = 1\,m/s$.

Remark 4.1. *We point out the fact that this particular kind of vehicles cannot generate reliable negative forward velocities due to its propeller design. Given the fact that control laws (10), (17) can generate negative forward velocities if the virtual point is behind the line-of-sight of the vehicle, a substitute controller was designed to override the original controller in case this happens.*

Therefore, while the forward velocity generated by (10) or (17) is negative ($v_{f,i} < 0$), the applied control signal will be

$$\boldsymbol{u}_i = \begin{bmatrix} v_C \\ -sgn([\boldsymbol{e}_i]_y - \epsilon_2)\,\omega_C \end{bmatrix}$$

instead, until (10), (17) generate a positive $v_{f,i}$ again. Constants $v_C, \omega_C \in \mathbb{R}$ are strictly positive. That means that the vehicle performs a "turning" maneuver with constant velocities until the virtual point is once again inside its line-of-sight. The direction of the turn is clockwise if the virtual point is to the right of the vehicle and counterclockwise if the virtual point is to the left of the vehicle. This strategy allows arbitrary initial configurations of the vehicles with respect to the initial position of the virtual point, and also allows the vehicles to recover from practical dead lock situations where their line-of-sight is kept facing away from the virtual point, which could happen, for example, in case of communication losses. In this case, $v_C = 1.7\,m/s$ and $\omega_C = 1\,rad/s$, approximately the upper saturation limits for the actuators.

4.2. Experimental Results

4.2.1. CMPF With Velocity Compensation

The first experiment shows the results of the CMPF controller with velocity compensation, $\rho_i = 0$ and no disturbance compensation ($\hat{\boldsymbol{d}}_{v,i} = 0$ and $\hat{d}_{\omega,i} = 0$). **Figure 3** shows the trajectories of the vehicles. The trajectory of the target is represented as the dashed black circle, in the clockwise direction. The small colored circles represent the beginning of the trajectory, while the colored asterisks represent its end. Noticeably, the three vehicles try to follow their respective paths (shown in dashed lines) around the rotating target, while maintaining their phase difference. **Figure 4** shows the obtained results. The initial position of the vehicles was distant from the network router (located closer to the northeast part of **Figure 3**), which affected the wireless communications for a while. However, the initially large path variable errors rapidly decrease and remain bounded to less than 4 m (**Figure 4B**). Because of the communication losses and possibly the presence of ocean currents, the secondary controller

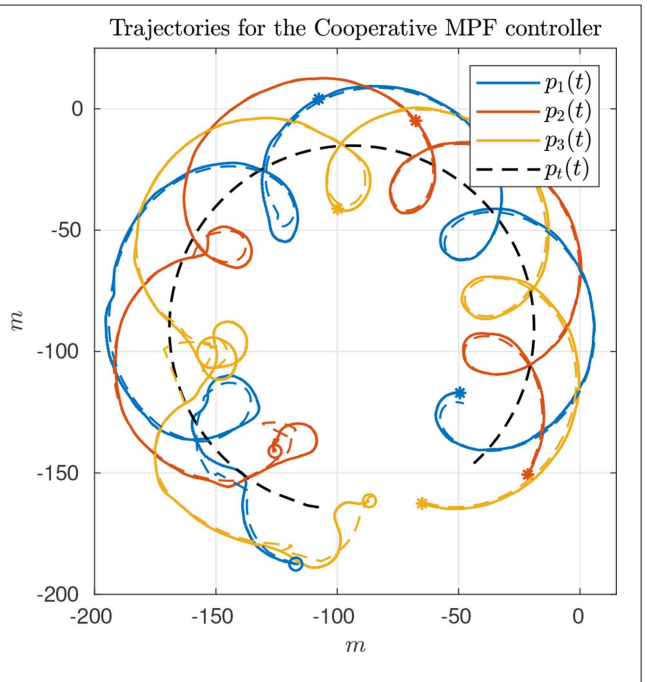


FIGURE 3 | Vehicle trajectories for the CMPF controller with velocity compensation.

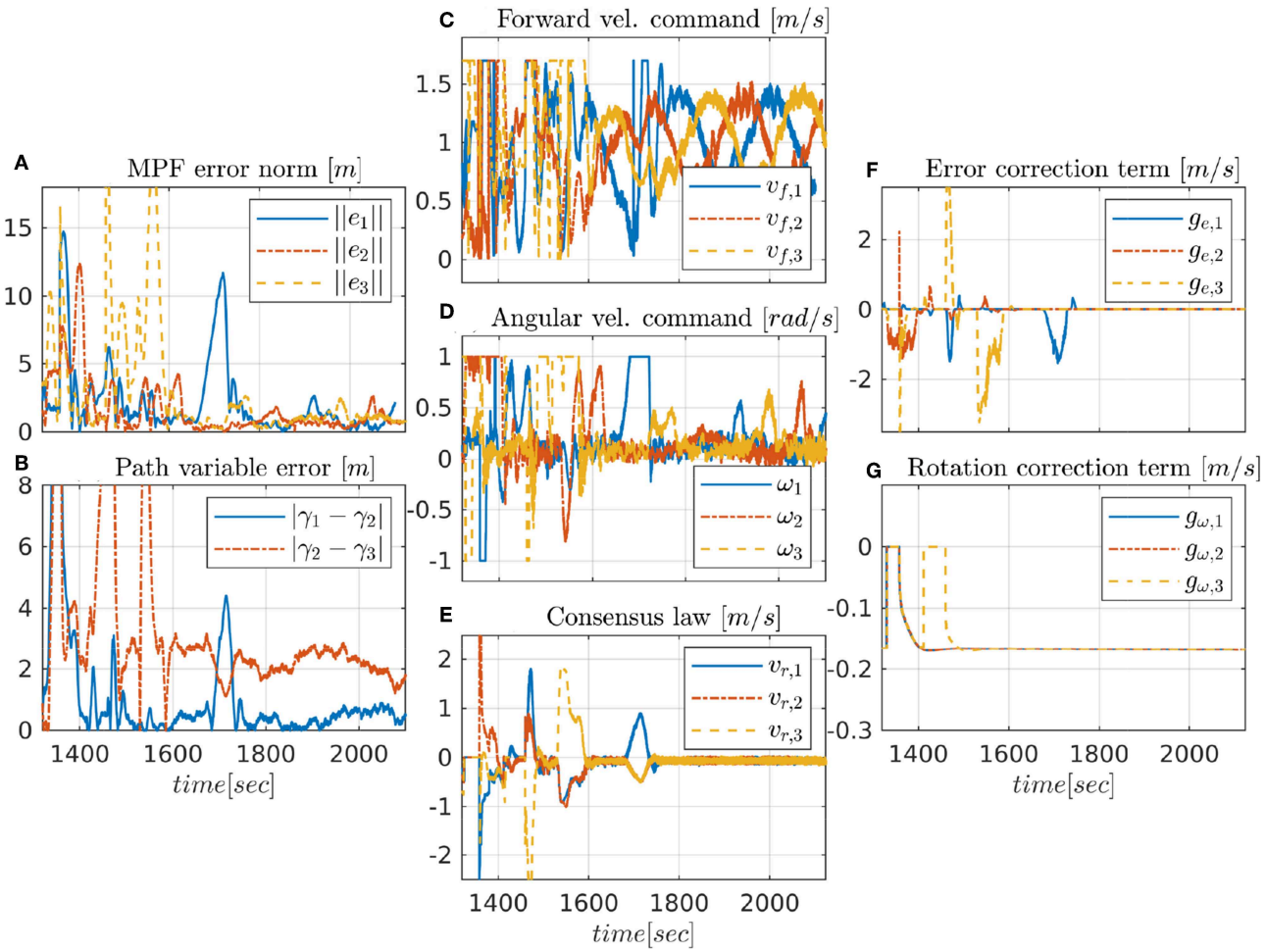


FIGURE 4 | Experimental results for the CMPF controller with velocity compensation.

described in Remark 4.1 had to recover some vehicles during the transient, resulting in some of the turning maneuvers we see in the beginning of the trajectories (**Figure 3**). After that transient, the norm of the MPF errors converge to a small region of less than 3 m while the control signal remains inside its linear region (**Figures 4C,D**). Note how the consensus law acts precisely when the path variable errors are high (**Figure 4E**), how the error correction terms acts when the MPF error norm is high (to prevent the evolution of the path variables), and how the rotation correction terms is fixed to a small value ($\approx 0.18\, m/s$) during the whole experiment. This is due to the fact that the target moves with constant angular velocity and the paths are circles to all three vehicles (see 23).

4.2.2. Robust CMPF With Sliding Mode Term

The second experiment shows the results of the robust CMPF controller with velocity compensation and Sliding Mode term, with MPF control law given by (10) with $\rho_i = 0.2$ for the three vehicles and $\epsilon_w = 0.5\, m$. The consensus law for cooperation among the vehicles is given by (24), as before.

Figure 5 shows the vehicle trajectories around the target, starting and ending in the southwest and southeast corners, respectively. Once more, due to communication losses and the presence of ocean currents in the southwest location, the secondary controller described in Remark 4.1 was activated for some of vehicles during the transient. However, the proposed controller was able to stabilize the error faster than the nominal controller. Besides, from **Figure 6A**, it is possible to notice the practical sliding mode phenomena around the origin $\boldsymbol{e}_i \equiv 0$. That means that the controller is able to achieve better performance than the previous one, given that ϵ_w can be designed to be arbitrarily small. However, from (11), small values of ϵ_w can result in higher gains for $\boldsymbol{w}_i$, which can potentially saturate the control inputs. In fact, sometimes the control saturation limits are reached after the transient, as shown in **Figures 6C,D**, and practical sliding mode is momentarily lost. The reason is the limited velocity range allowed by the actuators, combined with our particular value choice for ϵ_w, and moments of occasional increase in the target velocity. Even so, performance is slightly better than in the previous case, and the amount of control chattering is acceptable.

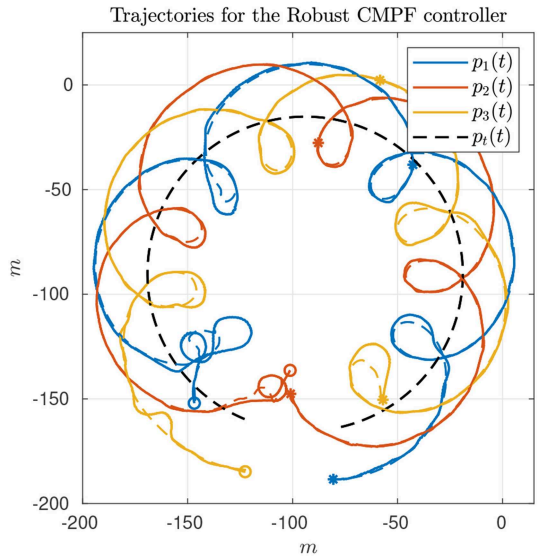


FIGURE 5 | Vehicle trajectories for the robust CMPF controller.

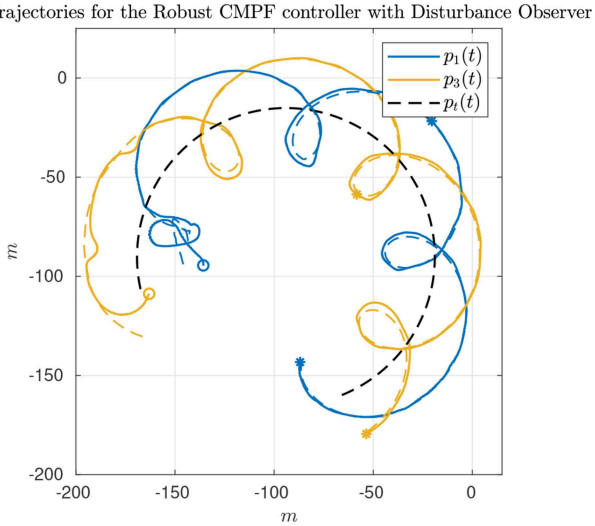


FIGURE 7 | Vehicle trajectories for the robust CMPF controller with disturbance observer.

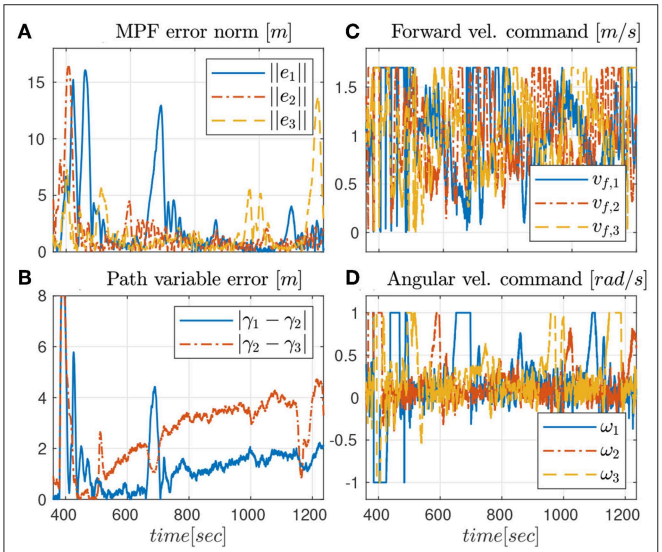


FIGURE 6 | Results for the robust CMPF controller.

The consensus law, error correction signals and rotation correction signals are omitted, but are similar to those observed in **Figure 4**.

4.2.3. Robust CMPF With Sliding Mode Term and Disturbance Compensation

The third and last experiment shows the results of the robust CMPF controller with velocity compensation, Sliding Mode term and direct disturbance compensation using a linear observer. The control law is given by (17) with $\rho_i = 0.2$ and $\epsilon_w = 0.5\,m$, as before. Again, the consensus law for cooperation among the vehicles is given by (24).

As seen from **Figure 7**, only the vehicles Noptilus 1 and 3 were used on this experiment, since the battery on Noptilus 2 was depleted. However, the results obtained by Noptilus 1 and 3 can still be compared to the previous results obtained for the same two vehicles. The chosen paths are the same circles defined in (28), but this time with $\phi_1 = 0$ and $\phi_3 = \pi\ rad$. This modification was used to guarantee that the two vehicles stay as far as possible from each another. Once again, in **Figure 8A**, notice the practical sliding mode phenomena around the origin $\boldsymbol{e}_i \equiv 0$, except during the instants where the control inputs are saturated (**Figures 8C,D**). However, in this case, the control chattering is significantly smaller than the one observed in **Figures 6C,D**, under the same experimental conditions. We explain this fact by the presence of the disturbance estimator. Since part of the disturbance is compensated, the sliding mode term can spend less effort compensating the remaining total disturbance, a result compatible with the theoretical insight of Remark 3.2. The path variable errors remain bounded by 4 m, as shown in **Figure 8B**. The estimated disturbances are shown in **Figure 9**. The linear velocity disturbances remained bounded by $< 0.3m\,s$ after the transient, while the angular velocity disturbances showed higher variation, but remained bounded to $< 0.5rad\,s$ after the transient.

5. CONCLUSIONS

This work addressed the robust cooperative MPF problem for marine vehicles. We demonstrated that the origin of the MPF errors associated to the vehicles are stable with the two proposed robust CMPF control schemes in the presence of bounded disturbances acting on the vehicles. Furthermore, it was theoretically demonstrated that the cooperative control scheme is ISS with respect to the path variable estimation errors and to two other bounded, auxiliary input variables, named error correction term and rotation correction terms. The proposed

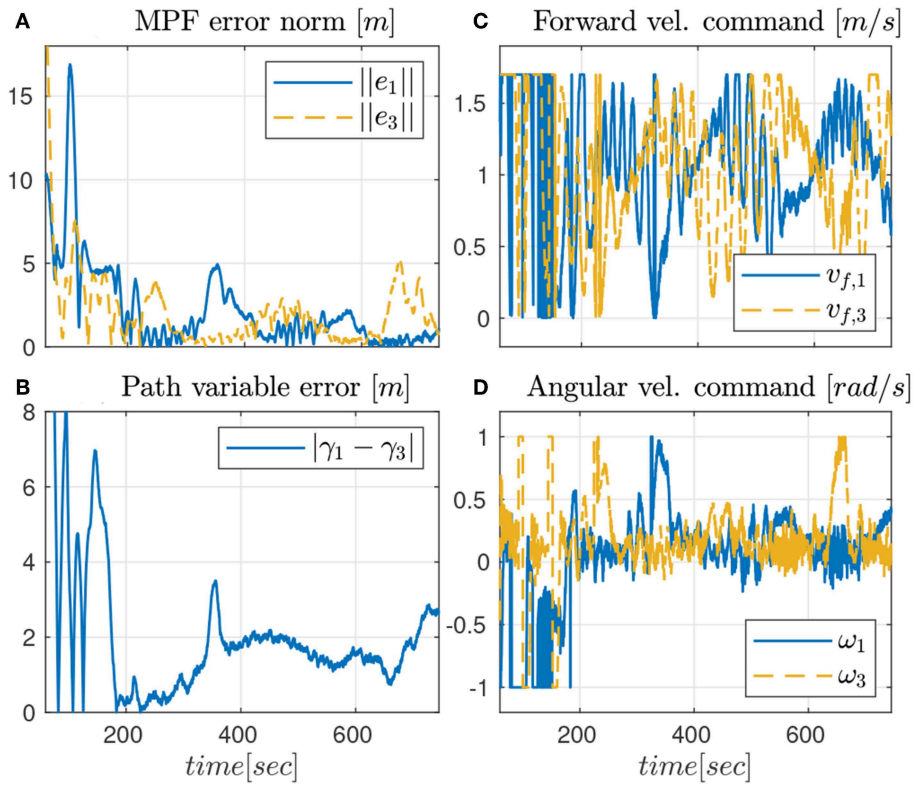


FIGURE 8 | Results for the robust CMPF controller with disturbance observer.

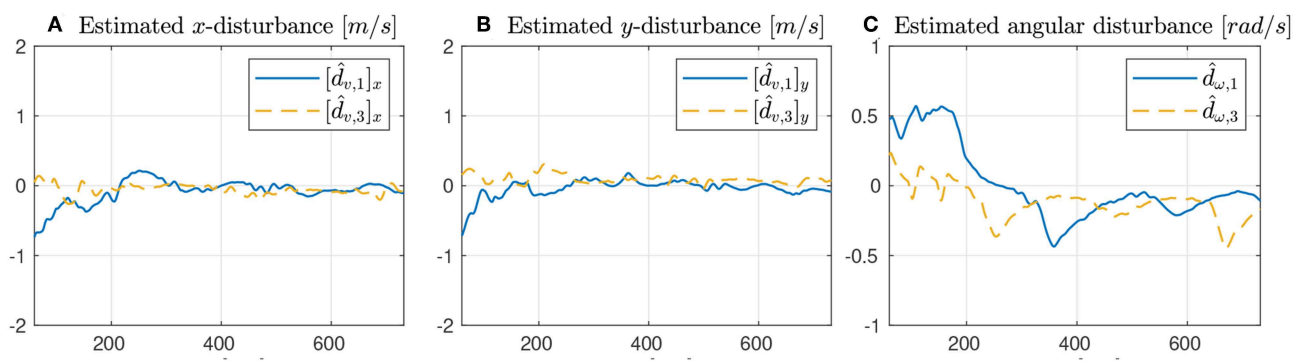


FIGURE 9 | Results obtained with the disturbance estimator.

robust controllers (10, 17) guarantee that the MPF error is globally uniformly bounded to a small neighborhood of the origin while maintaining acceptable control chattering. The narrow linear region of the actuators imposes limits on how small ϵ_w can be designed in practice. Lastly, we conclude that control law (17) actually improved the control chattering in practice, corroborating the theoretical insight of Remark 3.2.

Some of the future works are: (i) to investigate how to extend the proposed controllers to the case of unknown bounds for the disturbances (ii) to take the existence of actuator saturation limits in the control design and (iii) to incorporate obstacle avoidance techniques into the cooperative MPF approach to prevent vehicle collision during the cooperation tasks.

DATA AVAILABILITY STATEMENT

The datasets generated for this study are available on request to the corresponding author.

AUTHOR CONTRIBUTIONS

MR has written the manuscript, implemented the algorithms in C++ code, and performed the experiments using the LSTS vehicles. He also proposed the sliding mode based scheme for adding robustness to the moving path following controllers. RJ has proposed the cooperative control scheme using a consensus law, and contributed significantly to the stability proof of the

cooperative controller. He also helped by suggesting important changes on the code and with the organization of the manuscript. AA contributed with the proposition of the disturbance compensation method for improving the performance of the first controller (Theorem 2), and also strongly contributed to the stability proofs and overall organization of the paper. JS made the experiments possible by setting up the mission at Porto de Leixões and has contributed by suggesting some changes on the code.

FUNDING

This work was supported in part by projects POCI-01-0145-FEDER-031823 - IMPROVE, and POCI-01-0145-FEDER-031411 - HARMONY, both funded by FEDER funds through COMPETE2020 - POCI and by national funds (PIDDAC), and also by project POCI-01-0247-FEDER-024508 - OceanTech, approved through the Incentive Scheme R&TD Co-promotion Projects and co-funded by the European Regional Development Fund, supported by Portugal2020 through Compete2020.

ACKNOWLEDGMENTS

The authors would like to acknowledge the support of the LSTS staff in conducting the experiments on Porto de Leixões. The code used in the experiments was written in C++ and can be accessed from the GITHUB® account on https://github.com/CaipirUltron/dune. The video to the experiments can be found at https://www.youtube.com/watch?v=qFNnGjBHqPk.

REFERENCES

Aguiar, A. P. (2017). "Single and multiple motion control of autonomous robotic vehicles," in *2017 11th International Workshop on Robot Motion and Control (RoMoCo)* (Wasowo), 172–184. doi: 10.1109/RoMoCo.2017.8003910

Aguiar, A. P., and Hespanha, J. P. (2007). Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. *IEEE Trans. Automat. Control* 52, 1362–1379. doi: 10.1109/TAC.2007.902731

Aguiar, A. P., Hespanha, J. P., and Kokotovic, P. V. (2005). Path-following for nonminimum phase systems removes performance limitations. *IEEE Trans. Automat. Cont.* 50, 234–239. doi: 10.1109/TAC.2004.841924

Aguiar, A. P., and Pascoal, A. M. (2002). "Dynamic positioning and way-point tracking of underactuated auvs in the presence of ocean currents," in *Proceedings of the 41st IEEE Conference on Decision and Control, 2002*, Vol. 2 (Las Vegas, NV), 2105–2110.

Aguiar, A. P., and Pascoal, A. M. (2007). "Coordinated path-following control for nonlinear systems with logic-based communication," in *2007 46th IEEE Conference on Decision and Control* (New Orleans, LA), 1473–1479. doi: 10.1109/CDC.2007.4434835

Belleter, D., Paliotta, C., Maggiore, M., and Pettersen, K. (2016). "Path following for underactuated marine vessels. *IFAC-PapersOnLine* 49, 588–593. doi: 10.1016/j.ifacol.2016.10.229

Braga, J., Healey, A. J., and Sousa, J. (2012). Navigation scheme for the lsts seacon vehicles: theory and application. *IFAC Proc. Volumes* 45, 69–75. doi: 10.3182/20120410-3-PT-4028.00013

Cichella, V., Kaminer, I., Dobrokhodov, V., Xargay, E., Hovakimyan, N., and Pascoal, A. (2011). "Geometric 3D path-following control for a fixed-wing UAV on SO (3)," *in AIAA Guidance, Navigation, and Control Conference* (Portland, OR). doi: 10.2514/6.2011-6415

Dagci, O. H., Ogras, U. Y., and Ozguner, U. (2003). "Path following controller design using sliding mode control theory," in *Proceedings of the 2003 American Control Conference, 2003*, Vol. 1 (Denver, CO), 903–908.

Encarnação, P., Pascoal, A., and Arcak, M. (2000). Path following for marine vehicles in the presence of unknown currents1. *IFAC Proc. Volumes* 33, 507–512. doi: 10.1016/S1474-6670(17)37980-6

Gu, N., Peng, Z., Wang, D., Shi, Y., and Wang, T. (2019). Anti-disturbance coordinated path-following control of robotic autonomous surface vehicles: theory and experiment. *IEEE/ASME Trans. Mechatr.* 24 2386-2396. doi: 10.1109/TMECH.2019.2929216

Jain, R. P., Aguiar, A. P., and de Sousa, J. B. (2018a). Cooperative path following of robotic vehicles using an event-based control and communication strategy. *IEEE Robot. Automat. Lett.* 3, 1941–1948. doi: 10.1109/LRA.2018.2808363

Jain, R. P., Alessandretti, A., Aguiar, A. P., and De Sousa, J. B. (2018b). "Cooperative moving path following using event based control and communication," in *2018 13th APCA International Conference on Automatic Control and Soft Computing (CONTROLO)* (Ponta Delgada), 189–194.

Jain, R. P. K., Aguiar, A. P., Alessandretti, A., and Borges de Sousa, J. (2018c). "Moving path following control of constrained underactuated vehicles: a nonlinear model predictive control approach," in *AIAA SciTech Forum* (Kissimmee, FL: American Institute of Aeronautics and Astronautics).

Kapitanyuk, Y. A., de Marina, H. G., Proskurnikov, A. V., and Cao, M. (2017). Guiding vector field algorithm for a moving path following problem. *IFAC-PapersOnLine* 50, 6983–6988. doi: 10.1016/j.ifacol.2017.08.1340

Khalil, H. K. (2002). *Nonlinear Systems, 3rd Edn.* Upper Saddle River, NJ: Prentice-Hall.

Micaelli, A., and Samson, C. (1993). *Trajectory tracking for unicycle-type and two-steering-wheels mobile robots. Research Report RR-2097,* INRIA. Available online at: https://hal.inria.fr/inria-00074575/file/RR-2097.pdf

Nelson, D. R., Barber, D. B., McLain, T. W., and Beard, R. W. (2007). Vector field path following for miniature air vehicles. *IEEE Trans. Robot.* 23, 519–529. doi: 10.1109/TRO.2007.898976

Olfati-Saber, R., Fax, J. A., and Murray, R. M. (2007). Consensus and cooperation in networked multi-agent systems. *Proc. IEEE* 95, 215–233. doi: 10.1109/JPROC.2006.887293

Oliveira, T., Aguiar, A. P., and Encarnação, P. (2016). Moving path following for unmanned aerial vehicles with applications to single and multiple target tracking problems. *IEEE Trans. Robot.* 32, 1062–1078. doi: 10.1109/TRO.2016.2593044

Oliveira, T., Aguiar, A. P., and Encarnação, P. (2017). "Three dimensional moving path following for fixed-wing unmanned aerial vehicles," in *2017 IEEE International Conference on Robotics and Automation (ICRA)* (Singapore), 2710–2716.

Oliveira, T., and Encarnação, P. (2013). Ground target tracking control system for unmanned aerial vehicles. *J. Intell. Robot. Syst.* 69, 373–387. doi: 10.1007/s10846-012-9719-0

Pinto, J., Dias, P. S., Martins, R., Fortuna, J., Marques, E., and Sousa, J. (2013). "The LSTS toolchain for networked vehicle systems," in *2013 MTS/IEEE OCEANS-Bergen* (Bergen: IEEE), 1–9.

Reis, M. F., Jain, R. P., Aguiar, A. P., and de Sousa, J. B. (2019). Robust moving path following control for robotic vehicles: theory and experiments. *IEEE Robot. Automat. Lett.* 4, 3192–3199. doi: 10.1109/LRA.2019.2925733

Samson, C. (1992). "Path following and time-varying feedback stabilization of a wheeled mobile robot," in *Second International Conference on Automation, Robotics and Computer Vision* (Singapore), 3.

Wang, R., Jing, H., Hu, C., Yan, F., and Chen, N. (2016). Robust H_∞ path following control for autonomous ground vehicles with delay and data dropout. *IEEE Trans. Intell. Transport. Syst.* 17, 2042–2050. doi: 10.1109/TITS.2015.2498157

Xargay, E., Kaminer, I., Pascoal, A., Hovakimyan, N., Dobrokhodov, V., Cichella, V., et al. (2013). Time-critical cooperative path following of multiple uavs over time-varying networks. *J. Guid. Cont. Dyn.* 36, 499–516. doi: 10.2514/1.56538

Zhang, L.-J., Jia, H.-M., and Jiang, D.-P. (2014). Sliding mode prediction control for 3d path following of an underactuated auv. *IFAC Proc. Volumes* 47, 8799–8804. doi: 10.3182/20140824-6-ZA-1003.00372

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Robotics and AI

METHODS
published: 28 January 2020
doi: 10.3389/frobt.2020.00002

Marine Applications of the Fast Marching Method

Santiago Garrido, David Alvarez* and Luis E. Moreno*

Robotics Lab, Department of Systems and Automation Engineering, Universidad Carlos III de Madrid, Madrid, Spain

Path planning is general problem of mobile robots, which has special characteristics when applied to marine applications. In addition to avoid colliding with obstacles, in marine scenarios, environment conditions such as water currents or wind need to be taken into account in the path planning process. In this paper, several solutions based on the Fast Marching Method are proposed. The basic method focus on collision avoidance and optimal planning and, later on, using the same underlying method, the influence of marine currents in the optimal path planning is detailed. Finally, the application of these methods to consider marine robot formations is presented.

Keywords: fast marching, path planning, formations, vector field fast marching, trajectory planning

OPEN ACCESS

Edited by:
Enrica Zereik, Italian National Research Council (CNR), Italy

Reviewed by:
Fernando Gomez-Bravo, University of Huelva, Spain
Charalampos P. Bechlioulis, National Technical University of Athens, Greece

****Correspondence:***
Santiago Garrido
sgarrido@ing.uc3m.es
David Alvarez
dasanche@ing.uc3m.es

Specialty section:
This article was submitted to Robotic Control Systems, a section of the journal Frontiers in Robotics and AI

Received: *04 May 2019*
Accepted: *08 January 2020*
Published: *28 January 2020*

Citation:
Garrido S, Alvarez D and Moreno LE (2020) Marine Applications of the Fast Marching Method. Front. Robot. AI 7:2. doi: 10.3389/frobt.2020.00002

1. INTRODUCTION

Motion planning has been a very important field of research for many years. In the area of autonomous marine vehicles, both surface and underwater vehicles, some important aspects that are commonly optimized are travel time and safety conditions. This means that the path should avoid known obstacles and hazardous areas while reaching the goal pose as fast as possible.

An example of an approach using these concepts can be found in Bellingham and Willcox (1996), in which an underwater mission planning is proposed for optimizing energy consumption while guaranteeing spatio-temporal coverage. Following a similar goal, in Hert et al. (1996) the problem is formulated as a shortest path problem in order to guarantee the coverage of the terrain using a sonar system.

Besides, in marine environments, uncertainties due to the wind and water currents are complex and have a large impact on the path planning, as shown in Song et al. (2015). In order to deal with the environmental influence, a level set method based on the Fast Marching Method was proposed by Agarwal and Lermusiaux (2011). In Petres et al. (2005), an Anisotropic version of the Fast Marching Method (AFM) is used for submarine vehicles. This method provides collision free paths and their convergence is guaranteed, however, the water current model used does not take into account the power of the motor of the vehicle. Song et al. (2017) proposed an improvement of the AFM by using a multi-layered fast marching, which combines different environmental factors, such as currents and wind with attractive/repulsive maps. The proposed strategies deliver very interesting results, but do not guarantee the avoidance of local minima in the path planning due to the manner used to create the velocity maps.

The Fast Marching Method (*FMM*) and its evolution, known as the Fast Marching Square (FM^2), have proven their value for path planning applications and robot motion because of their plasticity and ease of use. They have been applied to many different path planning related problems such as: indoors and outdoors (Garrido et al., 2017) robot motion, path learning (Gomez et al., 2017) or unmanned aerial (Álvarez et al., 2015) and marine vehicles (Petres et al., 2005; Song et al., 2017). However, all these methods are based on a scalar model of the environment. If vector fields are present in the model, then the Fast Marching Method subjected to a Vector Field (FMVF) is a better choice to perform the path planning.

In the next sections of this article, an overview of how the Fast Marching Method (FMM) works, as well as several path planning versions based on the FMM are explained. Besides, their basic characteristics and their use in marine-like environments are shown.

2. THE EIKONAL EQUATION AND THE FAST MARCHING METHOD

The speed of light traversing different materials is defined as $v = c/n$, where v is the velocity in the specific medium, c = 300,000 m/s is the speed of light in vacuum and n is the refractive index which depends on the material that is traversed. For example, in water $n = 1.33$, while in glass $n = 1.5$, this difference provokes that when a ray of light passes from water to glass the ray changes its direction following the corresponding fastest path in each material. In cases in which there is a continuous change of refractive index, the path bends continuously, as in **Figure 1**. As it happens in a mirage in a hot road, the layers of air closest to the road are hotter than those that are further away. This creates a gradient of refractive indices that causes the rays coming from the sun to bend, therefore the driver has the optical illusion of seeing a kind of puddle of water on the road.

In general, the path that a ray of light follows (along any media) is the minimum in travel time. Therefore, the refractive index works as a viscosity or speed index that slows down the expansion of the light wave. Therefore, the path of a single ray of light among the wave expansion can be represented by its gradient.

One way to characterize the position of a front in expansion is to compute the arrival time, T, in which the front reaches each point of the space. For one dimension, the time of arrival value can be obtained simply considering that the traveled distance, x, is the product of the speed, F, and the time, T.

$$x = F \cdot T \tag{1}$$

Then, the one dimensional spatial derivative of this function is:

$$\frac{dT(x)}{dx} = \frac{1}{F(x)} \tag{2}$$

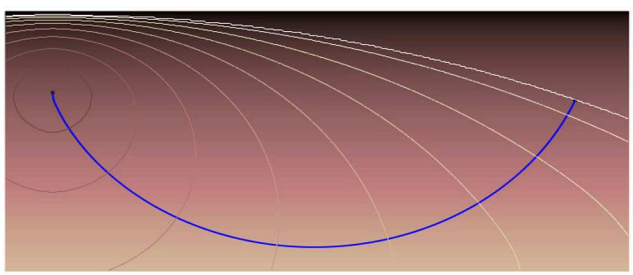


FIGURE 1 | The resulting path of the light when the refractive index changes continuously.

and therefore, the magnitude of the derivative of the arrival function $T(x)$ is inversely proportional to the speed.

When considering multiple dimensions, the same concept is valid and the solution is found by substituting the derivative by the gradient, since the gradient is orthogonal to the level sets of the arrival time function $T(x)$. In this way, the movement of the front of the wave can be characterized as the solution of a boundary condition problem. If the propagation speed depends only on the position, then equation 2 can be reformulated as the Eikonal equation:

$$\left|\nabla T(x)\right| F(x) = 1. \tag{3}$$

The Fast Marching Method (FMM) proposes a solution of the Eikonal equation for a grid map in which the velocity values at each point represent the refractive index. This artificial refractive index represents the cost function for the wave expansion. This method was originally proposed for a rectangular orthogonal mesh in Sethian (1996). As demonstrated in Yatziv et al. (2005), the FMM is an $O(n)$ algorithm where n is the total number of grid points. The algorithm relies on an upwind finite difference approximation to the gradient as a first order solution of the differential equation.

The FMM is used for problems in which the speed function never changes of sign, which means that the wave front always moves forwards (no reflections are admitted). This characteristic allows to use a stationary formulation, because the wave front crosses each grid point only once. The wave propagation given by the FMM represents a distance function that corresponds to the Geodesic distance measured with the metric defined by the refraction matrix. This matrix indicates the speed of the wave front at each point of the grid.

2.1. Algorithm Implementation on an Orthogonal Mesh

In general, the FMM can model any phenomena which evolves as a wave front that propagates along its normal direction. Let T_{ij} be the time at which the wave front crosses the point (i, j) of a 2-dimensional map, satisfying $|\nabla T|F = 1$, the Eikonal equation. F represents the speed function and, therefore, $F = F_{ij}$ represents the speed at each point of the map. As shown in Gómez et al. (2019), the most common first-order discretization of the Eikonal equation is given in Osher and Sethian (1988), which uses an upwind-difference scheme to approximate partial derivatives of $T(\mathbf{x})$ ($D_{\mathrm{ij}}^{\pm x}$ represents the one-sided partial difference operator in direction $\pm x$):

$$\begin{aligned} T_x(\mathbf{x}) &\approx D_{\mathrm{ij}}^{\pm x} T = \tfrac{T_{i\pm1,j} - T_{\mathrm{ij}}}{\pm \Delta_{\mathrm{x}}} \\ T_y(\mathbf{x}) &\approx D_{\mathrm{ij}}^{\pm y} T = \tfrac{T_{i,j\pm1} - T_{\mathrm{ij}}}{\pm \Delta_{\mathrm{y}}} \end{aligned} \tag{4}$$

A simple solution to Equation (4) is proposed in Sethian (1999):

$$\left\{ \begin{array}{c} \max(D_{\mathrm{ij}}^{-x} T, -D_{\mathrm{ij}}^{+x} T, 0)^2 + \\ \max(D_{\mathrm{ij}}^{-y} T, -D_{\mathrm{ij}}^{+y} T, 0)^2 \end{array} \right\} = \frac{1}{F_{\mathrm{ij}}^2} \tag{5}$$

in which Δx and Δy are the grid spacing in the x and y directions. Substituting (4) in (5) and letting

$$\begin{aligned} T &= T_{i,j} \\ T_{\mathrm{x}} &= \min(T_{i-1,j}, T_{i+1,j}) \\ T_{\mathrm{y}} &= \min(T_{i,j-1}, T_{i,j+1}) \end{aligned} \tag{6}$$

Then, for a discrete 2D space as, the Eikonal Equation can be written as:

$$\max\left(\frac{T-T_{\mathrm{x}}}{\Delta_{\mathrm{x}}}, 0\right)^2 + \max\left(\frac{T-T_{\mathrm{y}}}{\Delta_{\mathrm{y}}}, 0\right)^2 = \frac{1}{F_{\mathrm{ij}}^2} \tag{7}$$

Since the speed of the front is assumed to be positive ($F > 0$), T must be greater than T_{x} and T_{y} whenever the front wave has not already passed over the coordinates (i, j). Therefore, (7) can be simplified as:

$$\left(\frac{T-T_{\mathrm{x}}}{\Delta x}\right)^2 + \left(\frac{T-T_{\mathrm{y}}}{\Delta y}\right)^2 = \frac{1}{F_{\mathrm{ij}}^2} \tag{8}$$

Equation (8) is a regular quadratic equation of the form $aT^2 + bT + c = 0$, where:

$$\begin{aligned} a &= \Delta_{\mathrm{x}}^2 + \Delta_{\mathrm{y}}^2 \\ b &= -2(\Delta_{\mathrm{y}}^2 T_{\mathrm{x}} + \Delta_{\mathrm{x}}^2 T_{\mathrm{y}}) \\ c &= \Delta_{\mathrm{y}}^2 T_{\mathrm{x}}^2 + \Delta_{\mathrm{x}}^2 T_{\mathrm{y}}^2 - \frac{\Delta_{\mathrm{x}}^2 \Delta_{\mathrm{y}}^2}{F_{\mathrm{ij}}^2} \end{aligned} \tag{9}$$

where, in order to simplify the notation, we assume that the grid is composed of unit square cells, that is, $\Delta_{\mathrm{x}} = \Delta_{\mathrm{y}} = 1$.

The full procedure to compute the solution of FMM is detailed in Algorithm 1. The algorithm classifies the points of the map into three sets: frozen, open and unvisited. Frozen points are those for which the arrival time cannot change anymore. Unvisited points are those that have not been processed yet. Finally, open points are those which can be considered as an interface between frozen and unvisited regions of the map, belonging to the propagating wave front.

In the first step of the algorithm, the initialization, all the cells in the map are initialized with an infinite value (or the maximum value in the computing architecture) and set as unvisited, except for the starting point (the goal point in a path planning problem) which is set with an arrival time of 0 and considered as the first open point.

At each iteration, the open point with the smallest value of $T(x)$ is set as frozen. Then, the arrival time of its von-Neumann neighbors is analyzed (if they are not labeled as frozen) by solving Equation (8). The value of a cell is updated if the computed arrival time is smaller than the actual one (*UPDATE* in Algorithm 1). This procedure continues until all points are set as frozen or the starting point of a path planning problem is reached.

Figure 2 shows the first steps of the algorithm, in which different colors are used to identify the different level sets. In the center, the dark blue point is the source of the wave. The gray points near the corners represent open points which will be solved in the next iterations, Finally, the white circles are unvisited areas. The computed arrival time function starts at the minimum value ($T = 0$) and grows toward larger values of T, forming a level-set solution with a unique global minimum. If the solution is shown using the time of arrival as the third axis, a funnel potential is formed, as it is appreciated in the right image of **Figure 2**.

Algorithm 1: Fast Marching Method

```
1:  procedure FMM(X, x_0)
Require: A grid map X of size m × n, source point x_0.
    Initialization.
2:      for all x ∈ X do
3:          T(x) ← ∞;
4:      end for
5:      T(x_0) ← 0;
6:      frozen ← x_0;
7:      open ← N(x_0);                          ▷ Neighbors of x_0.
8:      open ← X\(frozen ∪ open);
    Iteration.
9:      while frozen ≠ X do
10:         x_1 ← arg min_{x∈open} d(x);
11:         for all x_i = N(x_1) ∈ T∩ ∉ frozen do
12:             UPDATE(x_i);
13:             open ← open ∪ {x_i};
14:         end for
15:         open ← open\{x_1};                  ▷ Updating sets.
16:         frozen ← frozen ∪ {x_1};
17:     end while
18: end procedure
```

Finally, since the time of arrival function has a funnel-like shape, a vehicle's path toward its goal point can be extracted using the gradient descent method. **Figure 3A** shows an example of a path computed with FMM. Note that, although the path is optimal in time, it traverses the environment too close to the obstacles and, besides, forces the vehicle to perform abrupt turns. In **Figure 3B**, the resulting expansion of the wave based on the FMM can be appreciated. The different colors in the image indicate different arrival time sets, being the dark blue the smallest values while the red area corresponds to larger arrival time points. Note that, while the computed path is the shortest is distance and time of arrival, it is not a feasible path since the autonomous ship would need to travel too close to the coast, with a great danger of collision or run aground.

2.2. The Fast Marching Square Method

The Fast Marching Square Method (FM^2) was introduced by Garrido et al. (2008) and consists on applying the basic FMM twice. Using this method, paths with an adequate smoothness and sufficient safety distances to the obstacles can be computed. The following procedure describes how the FM^2 computes paths:

1. The environment is modeled in the same way as when using the FMM, a binary grid map (see **Figure 3**). The cells belonging to obstacles are labeled in black (a 0 value) and

FIGURE 2 | (Left) Wave propagation using the FMM. Different colors represent different arrival times. In gray, the points of the next iteration. In white, unvisited points. (Right) The final result represented using the time as a third axis.

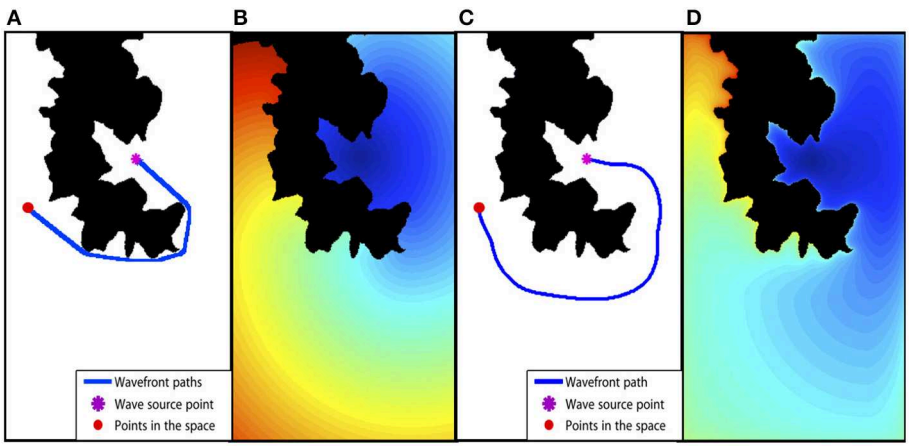


FIGURE 3 | Fast Marching Method based path planning example. **(A)** The binary map used in the path planning and the path computed with FMM from start to end point. **(B)** The time of arrival map computed with FMM. **(C)** The resulting path using the FM^2 method. **(D)** The time of arrival map obtained with FM^2.

the cells corresponding to free space are labeled in white (a 1 value).

2. The first time the FMM is applied over the binary map, each cell labeled as an obstacle is used as wave source, expanding several waves at the same time. The resulting value of each cell in the map indicates the time the wave needs to reach the closest obstacle, therefore, it is proportional to the distance from obstacles since the wave moves at a constant speed in the whole map. Reversing the meaning of these values, they can be interpreted as the speed of the vehicle (and the speed of the wave expansion). This way, the resulting map is understood as the maximum admissible speed at each point of the environment, so that if the autonomous ship is near to obstacles, the admissible speed is lower than when is away from the obstacles. Finally, the speed values are rescaled to fix a maximum cell value of 1.
3. Then, the FMM is applied again over the environment. This time, the robot's goal point is used as wave source (a unique wave source to ensure one global minimum). The wave is expanded over the map until the initial point of the vehicle is reached. At each cell in the environment, the speed at which the wave expands is taken from the map computed in the previous step. It is important to keep in mind that this speed is lower the closer the vehicle (wave) is to obstacles. **Figure 3D** shows the time of arrival map resulting of this process.
4. Finally, gradient descent is applied over the time of arrival map from the starting point of the ship, and moving toward its goal point (the global minimum of the resulting map), obtaining the optimal path in terms of time of arrival, smoothness and safety, as shown in **Figure 3C**.

It is important to note that, when using this method in a real autonomous vehicle, the user must be aware of two critical aspects. First, the resolution used to model the environment where the robot moves. Since FMM is a grid based method, the higher resolution used, the better model of the environment and movement of the vehicle, at the cost of computational time, as shown in Gómez et al. (2019). Second, the user should consider the cells of value equal to 1 in the speed map as the maximum speed the vehicle is able to use (or the user wants to consider).

Next, some interesting modifications of the speed map, which allow to achieve different behaviors of the wave expansion (and therefore the computed paths) are going to be explained.

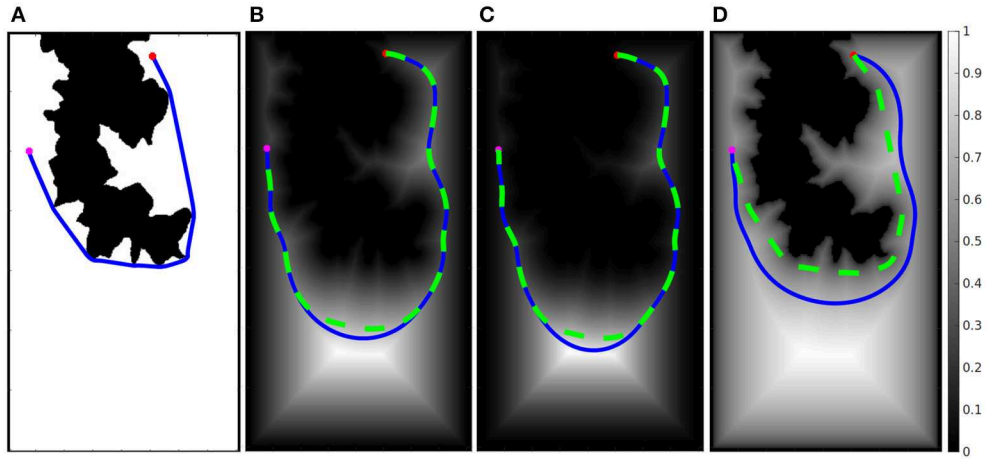


FIGURE 4 | Different speed maps and paths obtained by modifying α- and β-values. In **(A)**, the original map is shown with the path computed using the basic FMM method. In **(B–D)**, the velocity maps computed with $\beta = 1$ for all images, and $\alpha = 1$, $\alpha = 1.2$, $\alpha = 0.4$, respectively.

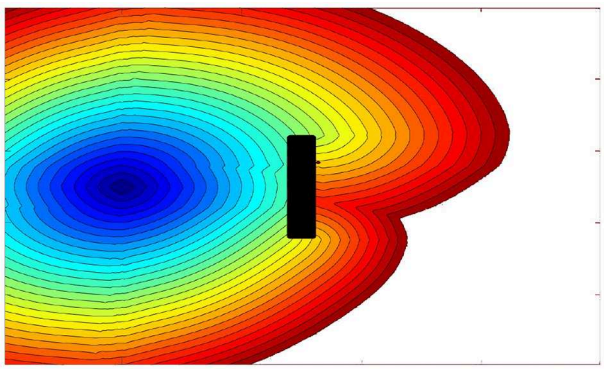


FIGURE 5 | Fast marching expansion wave with a rectangular obstacle in the middle. The upper part of the environment is subjected to a unitary vector field pointing to the right. In the lower part, the field points to the left.

2.2.1. The Flexibility of the Speed Map in *FM*^{2}

Although the paths generated by the FM^2 are good in terms of safety and smoothness, those paths can often be improved in terms of the traversed distance. For this reason, an adjustment parameter, α, that modifies the speed map to improve the planned path is proposed.

To perform the adjustment, each cell of the speed map, F_{ij}, is raised to the power indicated by this parameter as in:

$$newF_{ij} = F_{ij}^{\alpha} \tag{10}$$

When the value of α is lower than 1, the values in the speed map increase causing a lightening of the cells, which allows the wave expansion to use larger speeds. This causes the path to traverse the map closer to the obstacles. On the contrary, if the value is larger than 1, the cells are darkened causing paths stay further away from the obstacles.

Besides, it is commonly interesting to saturate the values in the speed map. For this reason, a value β is defined in the range of 0 and 1. The saturation is performed as follows: every cell in the speed map, F_{ij}, with a greater value than β is set to one. Since the speed map is a distance function, this means that the wave moves at the maximum speed in all the cells in the map whose distance to the closest obstacle is greater than β. Therefore, the value of this parameter depends on the deceleration capabilities of the vehicles in use.

Figure 4 illustrates the effect of modifying α and β values. In **Figure 4A**, the original map is shown with the path computed using the basic FMM method, the start and end points are marked with a red and purple point, respectively. **Figures 4B–D** show the velocity map computed with $\beta = 1$ for all images, and $\alpha = 1$, $\alpha = 1.2$, $\alpha = 0.4$, respectively. The resulting path is shown as a blue line. It is possible to appreciate that a value of α larger than 1 makes the velocity map to have greater values (darker in the image) which provoke the path to move farther from obstacles. On the other hand, when α is lower than 1, velocity values increase, allowing higher velocities around obstacles. Besides, for all cases, a second path is drawn using a dashed green line. This is the result of applying values $\beta = 0.7$, $\beta = 0.8$, $\beta = 0.5$, respectively. In all cases, the saturation value allows the path to move closer to obstacles, thus, reducing the path length at the cost of increasing the risk.

3. FAST MARCHING METHOD SUBJECTED TO A VECTOR FIELD (FMVF)

The methodologies explained in the previous sections share a common key characteristic, in all cases the expansion of the wave deals with scalar speed values. However, there are situations in which a vector speed function may better reflect the environmental conditions in the path planning process. For example, in Garrido et al. (2016), a vector field is used to model outdoors characteristics interesting in mobile robotics, such as slopes or landslides.

In order to represent the movement of a ship in the water it is necessary to, not only take into account its direction, but also

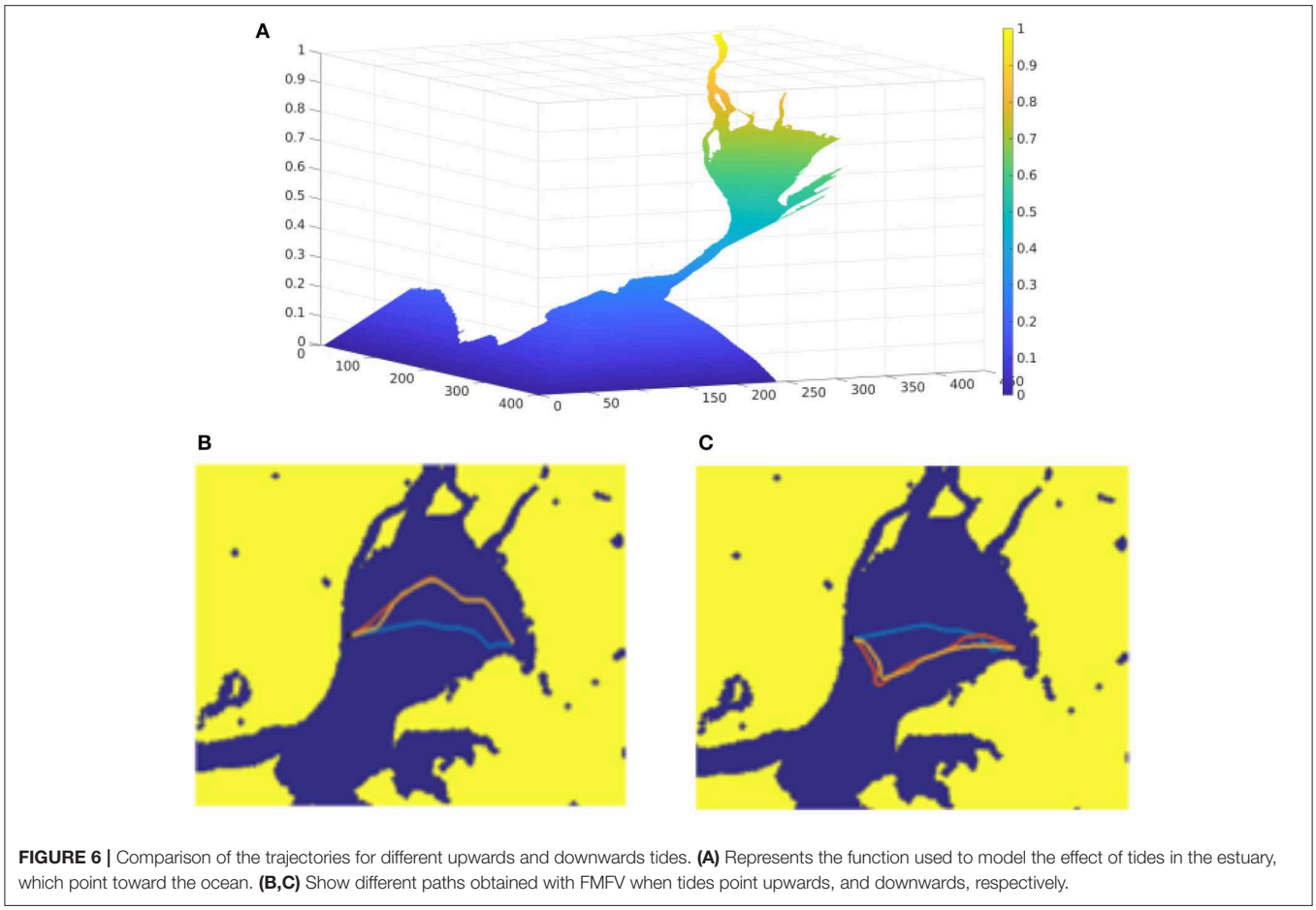


FIGURE 6 | Comparison of the trajectories for different upwards and downwards tides. **(A)** Represents the function used to model the effect of tides in the estuary, which point toward the ocean. **(B,C)** Show different paths obtained with FMFV when tides point upwards, and downwards, respectively.

FIGURE 7 | FM^2 trajectory followed by the model of a USV.

the effect of several vector variables such as wind flow or water currents. Mathematically, this can be done by computing a new cost function as in:

$$F_{ij} = F_{scal\_ij} + F_{vect\_ij} \tag{11}$$

where F_{scal_ij} represents the influence of the scalar cost map and F_{vect_ij} represents the external vector fields. F_{vect_ij} is computed as the sum of external vector fields that affect the process. In the case of a ship, wind, tides and marine currents.

In **Figure 5**, the effect of an external vector field on a wave propagation calculated by the Fast Marching Method subjected to a Vector Field (FMVF) is shown. Note that there is a rectangular obstacle in the middle shown, in black color, where the wave collapses. It is easy to appreciate how the wave propagates faster in the area where the vector field points in the same direction as the expansion of the wave (upper part of the image).

It is important to note that the authors in Petres et al. (2005) and Petres et al. (2007) treated this subject previously. In these works, a normalization of the magnitude of the external vector field is performed without taking into account the magnitude of the scalar cost function. This makes a vector field with an intensity of 1 to have the same effect on the final path than one with an intensity of 10, minimizing the real influence of the external field. However, in this work, the function that is normalized is the total cost function:

$$\tilde{f} = f_{dif} + f_{vect} \tag{12}$$

where f_{dif} is the cost function due to the distance to the obstacles in the environment converted into a vector field by:

$$f_{dif} = 1 - F_{ij} \tag{13}$$

This way, the influence of the vector field over the velocity of the vehicle depends on their magnitude as well as on the angle

between them, i.e., it depends on scalar product, and therefore the f_{vect} can be defined as:

$$f_{vect}(i,j) = 1 - \left\langle \nabla T_{i,j} \cdot \vec{F}_{i,j} \right\rangle \quad (14)$$

Physically, this is equivalent to say that a force favors the ship when both external vector field and vehicle are pointing to the same direction.

It is very important to remind that the new cost function defined in Equation (12) must always be positive, because in the methods based on FMM the wave-front cannot move backwards. More details on the algorithm can be consulted in Petres et al. (2005, 2007).

The next set of tests have been performed over a map of the Tagus River estuary, in the so-called Mar da Palha, in the city of Lisbon, **Figure 6**. In image A, the resultant wave expansion as a function of arrival time can be seen. Colors vary from dark blue for smaller to yellow for larger arrival time. This function is used to model the effect of tides in the estuary, which point toward the ocean (toward the dark blue area), inverting the sign makes the tides point in the opposite direction. In images B and C, different paths obtained with FMFV are shown. In **Figure 6B** the tides point upwards, while in **Figure 6C** tides point downwards, tides in both cases are identical in magnitude but in opposite direction. An example computed for a case in which the current is very close to zero, that is, the surface of the water is almost stationary, is shown in blue, which is used to analyze the influence of the introduction of a vector field of external forces. The magnitude of the currents is increased by a 5% from the yellow to the red test. It is clear that when the tide pushes either upwards or downwards, the calculated trajectories move away in comparison to the base trajectory (the blue one), since the vehicle undergoes a force that tends to take it away from the base path.

4. PATH FOLLOWING AND OBSTACLE AVOIDANCE USING FAST MARCHING BASED METHODS

In order to prove the smoothness of the paths computed with *FMM* based methods, a model of a real ship has been used to track them using a pure-pursuit method. The model uses a real-time control method for unmanned surface vehicles (USVs) based on Chaos et al. (2009).

Figure 7 shows the trajectory of a ship in the Atazar reservoir. Once the trajectory is computed with FM^2, the path is followed by the model using the pure-pursuit method. The control loop uses the orientation error to compute the rudder angle that best follows the path, then, pure-pursuit is used to calculate the velocity of the ship taking into account the speed function of FM^2. As shown in **Figure 7**, the calculated trajectory drawn in red, coincides with the poses of the ship reached using the simulated model.

In addition, it is commonly interesting for robots to be able to avoid obstacles while tracking their paths. Let us suppose that a mobile object is detected by the sensors in a ship (e.g., a LIDAR sensor) covering the path computed with FM^2. Also, let us define a region of influence (*roi*) around the ship covering the path. This area indicates the space in which any obstacle can cause a collision. Using this information, the method uses a cyclic execution described in Algorithm 2. First, a path from the start to the end point is obtained using FM^2. Then, pure pursuit is used to follow the path toward the next intermediate goal point. Next, if the end point is not reached, the *roi* is checked looking for mobile obstacles. When no mobile obstacle is detected, the path following continues with the original plan. However, when a mobile object is detected in this area, the previously computed path is no longer valid. In order to modify it, the velocity map is updated including the mobile object as a new obstacle, following the method explained in Garrido et al. (2013). The base of this update is to include a mobile obstacle location as an area with zero velocity (black in the velocity definition) which forces the wave expansion to avoid it. Since the velocity map is updated, a new path is computed (Second Potential). Therefore, the resulting new path avoids the mobile obstacle considering it as a static one during one control cycle.

Algorithm 2: Path Following and Obstacle Avoidance

```
1:  procedure FOLLOW_PATH(X, x_g, x_a, x_obs, roi)
Require: A grid binary map X of size m × n, goal point x_g, robot
    actual position x_a, location of obstacles at every iterations
    x_obs, region of interest around the ship roi.
    First Potential.
2:      vel = obtain_velocity_function(X)
    Second Potential.
3:      T = compute_FM^2(x_g, vel)
4:      path = compute_path(x_0, T)
    Path Following.
5:      rudder_angle = compute_angle(x_0, path)
6:      x_a = pure_pursuit(x_a, rudder_angle, vel)
7:      while x_a ≠ x_g do
8:          if x_obs_a+1 ⊂ roi ∧ x_obs_a+1 ≠ x_obs_a then
9:              vel = update_velocity_function(vel, x_obs_a+1)
10:             goto Second Potential
11:         else
12:             goto Path Following
13:
```

Figure 8 shows a sequence (top to bottom) of how a ship avoids another ship that acts as a mobile obstacle interfering its trajectory. The different columns show the process using different maps. In column A, a satellite image of the Atazar reservoir is used to draw the ships and the path at each moment. In column B, the inclusion of the obstacle in the speed map is shown. The point where the obstacle is detected is modeled as an obstacle (dark blue in the example) and the allowed speed around this obstacle increases slowly, as happens around every static obstacle in the map. In column C, on the right, the ship which follows the computed path and its area of influence (a green circle) are shown.

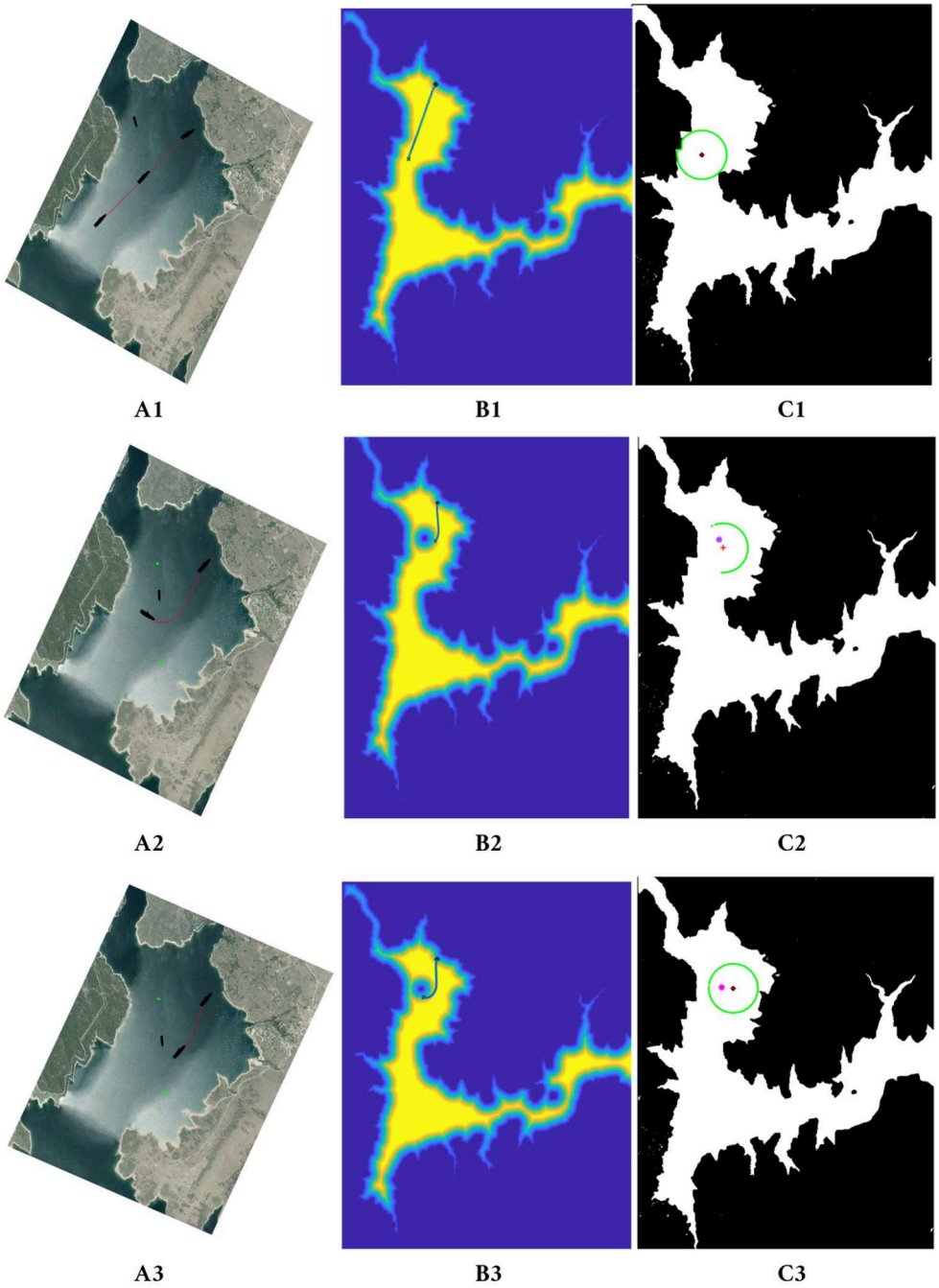


FIGURE 8 | FM^2 trajectory and path followed by the model of the ship avoiding the smaller ship moving from the left-upper corner. In **(A1–A3)** the sequence is shown on a satellite image of the reservoir. In **(B1–B3)** the mobile obstacle is included in the speed map as a circle. In **(C1–C3)** the ship following the path, its area of influence and the mobile obstacles are shown.

In the first row of **Figure 8**, the original computed path taking into account only static obstacles is shown, together with the area of influence of the ship at the starting position. In the second row, the mobile obstacle is detected on the left side of the ship and, therefore, included as a new obstacle. Because of this change, the updated path avoids this area turning to the right. Finally, in the third row, although the obstacle is still in the area of influence, the ship can follow its path toward the goal safely.

5. ROBOT FORMATIONS

The algorithm described next is an extension of previous works. Firstly, in Garrido et al. (2013), the use of FM^2 to control a

robot formation in 2D environments was presented. Then, its usage with unmanned aerial vehicles (UAVs) was treated in Alvarez et al. (2014). In this section, its adaptation to marine-like environments will be explained.

The algorithm for controlling the robot formation is based on a leader-followers scheme. The leader can be a robot or even a virtual leader. Using the leader as a reference, the poses for the follower robots are defined by geometric equations to form the shape of the formation. Therefore, the goal poses of each follower along the path are a function of the leader's pose.

In the proposed solution, the path of the leader is computed without taking into account the other robots in the formation. This may cause the followers to move too close to obstacles or even collide with them. In order to avoid these situations, a shape deformation scheme based on the two-level artificial potential of FM^2 can be used to calculate goal references to the followers during leader's navigation, as in reactive following. The main idea is to integrate an attracting potential toward the references of the formation (using the arrival time function) and a repelling potential from obstacles and other robots (the velocity/distances map).

Figure 9 shows an example of the use of the algorithm on a triangle-shaped robot formation. A 2D shape is used because it is easier to understand the behavior of the followers to avoid colliding with obstacles and among themselves. In **Figure 9A**, the main components of the robot formation are defined. In **Figure 9B**, the geometric definition of a triangle-shaped formation is presented, note that the tangential and perpendicular vectors of the leader's path are used as a reference. In **Figure 9C**, the goals of the followers adapt to the path of the leader's orientation. In **Figure 9D**, the use of the repelling potential to change the follower's partial goal and avoid obstacles in the environment is shown.

Algorithm 3 explains the integration of the control of the shape of the formation while covering the path. In the initialization (lines 2 to 7), the path for the leader is computed using FM^2. Then, the formation covers the path in a control loop in which: first, the leader tracks its path (lines 9 and 10), then, new goal poses for all the followers are computed based on formation geometry (line 12) and closeness to obstacles (line 13). Finally, paths for the followers are calculated and tracked (lines 14 to 17). The control loop ends when the leader reaches its goal.

Figure 10 shows the use of this method in marine-like environments. The formation uses a pyramid shape with a squared base, the followers are located in the corners of the square. The leader and the followers are submarines. The numbers in the axis are related to the voxelization of the environment. In the default shape, the formation is oriented so that two submarines are located in the same vertical line (up and down) and the other two are located in the same horizontal line (right and left). The deformation function used allows each corner of the shape to shrink the base toward its center proportionally to the closeness to obstacles (as indicated in the velocity map). The maximum allowed deformation is a 70% of the total distance, to avoid collisions within the formation. The part of the path the leader has already covered is shown in red, while the part that is yet to be covered is shown in blue. The geometry of the formation is shown in green. The past poses of the follower robots are shown as small dots.

Algorithm 3: Robot Formation Control based on FMM

1: **procedure** ROBOT_FORMATION_CONTROL(X, x_g, x_{la}, x_{obs})
Require: A grid binary map X of size $m \times n$, goal point x_{lg} for the leader of the formation, leader actual position x_{la}.
First Potential.
2: $vel_l = obtain\_velocity\_function(X)$
Second Potential.
3: $T_l = compute\_FM^2(x_g, vel_l)$
4: $path_l = compute\_path(x_{la}, T_l)$
5: **for all** k followers in formation **do**
6: $x_{pg\_k} = formation\_geometry(x_{la}, vel_k)$
7: **end for**
8: **while** $x_{la} \neq x_{lg}$ **do**
9: $rudder\_angle\_l = compute\_angle(x_{la}, path_l)$
10: $x_{la} = pure\_pursuit(x_{la}, rudder\_angle, vel)$
11: **for all** k followers in formation **do**
12: $x_{pg\_k} = formation\_geometry(x_{la}, vel_k)$
13: $x_{pg\_k} = update\_partial\_goal(vel_k, x_{pg\_k})$
14: $T_k = compute\_FM^2(x_{pg\_k}, vel_k)$
15: $path_k = compute\_path(x_{ka}, T_k)$
16: $rudder\_angle\_k = compute\_angle(x_{ka}, path_k)$
17: $x_{ka} = pure\_pursuit(x_{ka}, rudder\_angle_k, vel_k)$
18: **end for**
19: **end while**

The path of the leader traverses the environment over the valley formed by two peaks. **Figure 10A** shows the formation in the firsts steps of the movement. Note that the follower moving close to the bottom of the sea shrinks its position correcting its height. In **Figures 10B,C**, when the formation approaches and traverses the area around the peaks, all the followers except the upper one need to shrink toward the center. These deformations are provoked because the velocity map in the areas the followers traverse have velocity values close to zero, indicating that an obstacle is near. Therefore, the square based in shrunk to increase the security of the path. In **Figure 10D**, the followers are farther from obstacles and therefore enlarge the base of the pyramid.

Figure 11 shows the distance of the leader and the followers to the closest obstacle in the environment at every step of the algorithm. The distance is measured in voxels, so the real distance depends on the discretization used. Note how the distances are smaller in the central part of the path, in which the robots move between the peaks. Besides, the average deformation of the followers (also measured in voxels) is shown as a dashed line. Note that the deformation is larger when the distance to the obstacles is smaller.

6. CONCLUSIONS

In this paper, the use of the Fast Marching Method for marine-like environments is presented. Based on FMM, different

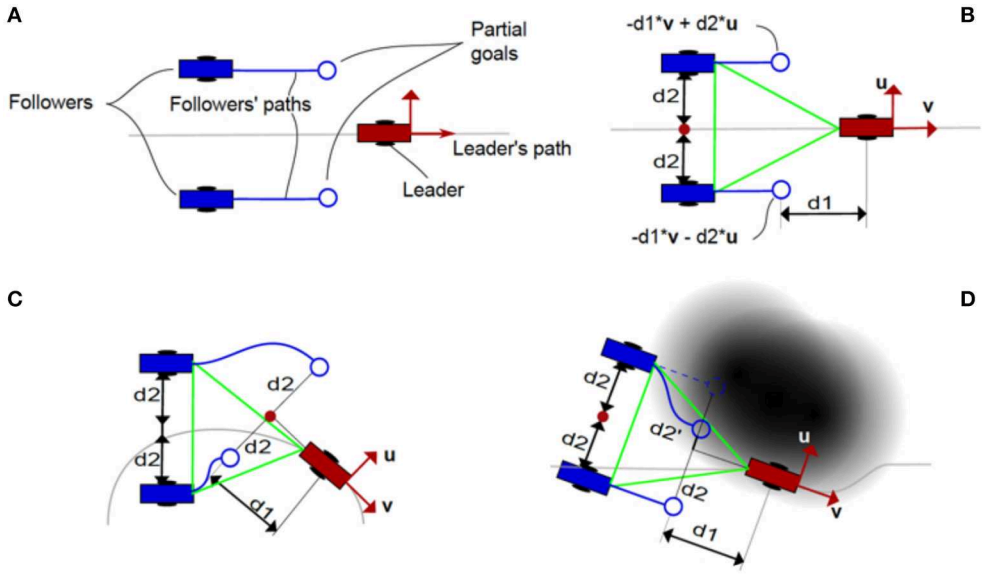


FIGURE 9 | (A) Main components of the robot formation algorithm. **(B)** Reference geometric definition of a simple, triangle-shaped robot formation, note that the definition is based on vectors *u* and *v* (tangential and perpendicular to the path, respectively). **(C)** Behavior of the partial goals depending on the leader's pose. **(D)** Behavior of the partial goals depending on the obstacles in the environment.

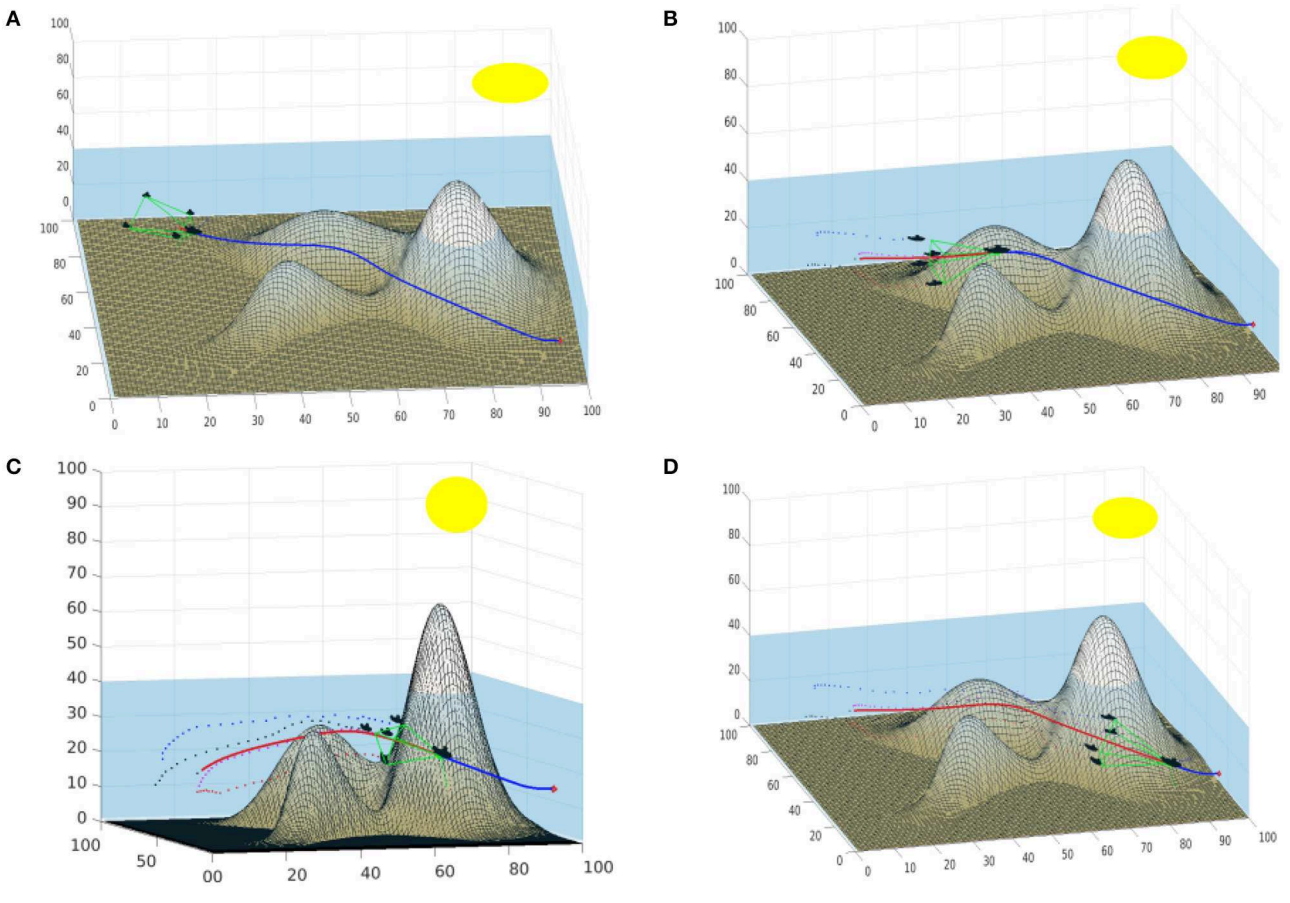


FIGURE 10 | Example of a formation of submarines with a pyramid shape. **(A)** Shows the formation in the first steps of the movement. **(B,C)** Show the iterations when the formation traverses the area around the peaks. **(D)** Shows the formation approaching the goal point.

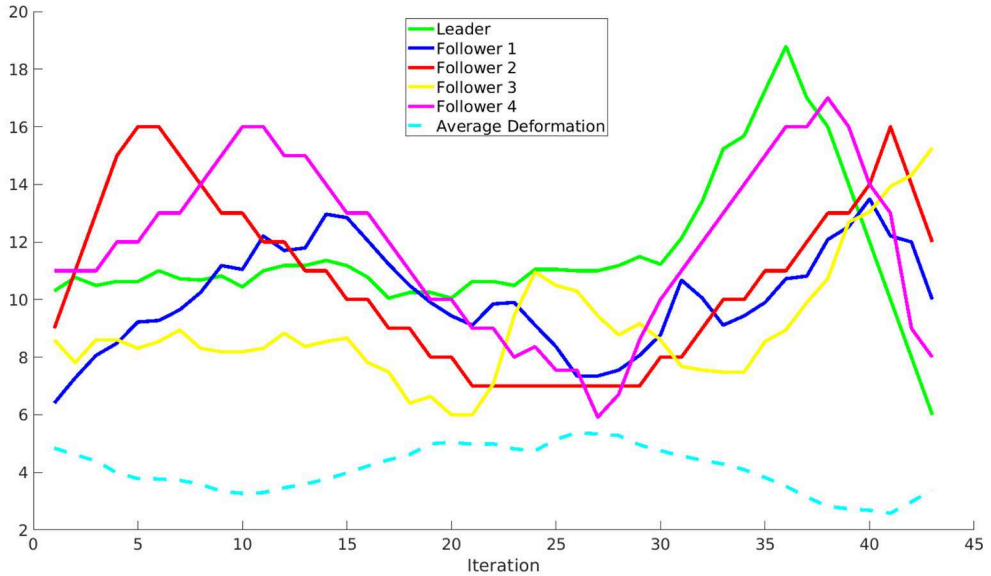


FIGURE 11 | Distance of the leader and the followers to the closest obstacle in the environment at every step of the algorithm. Besides, the average deformation of the followers is shown as a dashed line.

versions of the wave expansion and path planning solutions are introduced, explaining the specific characteristics of each method and solutions, which may help a user to decide which FMM based method fits a particular application.

Besides, the usage of FMM based methods on real-time path following, obstacle avoidance and formation control are presented. On every section, simulated paths over digital environments are shown in order to appreciate the differences introduced by the proposed changes on the basic FMM. It is important to note that any of the explained FMM-like methods may be used to implement these applications. However, formation control has not yet been tested with the FMFV method, which is one the main future works.

Besides, future work will also focus on improving the models of the mobiles obstacles by using directional models and on extracting numerical results the safety provided by FMM-like path planning algorithms and the usage of robot formations in marine-like environments. Also, the implementation of these algorithms in a real autonomous marine vehicle is an important future work.

DATA AVAILABILITY STATEMENT

The datasets generated for this study are available on request to the corresponding author.

AUTHOR CONTRIBUTIONS

SG developed the FMVF method. DA performed the testing, wrote the algorithm explanations, and developed the Simulink model. LM participated on the discussion and the development of the techniques around FM2.

FUNDING

The research leading to these results has received funding from HEROITEA-Sistema Inteligente Heterogéneo Multirobot para la Asistencia de Personas Mayores-RTI2018-095599-B-C21 and from RoboCity2030-DIH-CM, Madrid Robotics Digital Innovation Hub, S2018/NMT-4331), funded by Programas de Actividades I+D en la Comunidad de Madrid and cofunded by Structural Funds of the EU.

REFERENCES

Agarwal, A., and Lermusiaux, P. F. (2011). Statistical field estimation for complex coastal regions and archipelagos. *Ocean Model.* 40, 164–189. doi: 10.1016/j.ocemod.2011.08.001

Alvarez, D., Gomez, J., Garrido, S., and Moreno, L. (2014). "3d robot formations planning with fast marching square," in *IEEE International Conference on Autonomous Robot Systems and Competitions* (Espinho).

Álvarez, D., Gómez, J. V., Garrido, S., and Moreno, L. (2015). 3d robot formations path planning with fast marching square. *J. Intell. Robot. Syst.* 80, 507–523. doi: 10.1007/s10846-015-0187-1

Bellingham, J. G., and Willcox, J. S. (1996). "Optimizing auv oceanographic surveys," in *Proceedings of Symposium on Autonomous Underwater Vehicle Technology* (Monterey, CA), 391–398.

Chaos, D., Moreno, D., Aranda, J., and de la Cruz, J. M. (2009). A real-time control for path following of an USV. *IFAC Proc. Vol.* 42, 261–266. doi: 10.3182/20090916-3-BR-3001.0011

Garrido, S., Álvarez, D., and Moreno, L. (2016). "Path planning for mars rovers using the fast marching method," in *Robot 2015: Second Iberian Robotics Conference*, eds L. P. Reis, A. P. Moreira, P. U. Lima, L. Montano, and V. Muñoz-Martinez (Cham: Springer International Publishing), 93–105.

Garrido, S., Moreno, L., and Blanco, D. (2008). Exploration of a cluttered environment using voronoi transform and fast marching method. *Robot. Auton. Syst.* 56, 1069–1081. doi: 10.1016/j.robot.2008.02.003

Garrido, S., Moreno, L., Gomez, J., and Lima, P. (2013). General path planning methodology for leader-followers based robot formations. *Int. J. Adv. Robot. Syst.* 10, 1–10. doi: 10.5772/53999

Garrido, S., Moreno, L., Martín, F., and Álvarez, D. (2017). Fast marching subjected to a vector field–path planning method for mars rovers. *Exp. Syst. Appl.* 78, 334–346. doi: 10.1016/j.eswa.2017.02.019

Gomez, J. V., Alvarez, D., Garrido, S., and Moreno, L. (2017). Fast marching-based globally stable motion learning. *Soft Comput.* 21, 2785–2798. doi: 10.1007/s00500-015-1981-1

Gómez, J. V., Álvarez, D., Garrido, S., and Moreno, L. (2019). Fast methods for eikonal equations: an experimental survey. *IEEE Access* 7, 39005–39029. doi: 10.1109/ACCESS.2019.2906782

Hert, S., Tiwari, S., and Lumelsky, V. (1996). A terrain-covering algorithm for an AUV. *Auton. Robots* 3, 91–119. doi: 10.1007/BF00141150

Osher, S., and Sethian, J. A. (1988). Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.* 79, 12–49. doi: 10.1016/0021-9991(88)90002-2

Petres, C., Pailhas, Y., Evans, J., Petillot, Y., and Lane, D. (2005). "Underwater path planning using fast marching algorithms," in *Oceans European Conference*, Vol. 2 (Brest), 814–819. doi: 10.1109/OCEANSE.2005.1513161

Petres, C., Pailhas, Y., Patron, P., Petillot, Y., Evans, J., and Lane, D. (2007). Path planning for autonomous underwater vehicles. *IEEE Trans. Robot.* 23, 331–341. doi: 10.1109/TRO.2007.895057

Sethian, J. A. (1996). Theory, algorithms, and applications of level set methods for propagating interfaces. *Acta Numer.* 5, 309–395. doi: 10.1017/S0962492900002671

Sethian, J. A. (1999). *Level Set Methods and Fast Marching Methods*. Cambridge: Cambridge University Press.

Song, R., Liu, W., Liu, Y., and Bucknall, R. (2015). "A two-layered fast marching path planning algorithm for an unmanned surface vehicle operating in a dynamic environment," in *OCEANS 2015* (Genova), 1–8.

Song, R., Liu, Y., and Bucknall, R. (2017). A multi-layered fast marching method for unmanned surface vehicle path planning in a time-variant maritime environment. *Ocean Eng.* 129, 301–317. doi: 10.1016/j.oceaneng.2016.11.009

Yatziv, L., Bartesaghi, A., and Sapiro, G. (2005). O(n) implementation of the fast marching algorithm. *J. Comput. Phys.* 212, 393–399. doi: 10.1016/j.jcp.2005.08.005

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Robotics and AI

ORIGINAL RESEARCH
published: 31 January 2020
doi: 10.3389/frobt.2020.00003

Underwater Robotics Competitions: The European Robotics League Emergency Robots Experience With FeelHippo AUV

Matteo Franchi [1,2]*, *Francesco Fanelli* [1,2], *Matteo Bianchi* [1,2], *Alessandro Ridolfi* [1,2] and *Benedetto Allotta* [1,2]

[1] Department of Industrial Engineering, University of Florence, Florence, Italy, [2] Interuniversity Center of Integrated Systems for the Marine Environment, Genoa, Italy

Underwater robots are nowadays employed for many different applications; during the last decades, a wide variety of robotic vehicles have been developed by both companies and research institutes, different in shape, size, navigation system, and payload. While the market needs to constitute the real benchmark for commercial vehicles, novel approaches developed during research projects represent the standard for academia and research bodies. An interesting opportunity for the performance comparison of autonomous vehicles lies in robotics competitions, which serve as an useful testbed for state-of-the-art underwater technologies and a chance for the constructive evaluation of strengths and weaknesses of the participating platforms. In this framework, over the last few years, the Department of Industrial Engineering of the University of Florence participated in multiple robotics competitions, employing different vehicles. In particular, in September 2017 the team from the University of Florence took part in the European Robotics League Emergency Robots competition held in Piombino (Italy) using FeelHippo AUV, a compact and lightweight Autonomous Underwater Vehicle (AUV). Despite its size, FeelHippo AUV possesses a complete navigation system, able to offer good navigation accuracy, and diverse payload acquisition and analysis capabilities. This paper reports the main field results obtained by the team during the competition, with the aim of showing how it is possible to achieve satisfying performance (in terms of both navigation precision and payload data acquisition and processing) even with small-size vehicles such as FeelHippo AUV.

Keywords: underwater robots, autonomous underwater vehicle, robotics competitions, autonomous navigation, acoustic mosaicing

OPEN ACCESS

Edited by:
Enrica Zereik,
Italian National Research Council (CNR), Italy

Reviewed by:
Antonio Vasilijevic,
University of Zagreb, Croatia
Filippo Campagnaro,
University of Padova, Italy

***Correspondence:**
Matteo Franchi
matteo.franchi@unifi.it

Specialty section:
This article was submitted to Robotic Control Systems, a section of the journal Frontiers in Robotics and AI

Received: *24 April 2019*
Accepted: *08 January 2020*
Published: *31 January 2020*

Citation:
Franchi M, Fanelli F, Bianchi M, Ridolfi A and Allotta B (2020) Underwater Robotics Competitions: The European Robotics League Emergency Robots Experience With FeelHippo AUV. Front. Robot. AI 7:3. doi: 10.3389/frobt.2020.00003

1. INTRODUCTION

Unmanned underwater vehicles, both teleoperated and autonomous, are nowadays employed for many applications, effectively helping human operators performing a wide variety of tasks (or even replacing them during their execution) (CADDY, Mišković et al., 2016). Underwater vehicles come in different shapes and sizes: from those with a length of several meters and a weight of hundreds of kilograms (e.g., Rigaud, 2007; Furlong et al., 2012; Kaiser et al., 2016) to the more

compact and lightweight (for instance Hiller et al., 2012; Crowell, 2013; McCarter et al., 2014). While bigger vehicles naturally allow the use of more complex instrumentation and possess the ability to store heavy payload, smaller vehicles are commonly associated with lower performance and limited payload carrying capabilities. Hence, one of the current challenges that designers of small vehicles need to face consists in the optimization of the available space on board.

In this framework, the Mechatronics and Dynamic Modeling Laboratory (MDM Lab) of the Department of Industrial Engineering of the University of Florence (UNIFI DIEF) has been active in the field of underwater robotics since 2011, participating in different robotics-related research projects and developing and building several AUVs since then. Furthermore, throughout the years, UNIFI DIEF took part in multiple student and non-student robotics competitions. A team from UNIFI DIEF (UNIFI Team) took part in the Student Autonomous Underwater Vehicles Challenge - Europe (SAUC-E) Ferri et al. (2015) competition in 2012, 2013, and 2016, while in 2015 the team participated in euRathlon (Ferri et al., 2016); finally, it took part in the European Robotics Leaugue (ERL) Emergency Robots competition in September 2017 (Ferri et al., 2017).

This paper reports the field experience of the UNIFI Team at ERL Emergency Robots 2017, held in Piombino (Italy), from the 15th to the 23rd of September. During the nine competition days, the robots of the participating teams competed in a set of tasks in the land, air, and sea domains. This paper focuses on the results obtained in the sea domain with FeelHippo AUV: in particular, it will be shown how such vehicle, despite its small size, possesses a complete navigation system capable of offering satisfying accuracy while autonomously navigating; at the same time, it will be demonstrated how the diverse payload the vehicle is equipped with can be exploited for different purposes. In other words, the mechatronics design has been conceived to be a suitable trade-off between portability and high performance.

Other AUVs used in student robotics competitions can be found for example in Fietz et al. (2014) and Carreras et al. (2018) (the winner of ERL Emergency Robots 2017). The remainder of the paper is organized as follows: section 2 and section 3 are dedicated to the description of FeelHippo AUV; while the former focuses on the mechanical design of the vehicle and on the onboard devices, the latter describes its software architecture, giving an overview of its navigation system and describing some of its payload analysis and processing capabilities. Section 4 reports the most significant results obtained during the competition, and section 5 concludes the paper.

2. FEELHIPPO AUV: DESCRIPTION

FeelHippo AUV has been designed and developed specifically for the participation in student robotics competitions; it has been used by a team of UNIFI DIEF during SAUC-E 2013, euRathlon 2015, and ERL Emergency Robots in 2017.

In addition to student competitions, FeelHippo AUV has been used for short navigation missions, mainly in shallow waters, from 2015 onward. Thanks to the sensors added for the competition, the level of performance achieved was satisfying; hence, it was decided to incorporate such devices within the standard equipment of the vehicle. From early to mid 2017, FeelHippo AUV underwent a major overhaul, in terms of both mechanical components (identifying those parts and subsystems that could be redesigned to increase overall functionality) and navigation sensors (permanent integration of new instrumentation required indeed a general revision of the electronics of the vehicle, in order to optimize the occupied volume). In particular, the old oil-filled thrusters were replaced in favor of thrusters manufactured by BlueRobotics and tailored to underwater applications. In addition to this, a new DVL by Nortek has been placed under the center of gravity of the vehicle. Formerly, it was positioned in the stern. As a consequence, the stability of the vehicle is increased. More information concerning the payload can be found in the following. In its current version (as of end 2017, **Figure 1**), FeelHippo AUV can be efficiently used as a small survey and inspection AUV, suitable for use in present and future research projects or, generally speaking, autonomous sea operations.

The main characteristics of the vehicle are reported in **Table 1**; the reduced dimensions and weight, together with the convenient handles visible in **Figure 1**, allow for easy transportation and deployment (no more than two people are required, and even deployment from shore is possible).

FIGURE 1 | FeelHippo AUV, 2017 version.

TABLE 1 | FeelHippo AUV physical data and performance.

FeelHippo AUV main characteristics	
Dimensions [mm]	∼600×640×500
Mass [kg]	35
Max longitudinal speed [m/s] (kn)	∼1 (2)
Max depth [m]	35
Autonomy [h]	2–3

The central body of FeelHippo AUV is composed of a Plexiglass® hull with an internal diameter of 200 and 5 mm thickness, which houses all the non-watertight hardware and electronics. Two metal flanges constitute the connection between the central body and the Plexiglass® domes at each end of the main hull, and two O-rings ensure a watertight connection between the former and the domes.

Four thrusters in vectored configuration (two on the stern and two on both lateral sides tilted of 45°), used to control translational motion and yaw (limited roll and pitch are guaranteed by hydrostatic stability), are connected with the central frame by 3-D printed custom-made plastic parts. Concerning the internal electronics, all the components are mounted on two parallel Plexiglass® planes, placed on linear guides which facilitate assembly and maintenance operations (allowing to easily extract internal components from within the central body of the vehicle). An Intel i-7 Mobile CPU is used for onboard processing, while the sensor set FeelHippo AUV is equipped with includes:

- U-blox 7P precision Global Positioning System (GPS);
- Xsens MTi-300 AHRS, composed of triaxial accelerometers, gyroscopes and magnetometers;
- Nortek DVL1000 Doppler Velocity Log (DVL), measuring linear velocity and also acting as Depth Sensor (DS). The device has been placed under the central body of the AUV; indeed, being such component quite heavy (∼2.7 kg in air), this choice increases stability in water;
- KVH DSP 1760 single-axis high precision Fiber Optic Gyroscope (FOG) for a precise measurement of the vehicle heading.

For what concerns communication, in addition to a WiFi access point and a radio modem, an EvoLogics S2CR 18/34 acoustic modem is used underwater; in addition, a custom-made antenna houses four rows of RGB LEDs, used for easy optical communication of the state of the vehicle (e.g., low battery, acquisition of the GPS fix, mission start) while the former is on surface. Regarding payload, the following devices are currently mounted on the vehicle:

- One Microsoft Lifecam Cinema forward-looking camera, which also allows teleoperated guide;
- One bottom-looking ELP 720p MINI IP camera;
- Two lateral ELP 1080p MINI IP cameras, used for stereo vision;
- One Teledyne BlueView M900 2D Forward-Looking SONAR (FLS).

A scheme of the connections (logical and physical) among the components of the vehicle is reported in **Figure 2**. Despite its reduced size, FeelHippo AUV is able to equip diverse payload, both optical and acoustical. Furthermore, thanks to its particular structure, additional small devices (such as, e.g., supplementary cameras or LED illuminators) can be added to the main body of the vehicle with ease. More information about FeelHippo AUV versions from 2013 to 2017 can be found in Fanelli (2019), whereas more recent versions are described in Franchi et al. (2019). A comparison (in terms of dimensions and weight) with other competitors is reported in **Table 2**.

TABLE 2 | FeelHippo AUV compared with other AUVs present on the market.

AUV model	Dimensions (mm), weight (kg)
FeelHippo AUV	600 × 640 × 500, 35
Remus 100 (Kongsberg)	1,700 × 190, 37
Sparus *II* (IQUA Robotics)	1,600 × 230, 52
LAUV (OceanScan-Marine Systems Technology)	150 × 2,300, 35

The competitors are torpedo-shape vehicle and the dimensions are diameter×length.

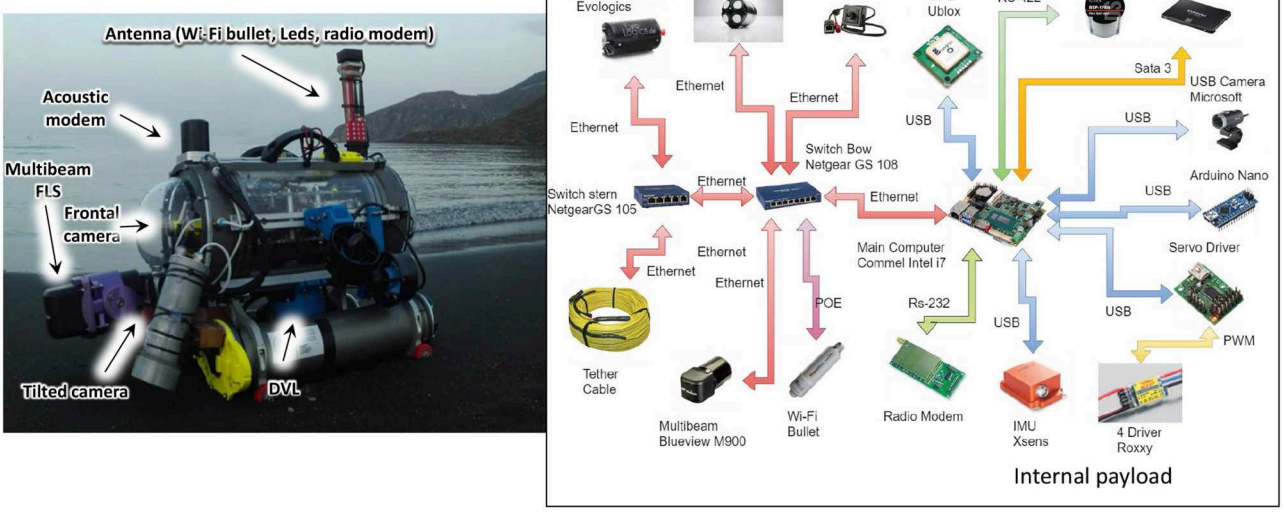


FIGURE 2 | FeelHippo AUV connections scheme.

With the aim of highlighting the compactness of FeelHippo AUV, its physical data are compared with other AUVs present on the market.

3. FEELHIPPO AUV: SOFTWARE ARCHITECTURE

The software architecture is modular with independent processes that share information through an adapted TCP/IP protocol called Transmission Control Protocol for Robot Operating System (TCPROS) (Amaran et al., 2015; ROS). In section 3.1 a quick overview of the Guidance, Navigation, and Control (GNC) system is depicted, whereas in section 3.2 how to manage acoustic payload is described.

3.1. FeelHippo AUV: Guidance, Navigation, and Control System

Thanks to the available navigation sensors on board, introduced in section 2, FeelHippo AUV is capable of successfully performing autonomous navigation missions for the full extent of its battery charge without the need to resurface: thanks to a careful mechanical design, the vehicle is able to house position, depth, inertial, magnetic field, and velocity sensors inside its main body, thus disposing of a complete navigation system used to compute the pose of the AUV in real-time. Additionally, thanks to the presence of an acoustic modem, the vehicle is able to receive acoustic position fixes sent by dedicated instrumentation (e.g., Long, Short, or Ultra-Short BaseLine systems), which can be integrated within its GNC system and exploited to correct the pose estimated on board while underwater (or in any GPS-denied scenario).

The navigation filter of FeelHippo AUV is the same as the one of the others AUVs of the MDM Lab, exploiting all the features developed at the University of Florence during past and present research projects; hence, this section only briefly reviews the core concepts.

The navigation system is used to determine an accurate estimate of the pose of the vehicle with respect to a local Earth-fixed reference frame whose axes point, respectively, North, East, and Down (NED frame). Resorting to the classic notation exploited to describe the motion of underwater vehicles (Fossen et al., 1994), such quantity is denoted with $\boldsymbol{\eta} = \left[\boldsymbol{\eta}_1\ \boldsymbol{\eta}_2\right]'$, where $\boldsymbol{\eta}_1$ indicates the position of the AUV, and $\boldsymbol{\eta}_2$ its orientation (exploiting a triplet of Euler angles; roll, pitch, and yaw are used in this context). Additionally, let us denote with $\boldsymbol{\nu} = [\boldsymbol{\nu}_1\ \boldsymbol{\nu}_2]'$ the velocity (linear and angular) of the vehicle with respect to a body-fixed reference frame, and with $\boldsymbol{\tau} \in \mathbb{R}^6$ the vector of forces and moments acting on the AUV.

A parallel structure has been chosen (refer to **Figure 3**): attitude is independently estimated using IMU, compass, and FOG data, and constitutes an input that is fed to the position estimation filter. In particular, the attitude estimation filter is based on the nonlinear observer proposed in Mahony et al. (2008), whose principle is to integrate angular rate changes measured by gyroscopes and correcting the obtained values exploiting accelerometers and magnetometers. The structure of the original filter proposed in Mahony et al. (2008) has then been suitably modified in order to better adapt it to the underwater field of application (Allotta et al., 2015; Costanzi et al., 2016); in particular, a real-time strategy to detect external magnetic disturbances (which would detrimentally affect the yaw estimate) has been developed in order to maintain the accuracy of the computed estimate in a wide variety of possible environmental conditions, promptly discarding corrupted compass reading, and relying on the high precision single-axis FOG.

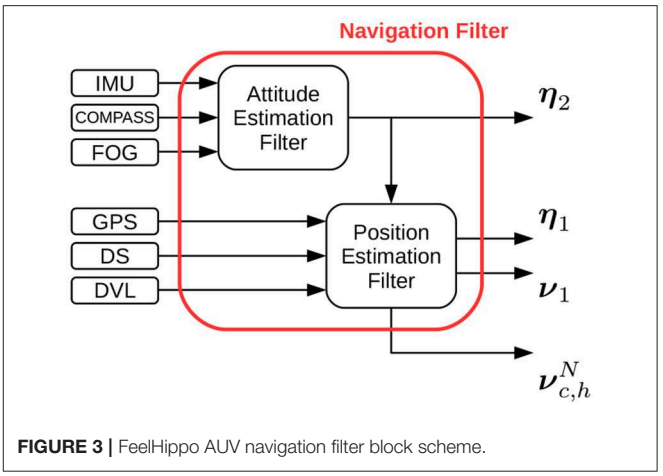


FIGURE 3 | FeelHippo AUV navigation filter block scheme.

For what concerns position estimation, in addition to being able to navigate in dead reckoning (which has proven to be satisfyingly reliable despite its straightforward philosophy if the adopted sensors are sufficiently accurate), the vehicle can resort to an Unscented Kalman Filter (UKF)-based estimator. Such filter makes use of a mixed kinematic/dynamic vehicle model (so as to capture more information about the evolution of the system with respect to a purely kinematic model, but at the same time offering a reduced burden on the processing unit of the vehicle with respect to a complete dynamic model), taking into account longitudinal dynamics only (the majority of torpedo-shape AUV motion takes place on the direction of forward motion, since it usually constitutes the direction of minimal resistance).

The reader can refer to Allotta et al. (2016), Caiti et al. (2018), and Costanzi et al. (2018) for more details.

3.2. Payload Acquisition and Processing

Object detection and mapping is a typical problem in the underwater domain. Research on this topic is crucial for both AUVs and Remotely Operated Vehicles (ROVs), permitting them to understand their surroundings. Unfortunately, different and a priori unknown scenarios, which affect the robot-environment interaction, need to be faced. Poor visibility conditions in murky and turbid waters can compromise the operations of optical devices. To overcome the above-mentioned issues, FeelHippo AUV presents, as stated in section 2, a FLS. In the first part of the section, an acoustic-based buoy detection algorithm with a reinforcement that exploits the known geometric dimensions of a static target is proposed.

3.2.1. FLS-Based Buoy Detection

The main concepts behind the algorithm are outlined:

- The acoustic video is acquired by one Teledyne BlueView M900 2D FLS and then it is real-time separated into a sequence of 8-bit grayscale images;
- Each frame is blurred with a Gaussian filter, leading to a smoother image.
- In order to detect high-reflection areas, which are likely to belong to a target object rather than to reverberation caused by the clutter, a direct binary threshold is applied to all the acoustic images. Let us define the source image as *src*, the destination image (namely the one after the binary threshold application) as *dst* and the threshold value $threshold \in [0, 255]$. Note that the interval limits depend on the depth of the image. As mentioned above, 8-bit grayscale images are considered;
- Each frame is modified by means of morphological dilations. Because of the environmental disturbances, some speckle areas, which do not belong to any buoy, can take place. Morphological operations make these areas to coalesce, so they can be easily ignored, avoiding false-positive detections. The situation is clearly visible in **Figure 4** where high-reflection areas are due to the bubbles in front of the vehicle.
- At this point, several white colored bounded sets are present. Geometric boolean requirements need to be met to distinguish buoy-like objects from the background. The main assumption behind the proposed method lies in the knowledge (even rough) of the shape and dimension of the target to detect. On the one hand, our technique exploits simple geometric conditions; on the other hand, targets that resemble elementary geometric shapes are meant to be identified (circles, ellipses, rectangles). Commonly, typical buoys fall inside the scope of applicability of the proposed algorithm, which appears as a good trade-off. Four geometric properties that lead to four boolean conditions are considered and it is worth highlighting that all the requirements need to be met. First of all, the area of all the bounded sets is checked. If it is between a minimum (a_{min}) and a maximum (A_{MAX}) value, the condition is verified. The goal is trivial: ignore too small or too big regions. Second of all, the circularity, which is defined below, is investigated. If it is between a minimum and a maximum value, the condition is confirmed. Its meaning lies in understanding how much the bounded sets resemble a circle. Obviously, ellipticity is taken into account when circularity is different from one.

$$Circularity = \frac{4\pi A}{P^2}, \tag{1}$$

where A is the area of the bounded set and P its perimeter. Afterwards, the convexity, defined as the ratio between the area of the set and the area of its convex hull (the smallest convex set that contains the original set), is checked. Another go/no go condition is applied.

$$Convexity = \frac{A}{A_{ch}}, \tag{2}$$

where A is the area of the bounded set and A_{ch} is the area of the convex hull. It is easy to understand that convexity $\in (0, 1]$.

Lastly, the inertia ratio, which is defined in Equation (3) is verified. The goal is trivial: detect whether the object is elongated along a particular direction. Note that the moments of inertia are calculated with respect to the center of mass of the set.

$$IR = \frac{Imax}{Imin}, \tag{3}$$

where $Imax$ and $Imin$ are respectively the maximum and the minimum moment of inertia (the inertia along the principal axes) and $IR \in \mathbb{R}^+$.

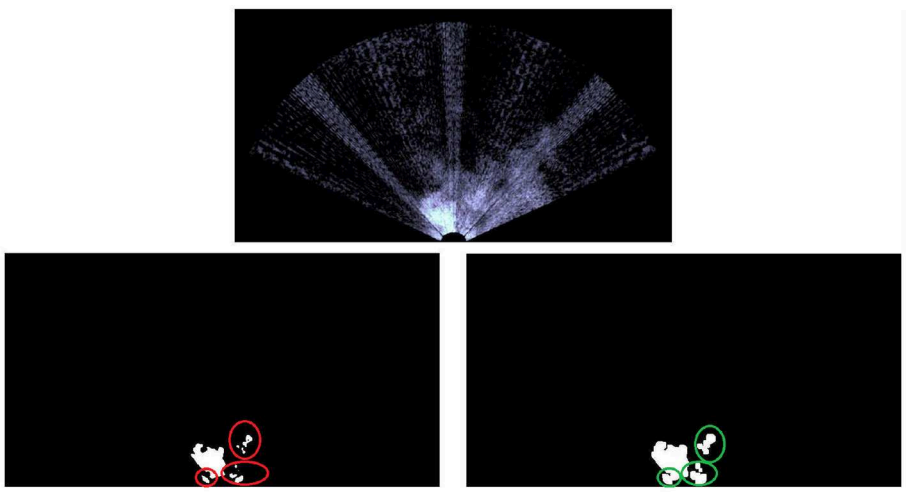


FIGURE 4 | The image acquired by the FLS on top (note the bubbles in front of the vehicle that create a strong acoustic echo, see the white area). The binary threshold down on the left, whereas the latter is morphological dilated on the right. The red circles are the speckle areas and the green ones the subsequent aggregation.

Unfortunately, as stated by Hurtós et al. (2015), FLS imagery are affected by low Signal-to-Noise Ratio (SNR), poor resolution and intensity modifications that depends upon viewpoint variations, so some false positive detections might arise anyway.

Assuming a static target (very often a buoy falls inside this class), a position-based clustering algorithm with the aim of removing false positives, can be exploited. If several detections are accumulated around a small region, then what is insonified by the FLS has high probability to be the buoy. In other words, the presence of spurious noise and mobile objects (e.g., fish) can be managed by the proposed technique, leading to a robust solution. The key idea exploits the solution proposed in Ester et al. (1996) where, basically, elements with many nearby neighbors are grouped together, whereas points that lie too far from their closest neighbors are classified as outliers.

To locate the exact position of the detected targets, starting from the known position of the vehicle, an imaging geometry model needs to be defined and the reader can refer to Franchi et al. (2018) for more information. In few words, exploiting the work of Johannsson et al. (2010), Ferreira et al. (2014), Hurtós et al. (2015), and Walter (2008), a simplified linear model, where the FLS can be treated as an orthographic camera, is adopted.

4. ERL EMERGENCY ROBOTS 2017 EXPERIMENTAL RESULTS

This section reports some of the results obtained during the robotics competition ERL Emergency Robots 2017, held in Piombino (Italy) in September 2017. In particular, the data shown here refer to multiple autonomous missions performed by FeelHippo AUV during the sea domain trials throughout the competition [refer to Ferri et al. (2017) for more details about the challenge]. Robots were asked to act in the following (recreated) catastrophic scenario: after an earthquake and a tsunami hit the shoreline area where a nuclear plant is located, evacuation procedures are issued; however, several people working at the plant are missing. Additionally, the premises have suffered damages of relevant intensity, with their lower sections flooded; furthermore, several pipes of the plant (both on land an underwater) are leaking radioactive material. Concerning the sea domain, the area of interest was constituted by a rectangular arena $\sim$50 $\times$ 50 m wide. Beyond a starting gate, composed of two submerged buoys, lied the area of interest where an underwater plastic pipe assembly represented the (flooded) lower section of the plant. Obviously, no substance was actually leaking; a set of five numbered underwater buoys was used to represent the leaking fluid plume (leading to a particular component of the pipe assembly, where a specific marker represented the breakage). In addition, several objects anchored on the seabed (e.g., tables and chairs) indicated a debris area where it was likely to find the body (i.e., a mannequin dressed in easy visible orange) of one of the missing workers. See **Figure 5** for a graphical representation of the arena and of the objects of interest (note that the picture is not to scale, and the positions of the depicted objects are not meant to represent actual shapes or dimensions).

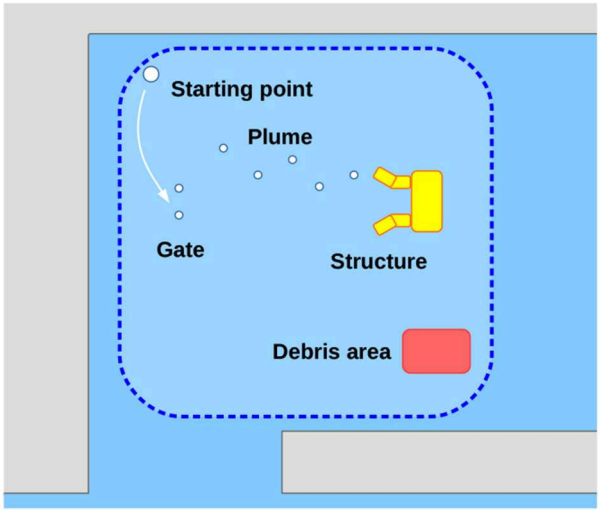


FIGURE 5 | ERL Emergency Robots 2017 sea domain arena.

Each participating team was allotted an exclusive time slot in the arena; from the starting point, the vehicle had to submerge, pass through the gate (without touching it, and providing optical or acoustical images of the gate itself), and it was then required to perform different tasks without resurfacing. Among the different tasks (but not limited to those mentioned here), each AUV was asked to inspect and map the area and the objects of interest (e.g., the plume, the gate, the underwater pipe assembly, the debris area) and to identify in real-time the mission targets, such as the leaking pipe and the missing worker. A specific score based on the degree of completeness and on the quality of the provided data (navigation and/or payload data, used to guarantee the veracity of team's claims on each submission) was assigned to each task. Hence, each AUV had to (a) precisely navigate through the arena, closely following the planned path in order to (b) efficiently make use of its own payload and payload processing algorithms, mapping the arena and identifying the objects of interest during navigation so as to score as much points as possible. In light of the above-mentioned considerations, this section is divided into two parts: at first, the focus will be given to the navigation performance of the vehicle, showing how FeelHippo AUV is able to follow a desired trajectory without incurring in an unacceptable position estimation error growth over time; then, it will be shown how the payload the vehicle is equipped with can be suitably used to accomplish the goals of the competition. Despite of the reduced size, its optimized mechatronics design, indeed leads to a compact but high-functional vehicle.

4.1. Navigation Results

The results reported in this section refer to the mission performed by FeelHippo AUV during the final trial of the competition; hence, the path executed by the vehicle was planned according to the estimated positions of the objects of interest, evaluated from the in-water runs executed during the previous days. In particular, after passing through the gate, the vehicle

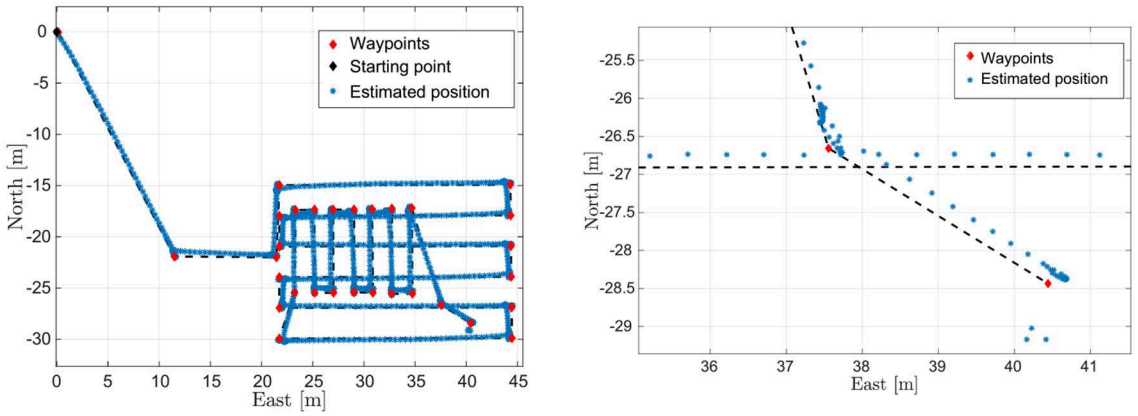


FIGURE 6 | On the left, FeelHippo AUV estimated path, whereas on the right focus on the resurfacing position.

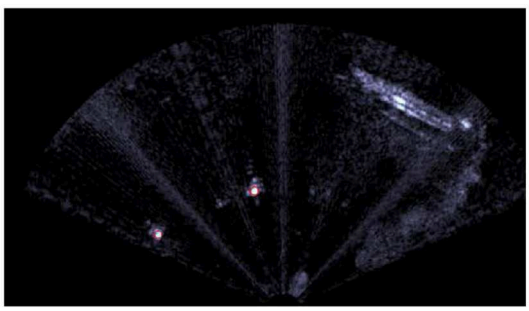


FIGURE 7 | The starting gate, the red circles on the target buoys state the detection.

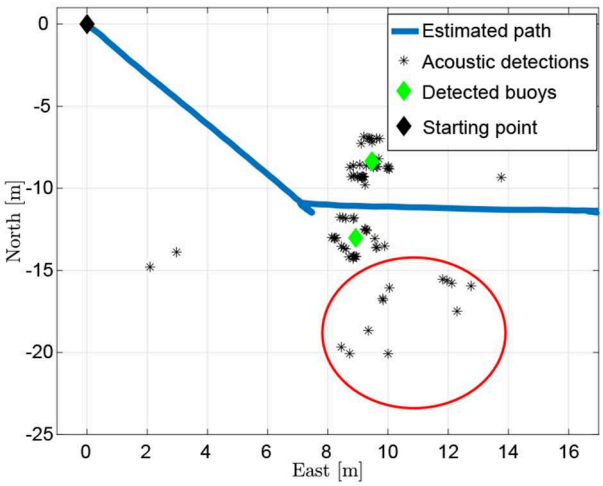


FIGURE 8 | The acoustic detection of the starting gate with the aid of the clustering algorithm.

autonomously performed a lawnmower path with West-East aligned transects, to cover as much as possible of the area of interest. Then, a second lawnmower path, perpendicular to the first, was executed in the northern part of the arena, where the plume buoys were supposed to be. Two final waypoints were included in the direction of the debris area in order to try to identify the objects composing the area itself or even the mannequin representing the worker.

Figure 6 shows the position estimate computed by FeelHippo AUV during the execution of the autonomous mission. The first waypoint (the starting point of **Figure 5**) was located at 42.954164° N, 10.6018952° E; the task was executed at the desired depth of 1 m (except for the last two waypoints, located at the depth of 3.5 m), with a desired longitudinal speed of 0.5 m/s and a covered path of about 240 m. The discontinuity visible in the lower-right corner of **Figure 6** is due to the error between the path estimated onboard the vehicle while navigating underwater and the GPS fix acquired after resurfacing. Indeed, such error is $<$1 m after about 21 min of navigation (or, equivalently, $<$1% of the total length of the path), highlighting the satisfying accuracy of the navigation system of the vehicle: it is worth remembering that FeelHippo AUV performed the whole underwater mission autonomously, without resurfacing; communication from the ground control station to the vehicle (exception made for mission starts and possible emergency aborts) was specifically forbidden by the competition rules.

4.2. Payload Processing Results

FeelHippo AUV was asked to autonomously (and possibly real-time) find the seven buoys located in the sea domain arena, see **Figure 5**. Their physical characteristics in terms of color (orange), shape (approximately spherical) and dimensions (radius around 0.3 m) were a priori known.

While FeelHippo AUV was performing the path described in section 4.1, the buoys detection took place. The starting gate, composed of two buoys, is visible in **Figure 7**, whereas the result of the proposed solution is depicted in **Figure 8**. In the former, the rubber boat where the judges monitored the course of the competition can be noticed on the top-right corner. In the latter, due to the presence of the rubber boat, false positive detections

take place (note the red circle). On the other hand, given their scattered nature, the clustering algorithm is able to handle the situation. In particular, before applying the clustering algorithm 82 detections take place, where 67 are true positives and 15 false positives. It is worth highlighting that the detection operation, as well as the target geolocalization, were conducted in real-time and any geometric constraints has been exploited for target detection (for example, the known geometric distance between the buoys that compose the starting gate).

After the end of the competition, a 2D mosaic of the area around the structure (see **Figures 5**, **9**), namely an underwater plastic pipe assembly with the aim of representing the (flooded) lower section of the plant, was performed. For a detailed description of the acoustic mosaic formation process, the interested reader can refer to Franchi et al. (2018). The proposed solutions make use of the OpenCV library (OpenCV). To this end, a new mission, where the FLS was mounted with a small tilt angle ($\sim 20^\circ$ with respect to the water surface), was executed. The collected dataset was composed of 72 FLS images recorded along a 20-meter transect. The maximum FLS range was set to 10 meters and the FOV of the device was 130° (uneditable by the user). A few FLS frames and the final composite are reported in **Figure 10**. In the latter, the covered area is ~ 500 m^2. Furthermore, the real dimensions of the underwater structure (which were a priori known) are in accordance with the size that can be obtained from **Figure 10**. Indeed, structure dimensions are about 2.20 × 3.20 × 1.20 m, whereas the obtained ones are 2.20×3.46 m. More information concerning the conversion from pixels to meters is presented by the authors in Franchi et al. (2018).

FIGURE 9 | The structure placed on the sea bottom (Ferri et al., 2017).

5. CONCLUSION

The paper shows how FeelHippo AUV, despite its small size, represents a compact and complete underwater platform, which can be employed in different application scenarios.

In particular, a reliable and versatile navigation system, able to perform satisfying accuracies, is shown in section 3.1; indeed, two navigation approaches (the vehicle can exploit a dead reckoning strategy as well as a UKF-based solution) that present a relative error <1% after about 21 min of autonomous navigation are proposed.

For what concerns the payload acquisition and processing, an acoustic-based object detection algorithm (in our case, applied to underwater buoys) is treated in section 3.2.1, where substantial improvements through clustering techniques (usable in presence of static targets) are presented (**Figure 8**). Good performance in terms of detection even with limited visibility ranges are shown and, in addition, the real-time implementation is proposed.

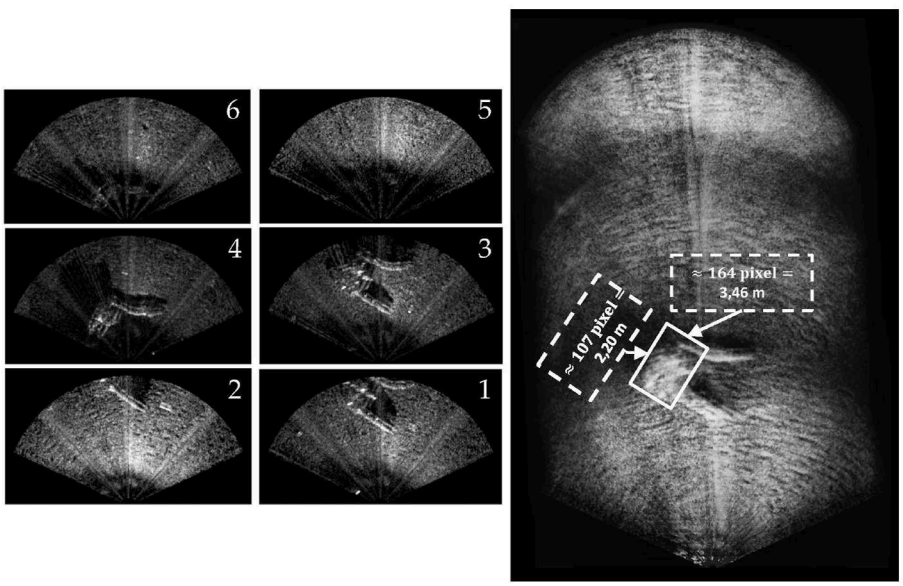


FIGURE 10 | The 2D mosaic of the underwater structure. On the right, the dimension of the underwater structure (retrieved by means of the mosaic) is reported.

Lastly, an underwater acoustic mosaic is presented in section 4.2. The presented solution is shown to perform satisfying 2D underwater reconstruction of the order of hundreds of square meters. Future works will involve machine learning-based detection techniques and a mixed detection approach that resorts to a FLS and an optical camera.

The UNIFI Team has been awarded Second-in-Class in "Pipe inspection and search for search for missing workers (Sea+Air)" during ERL Emergency Robots 2017.

DATA AVAILABILITY STATEMENT

The datasets generated for this study are available on request to the corresponding author.

AUTHOR CONTRIBUTIONS

MF: algorithm development, experiments, results validation, and writing. FF: experiments and results validation. MB: experiments and writing. AR: results validation, writing, and activities supervision. BA: writing and activities supervision.

ACKNOWLEDGMENTS

The authors would like to acknowledge the support of Mr. Nicola Palma and Mr. Tommaso Merciai in the development of the clustering algorithm and the acoustic mosaicing (section 4.2). Moreover, a special thank is addressed to the whole UNIFI Team.

REFERENCES

Allotta, B., Caiti, A., Costanzi, R., Fanelli, F., Fenucci, D., Meli, E., et al. (2016). A new auv navigation system exploiting unscented kalman filter. *Ocean Eng.* 113, 121–132. doi: 10.1016/j.oceaneng.2015.12.058

Allotta, B., Costanzi, R., Fanelli, F., Monni, N., and Ridolfi, A. (2015). Single axis fog aided attitude estimation algorithm for mobile robots. *Mechatronics* 30, 158–173. doi: 10.1016/j.mechatronics.2015.06.012

Amaran, M. H., Noh, N. A. M., Rohmad, M. S., and Hashim, H. (2015). A comparison of lightweight communication protocols in robotic applications. *Proc. Comp. Sci.* 76, 400–405. doi: 10.1016/j.procs.2015.12.318

CADDY (2019). *Official Website of the CADDY Project*. Available online at: http://www.caddy-fp7.eu (accessed January 2020).

Caiti, A., Costanzi, R., Fenucci, D., Allotta, B., Fanelli, F., Monni, N., et al. (2018). "Marine robots in environmental surveys: current developments at isme—localisation and navigation," in *Marine Robotics and Applications* (Springer), 69–86.

Carreras, M., Candela, C., Ribas, D., Palomeras, N., Magií, L., Mallios, A., et al. (2018). Testing sparus ii auv, an open platform for industrial, scientific and academic applications. *arXiv preprint arXiv:1811.03494*.

Costanzi, R., Fanelli, F., Meli, E., Ridolfi, A., Caiti, A., and Allotta, B. (2018). Ukf-based navigation system for auvs: online experimental validation. *IEEE J. Ocean. Eng.* 44, 633–641. doi: 10.1109/JOE.2018.2843654

Costanzi, R., Fanelli, F., Monni, N., Ridolfi, A., and Allotta, B. (2016). An attitude estimation algorithm for mobile robots under unknown magnetic disturbances. *IEEE/ASME Trans. Mech.* 21, 1900–1911.

Crowell, J. (2013). "Design challenges of a next generation small auv," in *Oceans-San Diego, 2013* (San Diego, CA: IEEE), 1–5.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, eds E. Simoudis, J. Han, and U. M. Fayyad (Portland, OR: AAAI Press), 226–231.

Fanelli, F. (2019). *Development and Testing of Navigation Algorithms for Autonomous Underwater Vehicles*. Springer.

Ferreira, F., Djapic, V., Micheli, M., and Caccia, M. (2014). Improving automatic target recognition with forward looking sonar mosaics. *IFAC Proc. Vol.* 47, 3382–3387. doi: 10.3182/20140824-6-ZA-1003.01485

Ferri, G., Ferreira, F., and Djapic, V. (2015). "Boosting the talent of new generations of marine engineers through robotics competitions in realistic environments: the sauc-e and eurathlon experience," in *OCEANS 2015-Genova* (Genoa: IEEE), 1–6.

Ferri, G., Ferreira, F., and Djapic, V. (2017). "Multi-domain robotics competitions: the cmre experience from sauc-e to the european robotics league emergency robots," in *OCEANS 2017-Aberdeen* (Aberdeen, UK: IEEE), 1–7.

Ferri, G., Ferreira, F., Djapic, V., Petillot, Y., Franco, M. P., and Winfield, A. (2016). The eurathlon 2015 grand challenge: The first outdoor multi-domain search and rescue robotics competition—a marine perspective. *Mar. Techn. Soc. J.* 50, 81–97. doi: 10.4031/MTSJ.50.4.9

Fietz, D., Hagedorn, D., Jähne, M., Kaschube, A., Noack, S., Rothenbeck, M., et al. (2014). *Robbe 131: The Autonomous Underwater Vehicle of the auv Team Tomkyle*. Available online at: https://auv-team-tomkyle.de/?page_id=411

Fossen, T. I. et al. (1994). *Guidance and Control of Ocean Vehicles*. Chichester; New York, NY: Wiley New York.

Franchi, M., Ridolfi, A., and Zacchini, L. (2018). "A forward-looking sonar-based system for underwater mosaicing and acoustic odometry," in *Autonomous Underwater Vehicles (AUV), 2018 IEEE/OES* (Porto: IEEE).

Franchi, M., Ridolfi, A., Zacchini, L., and Benedetto, A. (2019). "Experimental evaluation of a Forward-Looking SONAR-based system for acoustic odometry," in *Proceedings of OCEANS'19 MTS/IEEE MARSEILLE, Marseille (FR)* (IEEE).

Furlong, M. E., Paxton, D., Stevenson, P., Pebody, M., McPhail, S. D., and Perrett, J. (2012). "Autosub long range: a long range deep diving auv for ocean monitoring," in *Autonomous Underwater Vehicles (AUV), 2012 IEEE/OES* (Southampton, UK: IEEE), 1–7.

Hiller, T., Steingrimsson, A., and Melvin, R. (2012). "Expanding the small auv mission envelope; longer, deeper & more accurate," in *Autonomous Underwater Vehicles (AUV), 2012 IEEE/OES* (Southampton, UK: IEEE), 1–4.

Hurtós, N., Ribas, D., Cufí, X., Petillot, Y., and Salvi, J. (2015). Fourier-based registration for robust forward-looking sonar mosaicing in low-visibility underwater environments. *J. Field Robot.* 32, 123–151. doi: 10.1002/rob.21516

Johannsson, H., Kaess, M., Englot, B., Hover, F., and Leonard, J. (2010). "Imaging sonar-aided navigation for autonomous underwater harbor surveillance," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Taipei: IEEE), 4396–4403.

Kaiser, C. L., Yoerger, D. R., Kinsey, J. C., Kelley, S., Billings, A., Fujii, J., et al. (2016). "The design and 200 day per year operation of the autonomous underwater vehicle sentry," in *Autonomous Underwater Vehicles (AUV), 2016 IEEE/OES* (Tokyo: IEEE), 251–260.

Mahony, R., Hamel, T., and Pflimlin, J.-M. (2008). Nonlinear complementary filters on the special orthogonal group. *IEEE Trans. Automat. Control* 53, 1203–1218. doi: 10.1109/TAC.2008.923738

McCarter, B., Portner, S., Neu, W. L., Stilwell, D. J., Malley, D., and Minis, J. (2014). "Design elements of a small auv for bathymetric surveys," in *Autonomous Underwater Vehicles (AUV), 2014 IEEE/OES* (Oxford, MS: IEEE), 1–5.

Mišković, N., Bibuli, M., Birk, A., Caccia, M., Egi, M., Grammer, K., et al. (2016). Caddy—cognitive autonomous diving buddy: two years of underwater

human-robot interaction. *Mar. Technol. Soc. J.* 50, 54–66. doi: 10.4031/MTSJ.50.4.11

opencv (2019). *Official Website of the Open Source Computer Vision Library*. www.opencv.org. Available online at: (accessed January 2020).

Rigaud, V. (2007). Innovation and operation with robotized underwater systems. *J. Field Robot.* 24, 449–459. doi: 10.1002/rob.20195

ROS (2019). *Official Website of Robot Operating System (ROS)*. Available online at: http://www.ros.org (accessed January 2020).

Walter, M. R. (2008). *Sparse bayesian information filters for localization and mapping*. (Ph.D. disseration). Massachusetts Institute of Technology and Woods Hole Oceanographic Institution, Cambridge, MA, United States.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Robotics and AI

ORIGINAL RESEARCH
published: 11 February 2020
doi: 10.3389/frobt.2020.00011

Hybrid Collision Avoidance for ASVs Compliant With COLREGs Rules 8 and 13–17

Bjørn-Olav H. Eriksen, Glenn Bitar, Morten Breivik and Anastasios M. Lekkas*

Department of Engineering Cybernetics, Centre for Autonomous Marine Operations and Systems, Norwegian University of Science and Technology, Trondheim, Norway

This paper presents a three-layered hybrid collision avoidance (COLAV) system for autonomous surface vehicles, compliant with rules 8 and 13–17 of the International Regulations for Preventing Collisions at Sea (COLREGs). The COLAV system consists of a high-level planner producing an energy-optimized trajectory, a model-predictive-control-based mid-level COLAV algorithm considering moving obstacles and the COLREGs, and the branching-course model predictive control algorithm for short-term COLAV handling emergency situations in accordance with the COLREGs. Previously developed algorithms by the authors are used for the high-level planner and short-term COLAV, while we in this paper further develop the mid-level algorithm to make it comply with COLREGs rules 13–17. This includes developing a state machine for classifying obstacle vessels using a combination of the geometrical situation, the distance and time to the closest point of approach (CPA) and a new CPA-like measure. The performance of the hybrid COLAV system is tested through numerical simulations for three scenarios representing a range of different challenges, including multi-obstacle situations with multiple simultaneously active COLREGs rules, and also obstacles ignoring the COLREGs. The COLAV system avoids collision in all the scenarios, and follows the energy-optimized trajectory when the obstacles do not interfere with it.

Keywords: hybrid collision avoidance, autonomous surface vehicle (ASV), COLREGs, COLREGs compliant, model predictive control (MPC), energy-optimized control

OPEN ACCESS

Edited by:
Marco Bibuli,
Italian National Research Council, Italy

Reviewed by:
Jingfu Jin,
General Motors, United States
Rafal Szlapczynski,
Gdansk University of Technology, Poland

****Correspondence:***
Bjørn-Olav H. Eriksen
bjorn-olav.holtung.eriksen@ntnu.no

Specialty section:
This article was submitted to Robotic Control Systems, a section of the journal Frontiers in Robotics and AI

Received: *28 May 2019*
Accepted: *20 January 2020*
Published: *11 February 2020*

Citation:
Eriksen B-OH, Bitar G, Breivik M and Lekkas AM (2020) Hybrid Collision Avoidance for ASVs Compliant With COLREGs Rules 8 and 13–17. Front. Robot. AI 7:11. doi: 10.3389/frobt.2020.00011

1. INTRODUCTION

Motivated by the potential for reduced costs and increased safety, the maritime industry is rapidly moving toward autonomous operations. Following groundbreaking advances in the automotive industry, many sectors within the maritime industry are considering the benefits of autonomy, which includes more environmentally friendly operations. For instance, the agricultural chemical company *Yara* together with the maritime technology supplier *Kongsberg Maritime* are developing the electrical autonomous container vessel *Yara Birkeland*, which aims to replace 40,000 yearly truck journeys in urban eastern Norway[1]. Another example is the world's first autonomous car ferry, *Falco*, developed by *Rolls-Royce* (recently bought by Kongsberg Maritime) and *Finferries*. In

[1]https://www.wsj.com/articles/norway-takes-lead-in-race-to-build-autonomous-cargo-ships-1500721202 (accessed May 22, 2019).

2018, *Falco* navigated autonomously between two ports in Finland[2]. Reports state that in excess of 75 % of maritime accidents are due to human errors (Chauvin, 2011; Levander, 2017), indicating that there is also a potential for increased safety in addition to the economical and environmental benefits.

An obvious prerequisite for autonomous ship operations is the development of robust and well-functioning COLAV systems. In addition to generating collision-free maneuvers, a COLAV system must adhere to the "rules of the road" of the oceans, i.e., the COLREGs (Cockcroft and Lameijer, 2004). These rules are written for human ship operators and include qualitative requirements on how to perform safe and readily observable maneuvers. Part B of the COLREGs concern steering and sailing, and includes the following rules that are the most relevant to a motion control system:

Rule 8 Requires maneuvers to be readily observable and to be done in ample time.

Rules 13–15 Describe the maneuvers to perform in cases of overtaking, head-on and crossing situations. Participants in crossing situations are defined by the terms *give-way* and *stand-on* vessels.

Rule 16 Requires that a give-way vessel must take early and substantial action to keep clear of the stand-on vessel.

Rule 17 Consists of two main parts. The first part requires a stand-on vessel to maintain its course and speed, while the second part allows/requires[3] a stand-on vessel to take action to avoid collision if the give-way vessel is not taking action.

Since the rules are written for humans, with few quantitative figures, a challenge for autonomous operation is to quantify them into behaviors that can be executed algorithmically. The focus of the work in this paper is to do that, and to design a hybrid COLAV system that performs motion planning and generates maneuvers in compliance to rules 8 and 13–17 of the COLREGs.

A number of COLAV approaches considering the COLREGs have been proposed in the past. This includes algorithms using simulation-based model predictive control (Hagen et al., 2018), velocity obstacles (Kuwata et al., 2014), rule-based repairing A* (Campbell et al., 2014), and interval programming (Benjamin et al., 2006). All these approaches are single-layer approaches, where one algorithm solves the complete COLAV problem.

Another approach to the COLAV problem is to use a hybrid architecture, where the task of planning an obstacle-free path or trajectory, complying with the COLREGs and ultimately performing safe maneuvers is divided into layers in a control hierarchy. The idea of hybrid architectures is to divide the subtasks of the COLAV problem into multiple algorithms, exploiting their complementary strengths. This also has the side effect of making it easier for human operators or supervisors to understand the system. Most single-layer algorithms use sample-based approaches that consider a finite number of discrete control inputs, as opposed to conventional gradient-based search algorithms. The reason for this is that many gradient-based algorithms are not sufficiently numerically robust, not allowing a COLAV system to solely rely on such an algorithm. This issue can be handled in hybrid architectures, constrained by the bottom-level algorithm being numerically robust and able to handle extraordinary situations where the other algorithms fail. Hence, hybrid architectures also allows using gradient-based algorithms, which are able to solve problems with large search spaces more efficiently than sample-based algorithms. The works by Loe (2008) and Švec et al. (2013) are examples of two-layered hybrid COLAV architectures. The top layers perform trajectory planning among static obstacles, while the bottom layers perform moving obstacle avoidance in compliance with COLREGs rules 13–16. Casalino et al. (2009) presents a three-layered hybrid COLAV system where the top layer also performs trajectory planning amongst static obstacles. The middle layer avoids moving obstacles, while the bottom layer implements safety functions for handling cases where the two

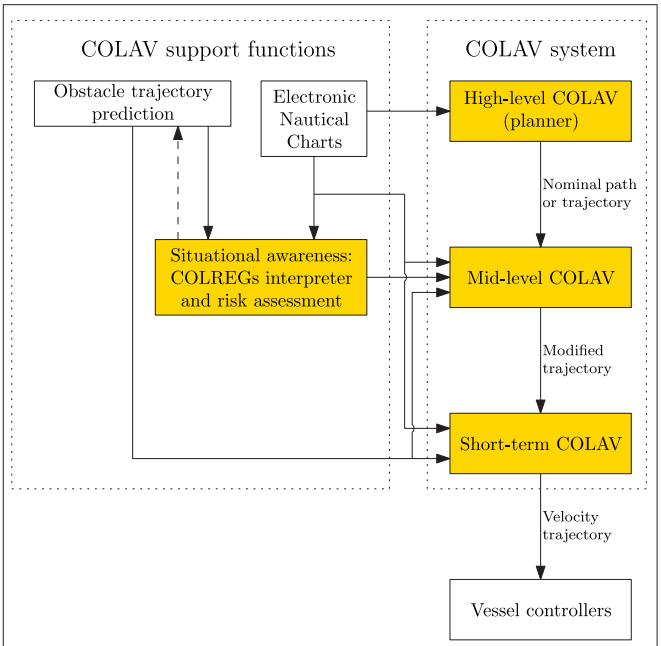


FIGURE 1 | Hybrid architecture with three COLAV layers, where the highlighted functions mark the areas of interest in this article. The COLAV system consists of a high-level planner, a mid-level COLAV algorithm and a short-term COLAV algorithm. The COLAV system is supported by data from electronic nautical charts, represented in a suitable manner for the algorithms, as well as situational awareness functions that track and predict obstacles, interpret the COLREGs and perform risk assessment.

[2]https://www.marinemec.com/news/view,rollsroyce-and-finferries-demonstrate-worlds-first-fully-autonomous-ferry_56102.htm (accessed April 11, 2019).

[3]Rule 17 *allows* the stand-on vessel to maneuver when it becomes apparent that the give-way vessel does maneuver to avoid collision. If the vessels are so close that the give-way vessel cannot avoid collision by itself, Rule 17 *requires* the stand-on vessel to maneuver.

other layers fail. This approach does, however, not consider the COLREGs.

Figure 1 shows a three-layered hybrid COLAV system for an autonomous surface vehicle (ASV). The authors have previously worked extensively on different components of this architecture. Examples include high-level COLAV algorithms (Bitar et al., 2018, 2019b), a mid-level algorithm (Eriksen and Breivik, 2017b; Bitar et al., 2019a), short-term algorithms (Eriksen et al., 2018, 2019; Eriksen and Breivik, 2019) and the development of high-performance vessel controllers (Eriksen and Breivik, 2017a, 2018).

In this paper, we demonstrate the three-layered hybrid COLAV shown in **Figure 1** by combining and extending the COLAV algorithms developed in Eriksen and Breivik (2017b, 2019), Bitar et al. (2019a,b), Eriksen et al. (2019). The high-level planner has a long temporal horizon, and finds an energy-optimized nominal trajectory from an initial to a goal position. It considers static obstacles, which may include bathymetric constraints. Since the high-level planner only considers static information, it is intended to be run offline, but it can also be run online, for instance if new static obstacles are detected. The mid-level algorithm attempts to follow the nominal trajectory, while performing COLAV of static and moving obstacles in compliance with COLREGs rules 8, 13–16, and the first part of Rule 17. The mid-level algorithm is run periodically with a shorter temporal horizon than the high-level algorithm, and produces a modified trajectory which is passed to the short-term layer. Both the high-level and mid-level algorithms use gradient-based optimization. The short-term algorithm attempts to follow the modified trajectory, while it in compliance with the second part of Rule 17 handles situations where obstacles ignore the COLREGs. This algorithm also handles other emergency situations, and uses sample-based optimization to achieve a high level of robustness, ensuring safe operation if the mid-level algorithm fails to find a solution. The following list summarizes our contributions:

- The high-level planner from Bitar et al. (2019b) is modified to include the mathematical model of the *Telemetron* ASV in Bitar et al. (2019a), including ocean currents.
- The development of a state-machine-based COLREGs interpretation scheme.
- The mid-level COLAV from Bitar et al. (2019a) is modified to include rules 13–16 and the first part of Rule 17.
- The branching-course model predictive control (BC-MPC) algorithm for short-term COLAV is modified to reduce oscillatory behavior in turns.
- The three-layered COLAV system is verified in simulations and shown to be compliant with rules 8 and 13–17.

The rest of the paper has the following structure: The mathematical model of the ASV *Telemetron* is described in section 2. The high-level planner, mid-level and short-term COLAV algorithms are described in sections 3–5, respectively. In section 6 we present and discuss the simulation scenarios and results, and we conclude the paper in section 7.

FIGURE 2 | The Telemetron ASV, designed for both manned and unmanned operations. Courtesy of Maritime Robotics.

2. ASV MODELING

The vessel of interest in this article is the Telemetron ASV, which is owned and operated by the Norwegian company Maritime Robotics and shown in **Figure 2**. The Telemetron ASV is a high-speed dual-use vessel propelled by a steerable outboard engine, capable of speeds up to 18 m/s.

Eriksen and Breivik (2017a) presents a model of the Telemetron ASV, which is extended to include ocean currents in Bitar et al. (2019a). The model has the form

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{R}(\psi)\boldsymbol{x}_r + \begin{bmatrix} \boldsymbol{V}_c^\top & 0 \end{bmatrix}^\top \\ \boldsymbol{M}(\boldsymbol{x}_r)\dot{\boldsymbol{x}}_r + \boldsymbol{\sigma}(\boldsymbol{x}_r) &= \boldsymbol{\tau}\,, \end{aligned} \tag{1}$$

where $\boldsymbol{\eta} = [x, y, \psi]^\top \in \mathbb{R}^2 \times S$ is the vessel pose and $\boldsymbol{V}_c = [V_x, V_y]^\top$ describes the ocean current, both in the Earth-fixed North-East-Down frame $\{n\}$. The vector $\boldsymbol{x}_r = [u_r, r]^\top \in \mathbb{X}_r \subset \mathbb{R}^2$ is the vessel velocity under the assumption of zero relative sway motion (Bitar et al., 2019a), where the set $\mathbb{X}_r$ describes the vessel-feasible steady-state velocities where (1) is valid. The transformation matrix $\boldsymbol{R}(\psi)$ is given by the heading $\psi \in S$ as

$$\boldsymbol{R}(\psi) = \begin{bmatrix} \cos\psi & 0 \\ \sin\psi & 0 \\ 0 & 1 \end{bmatrix}, \tag{2}$$

while $r \in \mathbb{R}$ describes the vessel yaw-rate. The matrix $\boldsymbol{M}(\boldsymbol{x}_r)$ is a state-dependent inertia matrix, while $\boldsymbol{\sigma}(\boldsymbol{x}_r)$ and $\boldsymbol{\tau} = [\tau_m, \tau_\delta]^\top \in \mathbb{U} \subset \mathbb{R}^2$ describe the vessel damping and control input, respectively. The set $\mathbb{U}$ describes the control inputs where (1) is valid.

In this work, we assume that the ocean current $\boldsymbol{V}_c$ is constant and known. For practical applications, the ocean current can be measured via appropriate instrumentation, estimated via sensor fusion methods, or predicted based on e.g., tide tables or sensor networks, such as the European marine observation and data network[4].

[4]http://www.emodnet.eu/ (accessed December 11, 2019).

3. HIGH-LEVEL PLANNER

To plan the ASV's nominal trajectory, we use a high-level trajectory planner developed in Bitar et al. (2019b). This trajectory planner uses the ASV model described in section 2 to generate an energy-optimized trajectory between the start and goal positions. The planning algorithm combines an A* implementation and an optimal control problem (OCP) solver to generate a feasible and optimized trajectory.

The high-level planning algorithm consists of three steps: First the A* implementation finds the shortest piecewise linear path between the start and goal position. Secondly, artificial temporal information is added to the path, converting it to a trajectory of states and inputs. Finally, the trajectory is used as an initial guess for an OCP solver, which finds a locally energy-optimized trajectory near the shortest path. All steps account for static obstacles in the form of elliptical boundaries.

3.1. Static Obstacles

The elliptical boundaries are described with the inequality:

$$\left(\frac{x-x_c}{x_a}\right)^2+\left(\frac{y-y_c}{y_a}\right)^2\geq 1\,, \tag{3}$$

where x_c and y_c is the ellipsis center, and $x_a, y_a > 0$ are the ellipsis major and minor axes, respectively. To allow for angled obstacles, the ellipses are rotated clockwise by an angle α. We add a small constant $\epsilon > 0$ to each side of the inequality, and take the logarithm to arrive at the following obstacle representation:

$$h_o(x,y,x_c,y_c,x_a,y_a,\alpha)=-\log\left[\left(\frac{(x-x_c)\cos\alpha+(y-y_c)\sin\alpha}{x_a}\right)^2+\left(\frac{-(x-x_c)\sin\alpha+(y-y_c)\cos\alpha}{y_a}\right)^2+\epsilon\right]+\log(1+\epsilon)\leq 0\,. \tag{4}$$

The logarithm operation is applied to reduce the numerical range of the inequality, which helps with numerical stability of the subsequently described solver, and the constant ϵ is included to avoid singularities when (4) is evaluated for $(x,y)\to(x_c,y_c)$ (Bitar et al., 2019a).

Modeling static obstacles as ellipses poses a challenge for handling obstacles of various shapes, from e.g., electronic nautical charts (ENCs). Generic obstacle shapes can be approximated as a set of elliptical obstacles (Wu, 2019), although this may require a large number of constraints for complex environments. Alternatively, the obstacle modeling can be modified to allow for generic shapes. Zhang et al. (2018) present an interesting solution to handle polygon-shaped obstacles by introducing a signed distance function in the optimization problem. Unfortunately, this approach introduces a large number of slack variables and constraints, limiting feasibility for more than a few static obstacles.

3.2. Trajectory Generation and Optimization

From a scenario consisting of static obstacles, as mentioned in section 3.1, we find the piecewise linear shortest path by performing an A* search on a uniformly decomposed grid. The resulting path is converted to a time-parameterized full-state trajectory by assuming a constant forward velocity, and connecting the shortest path with straight segments and circle arcs. The constant forward velocity is

$$u_{\text{nom}}=\frac{L_{\text{path}}}{t_{\max}}\,, \tag{5}$$

where L_{path} is the length of the connected path, and $t_{\max}$ is the maximum allowed time to complete the trajectory. This full-state trajectory is then used as an initial guess to solve the OCP that gives the energy-optimized trajectory:

$$\min_{\boldsymbol{z}(\cdot),\boldsymbol{\tau}(\cdot)}\int_0^{t_{\max}} F_{\text{hi}}(\boldsymbol{z}(t),\boldsymbol{\tau}(t))\mathrm{d}t \tag{6a}$$

subject to

$$\dot{\boldsymbol{z}}(t)=\boldsymbol{f}(\boldsymbol{z}(t),\boldsymbol{\tau}(t))\;\forall t\in[0,t_{\max}] \tag{6b}$$

$$\boldsymbol{h}_{\text{hi}}(\boldsymbol{z}(t),\boldsymbol{\tau}(t))\leq\boldsymbol{0}\;\forall t\in[0,t_{\max}] \tag{6c}$$

$$\boldsymbol{e}_{\text{hi}}(\boldsymbol{z}(0),\boldsymbol{z}(t_{\max}))=\boldsymbol{0}\,. \tag{6d}$$

The solution of this OCP is a trajectory of states $\boldsymbol{z}(\cdot)$ and inputs $\boldsymbol{\tau}(\cdot)$ that minimizes the cost functional in (6a). The ASV model from section 2 is rewritten as $\dot{\boldsymbol{z}}=\boldsymbol{f}(\boldsymbol{z},\boldsymbol{\tau})$, where $\boldsymbol{z}=[\boldsymbol{\eta}^\top,\boldsymbol{x}_r^\top]^\top$ and $\boldsymbol{f}(\boldsymbol{z},\boldsymbol{\tau})$ represents (1).

The cost functional (6a) is chosen to minimize energy. The cost-to-go function is

$$F_{\text{hi}}(\boldsymbol{z},\boldsymbol{\tau})=K_eF_e(\boldsymbol{z},\boldsymbol{\tau})+K_\delta\tau_\delta^2, \tag{7}$$

with tuning parameters $K_e, K_\delta > 0$. The first term consists of a function that is proportional to mechanical work performed by the ASV:

$$F_e(\boldsymbol{z},\boldsymbol{\tau})=|\underbrace{n(\tau_m)^2\cdot\cos\delta(\tau_\delta)}_{\propto\text{ surge force}}\cdot u_r|+|\underbrace{n(\tau_m)^2\cdot\sin\delta(\tau_\delta)\cdot L_m}_{\propto\text{ yaw moment}}\cdot r|\,. \tag{8}$$

The function $n:\mathbb{R}^+\to\mathbb{R}^+$ maps the control input τ_m to propeller angular velocity. The function $\delta:\mathbb{R}\to\mathrm{S}$ maps the control input τ_δ to outboard motor angle. The second term in (8) is a quadratic cost to yaw control, included to avoid issues with singularity when solving the OCP.

The inequality constraints (6c) observe state boundaries as well as the static obstacles as represented in section 3.1. The boundary conditions (6d) denote initial and final constraints, i.e., start and end states.

A detailed description of the transcription of the OCP (6) to a non-linear program (NLP) using multiple shooting with N_{hi} shooting intervals is found in Bitar et al. (2019b).

4. MID-LEVEL COLAV

The mid-level algorithm, initially presented in Eriksen and Breivik (2017b) and further developed in Bitar et al. (2019a), is a model predictive control (MPC)-based algorithm intended for long-term COLAV. The algorithm utilizes gradient-based

optimization, and takes both static and moving obstacles into account while attempting to follow an energy-optimized nominal trajectory from the high-level planner. The algorithm produces maneuvers complying with Rule 8 of the COLREGs, which requires maneuvers to be made in ample time and be readily observable for other vessels. The optimization problem is formulated as a NLP, which gives flexibility in designing the optimization problem.

In this section, the algorithm is extended to also consider COLREGs rules 13–16 and the first part of Rule 17.

4.1. The International Regulations for Preventing Collisions at Sea (COLREGs)

The COLREGs consists of a total of 37 rules and is divided into five parts (Cockcroft and Lameijer, 2004), where part B (rules 4–19) contains relevant rules on the conduct of vessels in proximity of each other. The most relevant rules for designing COLAV systems in part B are rules 8 and 13–17:

Rule 8 **Action to avoid collision.** This rule states that actions taken to avoid collision should be large enough to be readily observable of other ships, implying that series of small alternations in speed and/or course should not be applied. The rule also recommends that course changes should be prioritized over speed changes if there is enough free space available, and that maneuvers must be made in ample time.

Rule 13 **Overtaking.** This rule states that a vessel is overtaking another if it approaches the other vessel with a course more than 22.5° abaft her beam. The overtaking vessel has to stay clear of the overtaken vessel, but there is no statements on which side of the vessel one should pass.

Rule 14 **Head on.** When two power-driven vessels approach each other on reciprocal, or nearly reciprocal, courses, they are in a head-on situation. In such a situation, both vessels should change their course to starboard, passing each other port-to-port, as shown in **Figure 3A**. This rule states no explicit definition on what should be considered to be reciprocal, or nearly reciprocal, courses, but court decisions indicate head-on situations exist for opposing courses ±6°. Notice that the rule does not include sailing vessels, which are covered by Rule 12.

Rule 15 **Crossing.** When two vessels approach each other such that the situation is not a head on or an overtaking, it is a crossing situation. The vessel with the other one to her starboard side is deemed the give-way vessel, while the other vessel is deemed the stand-on vessel. As shown in **Figure 3B**, the give-way vessel should maneuver to avoid collision, preferably by passing behind the stand-on vessel, while the stand-on vessel should keep her speed and course.

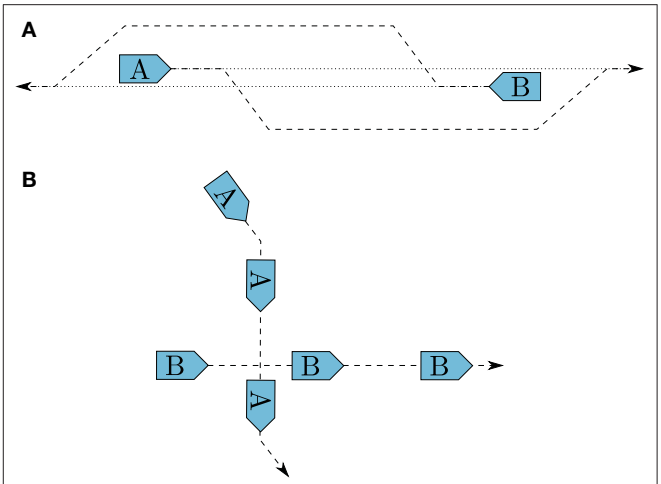


FIGURE 3 | Illustration of head-on **(A)** and crossing **(B)** situations, and how they should be resolved.

Rule 16 **Action by the give-way vessel.** Every vessel which is required to keep out of the way of another vessel should take early and large enough action to safely avoid collision.

Rule 17 **Action by the stand-on vessel.** This rule requires that a stand-on vessel should keep its current speed and course. The stand-on vessel may, however, maneuver to avoid collision if it becomes apparent that the give-way vessel is not taking appropriate actions to avoid collision. Furthermore, if the stand-on vessel finds itself so close to the obstacle that collision can not be avoided by the give-way vessel alone, the stand-on vessel should take such action which best aids to avoid collision. In a crossing situation, the stand-on vessel should avoid maneuvering to port, since this could lead to a collision if the give-way vessel maneuvers to starboard.

In the hybrid architecture illustrated in **Figure 1**, the mid-level algorithm is given the task of strictly enforcing COLREGs rules 13–16 and the stand-on requirement of Rule 17, while also complying with Rule 8.

In addition, we want the mid-level algorithm to comply with the first part of Rule 17, by not maneuvering to avoid collision in crossing situations if the ownship is the stand-on vessel. The hybrid COLAV system is inherently capable of adhering to the remaining requirement of Rule 17, where the stand-on vessel is allowed or required to maneuver, by having different prediction horizons and safety margins in the mid-level and short-term layers. The BC-MPC algorithm does not have any limitations of not maneuvering in stand-on situations, and will hence maneuver in stand-on situations if we come sufficiently close to the obstacle.

The mid-level algorithm as presented in Bitar et al. (2019a) only complies with Rule 8. Further in this section, we therefore present improvements to the mid-level algorithm to make it

comply with rules 13–16 and the stand-on requirement of Rule 17.

4.2. COLREGs Interpretation

A commonly used concept for interpreting obstacles in COLAV algorithms is to assign a spatial region to obstacles, which the ownship should not enter. This approach is commonly referred to as a *domain-based* approach. Specially designed ship domains are commonly used for interpreting the COLREGs in COLAV algorithms, where the required clearance to an obstacle is significantly larger if the maneuver violates the COLREGs (Szlapczynski and Szlapczynska, 2017; Eriksen et al., 2019). This approach is attractive since it continuously captures multiple COLREGs rules, and does not require logic or discrete decisions. However, such an approach does not strictly enforce the COLREGs rules, since it will allow maneuvers violating the rules if they are large enough. In addition, a ship-domain approach will not be able to strictly enforce the stand-on requirement of Rule 17, since a domain-based approach will avoid collision with all obstacles. One could ignore obstacles with give-way obligations, but this would require an explicit COLREGs interpretation which conflicts with domain-based approaches' core idea of implicit COLREGs interpretation. Therefore, we pursue an alternative approach to handling the COLREGs in the mid-level algorithm.

To simplify the COLREGs interpretation task, we look at the situation from a static perspective, assuming that the current COLREGs situations are valid throughout the entire prediction horizon of the mid-level algorithm. In reality, the COLREGs situations may, however, change during the prediction horizon depending on both the ownship's and obstacles future trajectory. For instance, an obstacle approaching from head on, but far enough away to not be considered as a danger may be put in a safe state. Hence, the mid-level algorithm will (for the current iteration) act like no COLREGs rule applies to this vessel for the entire prediction horizon, while the obstacle may get close enough during the prediction horizon to be considered as a head-on situation. An MPC scheme of only implementing a small part of the prediction horizon will reduce the implications of this, since the situation is reassessed each time mid-level algorithm is run, which justifies the assumption of considering the COLREGs from a static perspective. Investigating the possibilities for dynamically predicted future COLREGs situations as part of the MPC prediction will be considered as future work.

4.2.1. State Machine

We propose to utilize a state machine in order to decide which COLREGs rule is active with respect to each obstacle in the vicinity of the ownship. The state machine contains the states:

SF Safe state. This implies that the COLREGs do not enforce any rule with respect to this obstacle.

OT Overtaking state. This implies that COLREGs Rule 13 applies with respect to this obstacle. The state machine does not discriminate on whether the ownship is overtaking another vessel or is being overtaken, but this can be done by looking at which vessel has the higher speed (Tam and Bucknall, 2010).

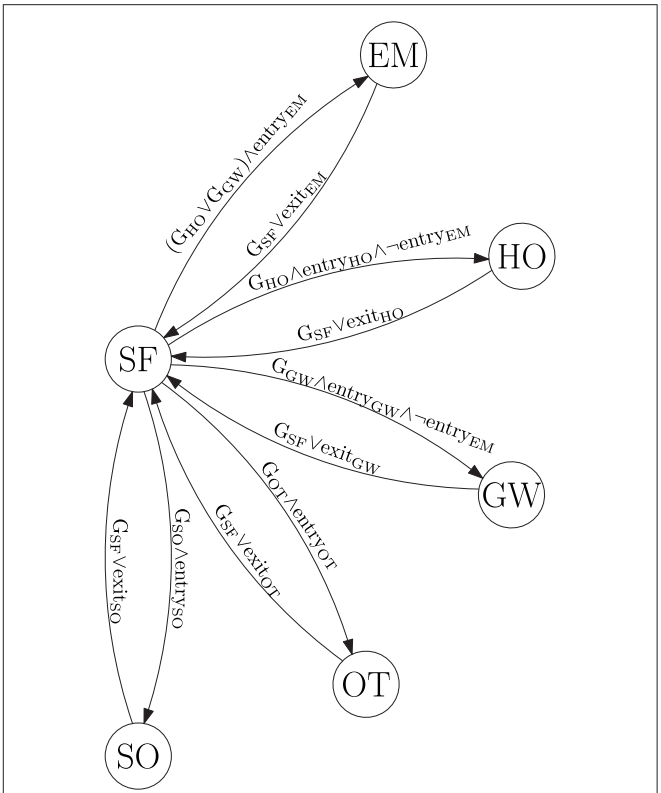


FIGURE 4 | COLREGs state machine. The abbreviations "G_{SF}," "G_{SO}," "G_{OT}," "G_{GW}," and "G_{HO}" denote geometrical situations, while "$entry_{xx}$" and "$exit_{xx}$" denote additional state-dependent entry and exit criterias.

HO Head-on state. This implies that COLREGs Rule 14 applies with respect to this obstacle.

GW Give-way state. This implies that COLREGs Rule 15 applies with respect to this obstacle, and the ownship has to give way.

SO Stand-on state. This implies that COLREGs Rule 15 applies with respect to this obstacle, and the ownship has to stand on.

EM Emergency state. This implies that the obstacle is so close and/or behaves unpredictably, such that special considerations must be made.

As shown in **Figure 4**, all transitions have to go either from or to the safe state.

This implies that when the state machine decides that a COLREGs (or emergency) situation exists with respect to an obstacle, it will not allow switching to another state without the situation being considered as safe first. One could argue that it should be able to transition between specific states, like e.g., from head-on, give-way and overtaking to emergency. This is an interesting topic, which should receive attention in the future. To control the transitions between the different states, we combine the time to and distance at the CPA, a CPA-like measure of the

time until a critical point and a geometrical interpretation of the situation.

4.2.2. Entry and Exit Criteria

CPA is a common concept in maritime risk assessment. Given the current speed and course of the ownship and an obstacle, CPA describes the time to the point where the two vessels are the closest, and the distance to the obstacle at this point. Given the position and velocity vector of the ownship $\boldsymbol{p}, \boldsymbol{v}$ and an obstacle $\boldsymbol{p}_o, \boldsymbol{v}_o$, the time to CPA is calculated as (Kufoalor et al., 2018)

$$t_{\text{CPA}} = \begin{cases} 0 & \text{if } \|\boldsymbol{v} - \boldsymbol{v}_0\|_2 \leq \epsilon \\ \frac{(\boldsymbol{p}-\boldsymbol{p}_o)\cdot(\boldsymbol{v}-\boldsymbol{v}_o)}{\|\boldsymbol{v}-\boldsymbol{v}_o\|_2^2} & \text{else,} \end{cases} \tag{9}$$

where $\epsilon > 0$ is a small constant in order to avoid division by zero in the case where the relative velocity between the ownship and obstacle is zero. Given t_{CPA}, we calculate the distance between the vessels at CPA as

$$d_{\text{CPA}} = \|(\boldsymbol{p} + t_{\text{CPA}}\boldsymbol{v}) - (\boldsymbol{p}_o + t_{\text{CPA}}\boldsymbol{v}_o)\|_2. \tag{10}$$

While the CPA is the point where the distance to an obstacle is at its minimum, the critical point is where the distance to an obstacle crosses underneath a critical distance d_{crit}. This critical distance describes a minimum obstacle distance that the mid-level algorithm is designed for. The time to the critical point t_{crit} can be calculated by solving the equation

$$\|(\boldsymbol{p} + t_{\text{crit}}\boldsymbol{v}) - (\boldsymbol{p}_o + t_{\text{crit}}\boldsymbol{v}_o)\|_2 = d_{\text{crit}}\,. \tag{11}$$

In the cases where the distance between the ships does not fall below d_{crit}, t_{crit} is undefined. Otherwise, there are generally two solutions. The interesting solution is the one with the lowest t_{crit} value, as this is when the obstacle enters the d_{crit} boundary.

The state-machine entry criteria in **Figure 4** are defined as

$$\begin{aligned} \text{entry}_i &= \begin{cases} \text{true} & \text{if } d_{\text{CPA}} < \bar{d}_{\text{CPA}}^{i,\text{enter}} \wedge t_{\text{CPA}} \in [\underline{t}_{\text{CPA}}^{i,\text{enter}}, \bar{t}_{\text{CPA}}^{i,\text{enter}}], \\ \text{false} & \text{otherwise} \end{cases} \\ &\forall i \in \{\text{SO,OT,GW,HO}\} \\ \text{entry}_{\text{EM}} &= \begin{cases} \text{true} & \text{if } t_{\text{crit}} < \bar{t}_{\text{crit}}^{\text{EM,enter}} \wedge t_{\text{CPA}} > 0 \\ \text{false} & \text{otherwise,} \end{cases} \end{aligned} \tag{12}$$

where $\bar{d}_{\text{CPA}}^{i,\text{enter}}$, $\underline{t}_{\text{CPA}}^{i,\text{enter}}$, and $\bar{t}_{\text{CPA}}^{i,\text{enter}}$ for $i \in$ {SO,OT,GW,HO} are tuning parameters denoting thresholds on d_{CPA} and t_{CPA} in order to satisfy the entry criteria for the stand-on, overtaking, give-way and head-on states. The tuning parameter $\bar{t}_{\text{crit}}^{\text{EM,enter}}$ denotes an upper limit on t_{crit} in order to enter the emergency state. The idea behind the stand-on, overtaking, give-way and head-on entry criterias are that in order for the obstacle to represent a risk, both t_{CPA} and d_{CPA} need to be within some tunable thresholds. Situations with a very low d_{CPA}, but with a high t_{CPA}, will not trigger the entry criteria, since the situations will not occur in the near future. Similarly, if t_{CPA} is within the thresholds, but d_{CPA} is large, this indicates a safe passing where risk of collision does not exist. The lower bound on t_{CPA} will typically be selected as zero, and is useful to distinguish between obstacles moving toward of away from the ownship. For the emergency state, the entry criteria is based on the critical point, at which we are so close that the mid-level algorithm may struggle with providing meaningful maneuvers. In addition to t_{crit} being under the threshold $\bar{t}_{\text{crit}}^{\text{EM,enter}}$, we require that t_{CPA} is positive, indicating that we are getting closer to the obstacle. Currently, we only allow entering the emergency state if the situation is a geometrical give-way or head-on, since an overtaking situation represents a smaller danger and has less requirement for special consideration.

The state-machine exit criterias in **Figure 4** are defined as

$$\begin{aligned} \text{exit}_i &= \begin{cases} \text{true} & \text{if } d_{\text{CPA}} \geq \underline{d}_{\text{CPA}}^{i,\text{exit}} \vee t_{\text{CPA}} \notin [\underline{t}_{\text{CPA}}^{i,\text{exit}}, \bar{t}_{\text{CPA}}^{i,\text{exit}}], \\ \text{false} & \text{otherwise} \end{cases} \\ &\forall i \in \{\text{SO,OT,GW,HO}\} \\ \text{exit}_{\text{EM}} &= \begin{cases} \text{true} & \text{if } t_{\text{crit}} \geq \underline{t}_{\text{crit}}^{\text{EM,exit}} \vee t_{\text{CPA}} \leq 0 \\ \text{false} & \text{otherwise,} \end{cases} \end{aligned} \tag{13}$$

where $\underline{d}_{\text{CPA}}^{i,\text{exit}}$, $\underline{t}_{\text{CPA}}^{i,\text{exit}}$, and $\bar{t}_{\text{CPA}}^{i,\text{exit}}$ for $i \in$ {SO,OT,GW,HO} are tuning parameters denoting thresholds on d_{CPA} and t_{CPA} in order to satisfy the exit criteria for the stand-on, overtaking, give-way and head-on states. The exit criteria for the emergency state is satisfied if t_{crit} is larger than the tuning parameter $\underline{t}_{\text{exit}}^{\text{EM,enter}}$, or t_{CPA} is negative, implying that the obstacle is moving further away from the ownship. Note that the exit criterias are obtained by negating the entry criterias, but with other thresholds in order to implement hysteresis to avoid shattering. In general, we allow for different tuning parameters for the different states, but in our simulations we see that selecting the same tuning parameters for all states provides good results. Therefore, we define:

$$\begin{aligned} \bar{d}_{\text{CPA}}^{i,\text{enter}} &= \bar{d}_{\text{CPA}}^{\text{enter}} \\ \underline{t}_{\text{CPA}}^{i,\text{enter}} &= \underline{t}_{\text{CPA}}^{\text{enter}} \\ \bar{t}_{\text{CPA}}^{i,\text{enter}} &= \bar{t}_{\text{CPA}}^{\text{enter}}, \end{aligned} \tag{14}$$

and

$$\begin{aligned} \underline{d}_{\text{CPA}}^{i,\text{exit}} &= \underline{d}_{\text{CPA}}^{\text{exit}} \\ \underline{t}_{\text{CPA}}^{i,\text{exit}} &= \underline{t}_{\text{CPA}}^{\text{exit}} \\ \bar{t}_{\text{CPA}}^{i,\text{exit}} &= \bar{t}_{\text{CPA}}^{\text{exit}} \end{aligned} \tag{15}$$

for all $i \in$ {SO,OT,GW,HO}.

4.2.3. Geometrical Situation Interpretation

Tam and Bucknall (2010) present a geometrical interpretation scheme for deciding COLREGs situations based on the relative position, bearing and course of the obstacle with respect to the ownship. We base our geometrical interpretation on a slightly modified version of this scheme, where we include the sign of t_{CPA} to distinguish between situations where the obstacle moves closer toward or farther away from the ownship. The geometrical interpretation is shown in **Figure 5**, where the geometrical situation is obtained by finding which region the obstacle position and course resides in.

Notice that the head-on region is larger than the threshold of $\pm 6°$ as described by the COLREGs. The reason for this is that

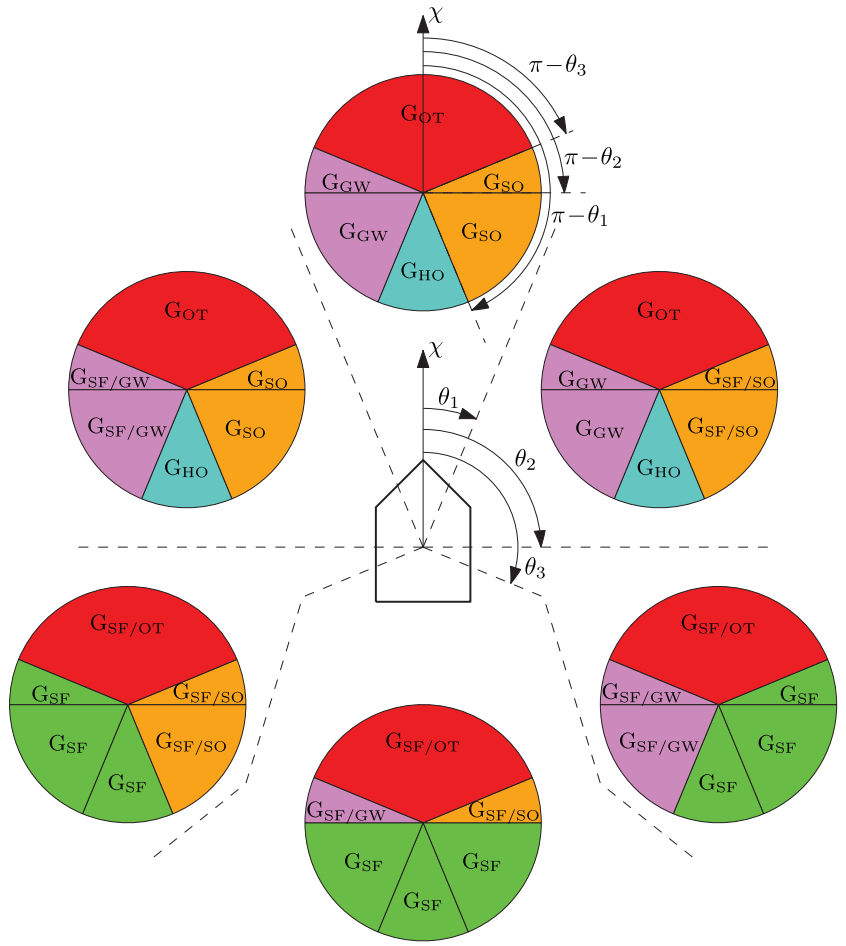


FIGURE 5 | Illustration of the geometrical COLREGs interpretation, where the ownship course is denoted as χ and $\theta_1, \theta_2, \theta_3$ denote symmetrical regions given as [22.5, 90, 112.5°] offset from ahead. The circles illustrate obstacles in different relative bearing regions, and have a fixed orientation with respect to the ownship. The geometrical situations are color-coded and denoted as $G_i, i \in \{SF, SO, OT, GW, HO\}$ for safe, stand-on, overtaking, give-way and head-on situations, respectively. When two situations are given, like e.g., $G_{SF/SO}$, we use the former (SF) if $t_{CPA} < 0$ and the latter (SO) if $t_{CPA} \geq 0$, analogous to the obstacle moving away or toward the ownship. To decide the geometrical situation, we first find which relative bearing region the obstacle resides in, before finding which obstacle region the obstacle's course resides in. The figure is inspired by Tam and Bucknall (2010).

Tam and Bucknall recommend using a larger region of 22.5° in order to increase the robustness of the geometrical COLREGs interpretation scheme.

4.3. Interface to the High-Level Planner

The high-level planner produces an energy-optimized nominal trajectory for the ownship to follow. However, since the high-level planner does not consider moving obstacles, the speed is the only time-relevant factor of the desired trajectory. In a case where the ownship for some reason, e.g., avoiding moving obstacles, lag behind the nominal trajectory, following the nominal trajectory in absolute time would cause a speed increase in order to catch up with it. Therefore, the mid-level algorithm performs *relative trajectory tracking*, where it tracks the nominal trajectory with a time offset $t_b \in \mathbb{R}$. This results in a relative nominal trajectory for the mid-level algorithm:

$$\bar{\boldsymbol{p}}_d(t) = \boldsymbol{p}_d(t + t_b), \tag{16}$$

where $\boldsymbol{p}_d = [N_d(t), E_d(t)]^\top$ is the nominal trajectory from the high-level planner. The time offset t_b is calculated each time the mid-level algorithm is run by solving a separate optimization problem, and is selected such that $\bar{\boldsymbol{p}}_d(t_0)$ is the point on the nominal trajectory closest to the ownship. See Bitar et al. (2019a) for a detailed description of this concept.

4.4. Optimization Problem Formulation

The mid-level algorithm is formalized as an OCP:

$$\min_{\boldsymbol{\eta}(\cdot), \boldsymbol{x}_r(\cdot)} \quad \phi(\boldsymbol{\eta}(\cdot), \boldsymbol{x}_r(\cdot)) \tag{17a}$$

subject to

$$\dot{\boldsymbol{\eta}}(t) = \boldsymbol{R}(\psi(t))\boldsymbol{x}_r(t) + \begin{bmatrix} \boldsymbol{V}_c \\ 0 \end{bmatrix} \forall t \in [t_0, t_0 + T_h] \tag{17b}$$

$$\boldsymbol{h}_{\text{mid}}(\boldsymbol{\eta}(t), \boldsymbol{x}_r(t), t) \leq \boldsymbol{0} \; \forall t \in [t_0, t_0 + T_h] \tag{17c}$$

$$\boldsymbol{e}_{\text{mid}}(\boldsymbol{\eta}(t_0)) = \boldsymbol{0}, \tag{17d}$$

where $T_h > 0$ is the prediction horizon, $\phi(\cdot, \cdot)$ is the objective functional, (17b) contains a kinematic vessel model, (17c) contains inequality constraints and (17d) contains boundary constraints.

Analytical solutions of OCPs are in general not possible to find. A more common approach is to transcribe the OCP to an NLP, and solve that using a gradient optimization scheme. In our case, we transcribe (17) into an NLP with N_p samples using multiple shooting, where the vessel model is discretized using 4th order Runge Kutta and the cost functional is discretized using forward Euler. The resulting NLP is given as

$$\begin{aligned} &\min_{\boldsymbol{w},\boldsymbol{\omega},\boldsymbol{\mu},\boldsymbol{\xi}} \; \phi_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}) + \phi_c(\boldsymbol{w}) + \phi_{\text{COLREGs}}(\boldsymbol{w}) + \phi_{\xi}(\boldsymbol{\xi}) \\ &\text{subject to} \\ &\boldsymbol{g}(\boldsymbol{w}, \boldsymbol{\eta}(t_0)) = \boldsymbol{0} \\ &\boldsymbol{h}(\boldsymbol{w}, \boldsymbol{\xi}) \leq \boldsymbol{0} \\ &\bar{\boldsymbol{h}}_k(\boldsymbol{\eta}_k, \boldsymbol{\omega}_k, \boldsymbol{\mu}_k, \bar{\boldsymbol{p}}_{d,k}) \leq \boldsymbol{0} \; \forall k \in \{1, \ldots, N_p\} \\ &\boldsymbol{\xi} \geq \boldsymbol{0}, \end{aligned} \tag{18}$$

where $\boldsymbol{w} = [\boldsymbol{\eta}_0^\top, \boldsymbol{x}_{r,0}^\top, \ldots, \boldsymbol{\eta}_{N_p-1}^\top, \boldsymbol{x}_{r,N_p-1}^\top, \boldsymbol{\eta}_{N_p}^\top]^\top \in \mathbb{R}^{5N_p+3}$ is a vector of $5N_p + 3$ decision variables and $\bar{\boldsymbol{p}}_{d,1:N_p} = [\bar{\boldsymbol{p}}_{d,1}, \bar{\boldsymbol{p}}_{d,2}, \ldots, \bar{\boldsymbol{p}}_{d,N_p}]$ is a sequence of desired positions. The vectors $\boldsymbol{\omega} \in \mathbb{R}^{2N_p}$, $\boldsymbol{\mu} \in \mathbb{R}^{2N_p}$ and $\boldsymbol{\xi} \in \mathbb{R}^{MN_p}$ contain slack variables, where M is the number of moving obstacles to be included in the constraints.

The vector $\boldsymbol{g}(\boldsymbol{w}, \boldsymbol{\eta}(t_0)) \in \mathbb{R}^{3N_p+3}$ contains shooting and boundary constraints, while $\boldsymbol{h}(\boldsymbol{w}) \in \mathbb{R}^{(M+D+4)N_p}$, where D is the number of static obstacles, contain inequality constraints ensuring COLAV and steady-state vessel velocity feasibility. The vectors $\bar{\boldsymbol{h}}_k(\boldsymbol{\eta}_k, \boldsymbol{\omega}_k, \boldsymbol{\mu}_k, \bar{\boldsymbol{p}}_{d,k}) \in \mathbb{R}^6$, $k \in \{1, N_p\}$ contain constraints on the slack variables $\boldsymbol{\omega}$ and $\boldsymbol{\mu}$.

In the following subsections, we describe the terms in (18) in more detail.

4.4.1. Objective Function

The objective function contains four functions, where $\phi_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu})$ introduces cost on deviating from the relative nominal trajectory $\bar{\boldsymbol{p}}_d(t)$, $\phi_c(\boldsymbol{w})$ introduces cost on using control input, $\phi_{\text{COLREGs}}(\boldsymbol{w})$ is a COLREGs-specific function and $\phi_{\xi}(\boldsymbol{\xi})$ introduces slack variable cost.

To avoid that the NLP changes behavior when moving away from the nominal trajectory, we wish to have linear growth in the position error function $\phi_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu})$. This is achieved by instead of using quadratic terms in the position error function, we use the Huber loss function which is quadratic around the origin and resembles the absolute value function above a given threshold $\sigma > 0$:

$$H(\rho) = \begin{cases} \frac{1}{2}\rho^2 & |\rho| \leq \sigma \\ \sigma(|\rho| - \frac{1}{2}\sigma) & |\rho| > \sigma \, . \end{cases} \tag{19}$$

The Huber loss function has a discontinuous gradient, making it slightly complicated to implement in gradient-based optimization problems. It can, however, be implemented in a continuous fashion by utilizing lifting, where slack variables are introduced to create a problem of a higher dimensionality which is easier to solve. Using this technique, $\bar{\phi}_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu})$ is defined as

$$\bar{\phi}_p(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}) = K_p \sum_{k=1}^{N_p} \sigma \boldsymbol{1}^\top \boldsymbol{\omega}_k + \frac{1}{2}\boldsymbol{\mu}_k^\top \boldsymbol{\mu}_k, \tag{20}$$

where $K_p > 0$ is a tuning parameter, and $\boldsymbol{\omega}_k \in \mathbb{R}^2$ and $\boldsymbol{\mu}_k \in \mathbb{R}^2$ are slack variables constrained by

$$\bar{\boldsymbol{h}}_k(\boldsymbol{w}, \boldsymbol{\omega}, \boldsymbol{\mu}, \bar{\boldsymbol{p}}_{d,k}) = \begin{bmatrix} \boldsymbol{v_k} + \boldsymbol{\mu_k} + \boldsymbol{p}_k - \bar{\boldsymbol{p}}_{d,k} \\ \boldsymbol{v_k} + \boldsymbol{\mu_k} - (\boldsymbol{p}_k - \bar{\boldsymbol{p}}_{d,k}) \\ -\boldsymbol{\omega}_k \end{bmatrix} \leq \boldsymbol{0} \quad \forall k \in \{1, \ldots, N_p\}, \tag{21}$$

where $\boldsymbol{p}_k$ is the predicted vessel position at time step k, i.e., $\boldsymbol{\eta}_k = [\boldsymbol{p}_k^\top, \psi_k]^\top$. See Bitar et al. (2019a) for more details.

Rule 8 of the COLREGs requires that maneuvers are readily observable for other vessels, implying that speed and course changes should have a sufficiently large magnitude, and not be performed as a sequence of small changes. In order to enforce this in the optimization problem, the control cost function $\phi_c(\boldsymbol{w})$ introduces a non-linear cost on the change in speed and course, which makes the algorithm favor readily observable maneuvers. The function is defined as

$$\phi_c(\boldsymbol{w}) = \sum_{k=0}^{N_p-1} K_{\dot{U}} q_{\dot{U}}(\dot{U}_k) + K_{\dot{\chi}} q_{\dot{\chi}}(\dot{\chi}_k), \tag{22}$$

where $K_{\dot{U}}, K_{\dot{\chi}} > 0$ are tuning parameters, while $q_{\dot{U}}(\dot{U}_k)$ and $q_{\dot{\chi}}(\dot{\chi}_k)$ are the non-linear cost functions. Notice that neither the speed over ground (SOG) U nor the course χ are elements of the search space, but they can be computed as $U = \sqrt{u^2 + v^2}$ and $\chi = \psi + \arcsin\frac{v}{U}$. Their derivatives are then calculated by finite differencing. See Eriksen and Breivik (2017a) and Bitar et al. (2019a) for more details on the control cost function.

The $\phi_{\text{COLREGs}}(\boldsymbol{w})$ function introduces a COLREGs-specific cost with respect to obstacles based on the rule currently applicable as defined by the state machine. We hence tailor the NLP to the current situation. The function is defined as

$$\phi_{\text{COLREGs}}(\boldsymbol{w}) = \sum_{k=1}^{N_p} \Bigg[\sum_{i \in \mathcal{O}_{\text{HO}}} K_{\text{HO}} V_{\text{HO},i,k}(\boldsymbol{p}_k) + \sum_{i \in \mathcal{O}_{\text{GW}}} K_{\text{GW}} V_{\text{GW},i,k}(\boldsymbol{p}_k) + \sum_{i \in \mathcal{O}_{\text{SO}}} K_{\text{SO}} V_{\text{SO},k}(\boldsymbol{w}) + \sum_{i \in \mathcal{O}_{\text{EM}}} K_{\text{EM}} V_{\text{EM},k}(\boldsymbol{w}) \Bigg], \tag{23}$$

where $\mathcal{O}_{\text{HO}}, \mathcal{O}_{\text{GW}}, \mathcal{O}_{\text{SO}}$, and $\mathcal{O}_{\text{EM}}$ contain obstacles which are in the head-on, give-way, stand-on and emergency states, respectively, and $K_{\text{HO}}, K_{\text{GW}}, K_{\text{SO}}, K_{\text{EM}} > 0$ are tuning parameters. The functions $V_{\text{HO},i,k}(\boldsymbol{p}_k)$, $V_{\text{GW},i,k}(\boldsymbol{p}_k)$, $V_{\text{SO},k}(\boldsymbol{w})$, and $V_{\text{EM},k}(\boldsymbol{w})$ describe functions capturing head-on, give-way, stand-on and emergency behavior with respect to obstacle i, respectively. Notice that the head-on and give-way functions vary with both the obstacle number and time step number, which is due to the functions depending on the given obstacles position and course at time step k.

For head-on situations, we define a potential function with a positive value on the obstacle's starboard side, and a negative value on its port side. When used in the objective function, this will favor trajectories passing a head-on obstacle on its port side, in compliance with Rule 14 of the COLREGs. In addition, the potential function has an attenuation term, reducing the impact of the function when far away from an obstacle:

$$V_{\text{HO},i,k}(\boldsymbol{p}) = \frac{\tanh\left(\alpha_{x,\text{HO}}(x_{0,\text{HO}} - x^{\{i,k\}})\right)}{2} \tanh(\alpha_{y,\text{HO}} y^{\{i,k\}}) \in (-1, 1), \tag{24}$$

where $\alpha_{x,\text{HO}}, \alpha_{y,\text{HO}} > 0$ are tuning parameters controlling the steepness of the head-on potential function and $\bar{x}_{0,\text{HO}} > 0$ is a tuning parameter controlling the influence of the attenuating potential. The coordinate $(x^{\{i,k\}}, y^{\{i,k\}})$ is $\boldsymbol{p}$ given in obstacles i's course-fixed frame (in which the x-axis points along the obstacle's course) at time step k, computed as

$$\begin{bmatrix} x^{\{i,k\}} \\ y^{\{i,k\}} \end{bmatrix} = \boldsymbol{R}(\chi_{i,k})^{\top} \left(\boldsymbol{p} - \boldsymbol{p}_{o,k,i}\right), \tag{25}$$

where $\boldsymbol{p}_{o,k,i}$ and $\chi_{i,k}$ are the position and course of obstacle i at time step k. The head-on potential function with parameters $\alpha_{x,\text{HO}} = 1/500, \alpha_{y,\text{HO}} = 1/400$ and $x_{0,\text{HO}} = 1,000$ m is shown in **Figure 6A**.

For give-way situations, we define a similar potential function, but rotated such that the function is positive in front of an obstacle and negative behind it. This will favor trajectories passing behind an obstacle, as desirable with respect to Rule 15 when a give-way obligation is active. The give-way potential function is defined as

$$V_{\text{GW},i,k}(\boldsymbol{p}) = \frac{\tanh\left(\alpha_{y,\text{GW}}(y^{\{i,k\}} - y_{0,\text{GW}})\right)}{2} \tanh(\alpha_{x,\text{GW}} x^{\{i,k\}}) \in (-1, 1), \tag{26}$$

where $\alpha_{x,\text{GW}}, \alpha_{y,\text{GW}} > 0$ control the steepness of the give-way potential function and $\bar{y}_{0,\text{GW}} < 0$ control the attenuation on the port side of an obstacle. The give-way potential function with parameters $\alpha_{x,\text{GW}} = 1/400, \alpha_{y,\text{GW}} = 1/500$ and $y_{0,\text{GW}} = -500$ m is shown in **Figure 6B**.

In stand-on situations, we want the mid-level algorithm to disregard the obstacle and keep the current speed and course

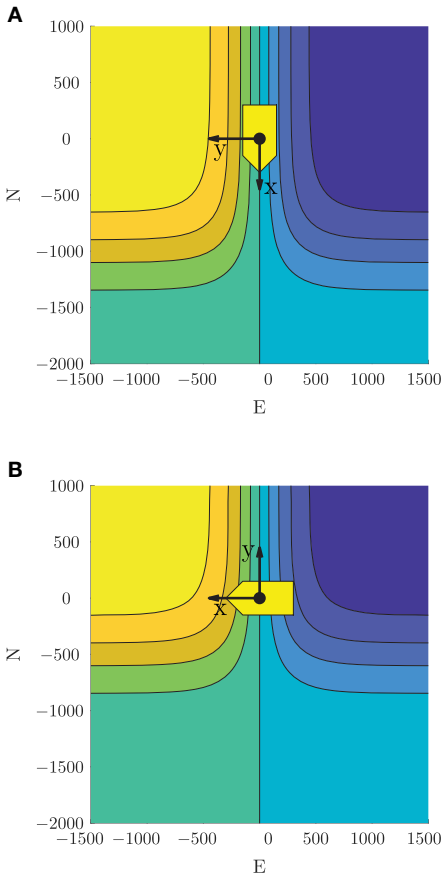


FIGURE 6 | Potential functions ensuring passing on the correct side in head-on and give-way situations. Yellow indicates a positive value, blue indicates a negative value, while the yellow patch and axis cross show the obstacle location and course-fixed coordinate system. Used in a minimization scheme, this will favor starboard maneuvers in head-on situations, and passing behind obstacles in give-way situations. Note that the obstacle here has zero sideslip, resulting in the heading and course pointing in the same direction. **(A)** Head-on potential function. **(B)** Give-way potential function.

in order to comply with the first part of Rule 17. One could simply constrain the algorithm to not maneuver, but this would be perilous in situations where the ownship simultaneously finds itself in a head-on or give-way situation. In such a situation it would be of extra importance to choose readily observable maneuvers, and we therefore design the stand-on cost with the same terms as used in the control cost (22) to amplify the effect:

$$V_{\text{SO},k}(\boldsymbol{w}) = K_{\dot{U}} q_{\dot{U}}(\dot{U}_k) + K_{\dot{\chi}} q_{\dot{\chi}}(\dot{\chi}_k). \tag{27}$$

If an obstacle is in an emergency state, the obstacle is disregarded in the mid-level algorithm and left for the short-term algorithm to handle. In such a situation, it is important that the mid-level algorithm behaves predictable, and we therefore use the same cost function as for stand-on situations:

$$V_{\text{EM},k}(\boldsymbol{w}) = V_{\text{SO},k}(\boldsymbol{w}). \tag{28}$$

The slack variable $\boldsymbol{\xi}$ is used in a homotopy scheme, which we introduce to avoid getting trapped in local minima around moving obstacles. The homotopy scheme is described in further detail in section 4.5. The homotopy cost function $\phi_{\xi}(\boldsymbol{\xi})$ introduces slack cost on $\boldsymbol{\xi}$:

$$\phi_{\xi}(\boldsymbol{\xi}) = K_{\xi}\mathbf{1}^{\top}\boldsymbol{\xi}, \tag{29}$$

where $K_{\xi} > 0$ is iteratively increased as part of the homotopy scheme.

4.5. Obstacle Handling and Steady-State Feasibility

The inequality constraint $\boldsymbol{h}(\boldsymbol{w}, \boldsymbol{\xi}) \leq \mathbf{0}$ ensures COLAV and steady-state feasibility with respect to actuator limitations.

Static obstacles are handled similarly as in the high-level algorithm, with (4) representing an elliptical obstacle with center (x_c, y_c), angle α and major and minor axes x_a and y_a, respectively. The constraint (4) needs to be enforced at each time step. Hence, for the i-th static obstacle, we define the constraint

$$\boldsymbol{h}_{s_i}(\boldsymbol{w}) = \begin{bmatrix} h_o(x_1, y_1, x_{c,i}, y_{c,i}, x_{a,i}, y_{a,i}, \alpha_i) \\ h_o(x_2, y_2, x_{c,i}, y_{c,i}, x_{a,i}, y_{a,i}, \alpha_i) \\ \vdots \\ h_o(x_{N_p}, y_{N_p}, x_{c,i}, y_{c,i}, x_{a,i}, y_{a,i}, \alpha_i) \end{bmatrix} \leq \mathbf{0}. \tag{30}$$

Moving obstacles are handled in a similar fashion, but letting the ellipsis center position and angle be time varying. Obstacles in stand-on situations should, however, not be included in the constraints, since the mid-level algorithm is supposed to stand on in such situations. Moreover, if an obstacle has entered an emergency state, the obstacle is so close and behaving unpredictably that the mid-level algorithm should disregard it and leave it for the short-term layer. Hence, for the i-th moving obstacle not in a stand-on or an emergency situation, we define the constraint

$$\boldsymbol{h}_{m_i}(\boldsymbol{w}) = \begin{bmatrix} h_o(x_1, y_1, x_{c,i,1}, y_{c,i,1}, x_{a,i}, y_{a,i}, \alpha_{i,1}) \\ \vdots \\ h_o(x_{N_p}, y_{N_p}, x_{c,i,N_p}, y_{c,i,N_p}, x_{a,i}, y_{a,i}, \alpha_{i,N_p}) \end{bmatrix} \leq \mathbf{0}, \tag{31}$$

where $x_{c,i,k}, y_{c,i,k}$, and $\alpha_{i,k}$ denote the position and course of the i-th moving obstacle at time step k.

Given D static obstacles and M obstacles not in stand-on or emergency situations, we define the constraint

$$\boldsymbol{h}_o(\boldsymbol{w}, \boldsymbol{\xi}) = \begin{bmatrix} \boldsymbol{h}_{s_1}(\boldsymbol{w}) \\ \vdots \\ \boldsymbol{h}_{s_D}(\boldsymbol{w}) \\ \boldsymbol{h}_{m_1}(\boldsymbol{w}) \\ \vdots \\ \boldsymbol{h}_{m_M}(\boldsymbol{w}) \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\xi} \end{bmatrix}, \tag{32}$$

where we include slack variables $\boldsymbol{\xi} \geq \mathbf{0}$ on the moving obstacle constraints as part of a homotopy scheme. The reason for using homotopy is that NLP solvers in general only finds local minima, and can have issues with moving an initial guess "through" obstacles. Normally, this is not an issue, but for the mid-level algorithm the optimal solution can change drastically from one iteration to another. This can for instance happen if an obstacle enters a head-on or give-way state, where the solution can be trapped on the wrong side of an obstacle. In general, homotopy describes introducing an extra parameter which is iteratively adjusted in order to iteratively move a local solution toward a global solution (Deuflhard, 2011). In our homotopy scheme, we introduce slack variables on the moving obstacle constraints, which will allow solutions to travel through obstacles at the cost of a homotopy cost (29) scaled by the homotopy parameter K_{ξ}. Initially, this is selected as a low value to have a high amount of slack on the moving obstacles, while it is iteratively increased toward $K_{\xi} \to \infty$, which results in $\boldsymbol{\xi} = \mathbf{0}$ and hence no slack on moving obstacles. Currently, we only introduce slack on moving obstacles, but slack should also be introduced to static obstacles if they are small enough for the algorithm to be able to pass on both sides, like e.g., rocks, navigational marks, etc.

Similarly as in Eriksen and Breivik (2017b) and Bitar et al. (2019a), we ensure steady-state feasible trajectories at each time step through a constraint $\boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,k}) \leq \mathbf{0} \in \mathbb{R}^4$, which captures the state constraint $\boldsymbol{x}_r \in \mathcal{X}_r$ at time step k. To ensure steady-state feasibility for the entire prediction horizon, we define the constraint

$$\boldsymbol{h}_{\boldsymbol{x}_r}(\boldsymbol{w}) = \begin{bmatrix} \boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,0}) \\ \boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,1}) \\ \vdots \\ \boldsymbol{h}_{\boldsymbol{x}_{r,k}}(\boldsymbol{x}_{r,N_p-1}) \end{bmatrix} \leq \mathbf{0}. \tag{33}$$

Finally, the inequality constraints are combined as.

$$\boldsymbol{h}(\boldsymbol{w}, \boldsymbol{\xi}) = \begin{bmatrix} \boldsymbol{h}_o(\boldsymbol{w}, \boldsymbol{\xi}) \\ \boldsymbol{h}_{\boldsymbol{x}_r}(\boldsymbol{w}) \end{bmatrix} \in \mathbb{R}^{(M+D+4)N_p}. \tag{34}$$

5. SHORT-TERM COLAV

For the short-term layer, the branching-course model predictive control (BC-MPC) algorithm is used, which is a sample-based MPC algorithm intended for short-term ASV COLAV. The BC-MPC algorithm was initially developed in Eriksen et al. (2019), extended to also consider static obstacles in Eriksen and Breivik (2019) and is experimentally validated in several full-scale experiments using a radar-based system for detecting and tracking obstacles. The algorithm complies with COLREGs rules 8, 13, and the second part of Rule 17, while favoring maneuvers complying with the maneuvering aspects of rules 14 and 15. Notice that Rule 17 allows a ship to ignore the maneuvering aspects of rules 14 and 15 in situations where the give-way vessel does not maneuver. The obstacle clearance will be larger if the algorithm ignores the maneuvering aspects of rules 14 and 15, like e.g., passing in front of an obstacle in a crossing situation where the ownship is the give-way vessel. Moving obstacles are in general handled by the mid-level algorithm, making this applicable only in emergency situations and for

obstacles detected so late that the mid-level algorithm is unable to avoid them.

The algorithm constructs a search space consisting of a finite number of trajectories, which each contain a sequence of maneuvers. The maneuvers are constructed using a dynamic model of the ownship and a set of acceleration motion primitives, resulting in feasible trajectories being specified to the vessel controller. For each maneuver, a discrete set of SOG and course accelerations are created as

$$\begin{aligned} \dot{\boldsymbol{U}}_{\text{samples}} &= \left\{\dot{U}_1, \dot{U}_2, \ldots, \dot{U}_{N_U}\right\} \\ \ddot{\boldsymbol{\chi}}_{\text{samples}} &= \left\{\ddot{\chi}_1, \ddot{\chi}_2, \ldots, \ddot{\chi}_{N_\chi}\right\}, \end{aligned} \tag{35}$$

where $\dot{U}_i, i \in [1, N_U]$ and $\ddot{\chi}_i, i \in [1, N_\chi]$ denote $N_U \in \mathbb{N}$ and $N_\chi \in \mathbb{N}$ vessel-feasible speed and course accelerations. Given the acceleration samples (35) and motion primitives for each maneuver in a trajectory, we create a set of desired SOG and course trajectories $\mathcal{U}_d$. These trajectories have continuous acceleration, and is designed in an open-loop fashion by using the current reference tracked by the vessel controller for initialization, rather than the current vessel SOG and course. The reason for this is that the reference to the vessel controller should be continuous in order to avoid jumps in the actuator commands. To include feedback in the trajectory prediction, a set of feedback-corrected SOG and course trajectories $\bar{\mathcal{U}}_d$ is predicted using a simplified error model of the vessel and vessel controller. Finally, the feedback-corrected SOG and course trajectories are used to compute a set of feedback-corrected pose trajectories:

$$\bar{\mathcal{H}} = \left\{\bar{\boldsymbol{\eta}}(\cdot) \middle| (\bar{U}(\cdot), \bar{\chi}(\cdot)) \in \bar{\mathcal{U}}\right\}, \tag{36}$$

where $\bar{\boldsymbol{\eta}}(\cdot)$ denotes a kinematic simulation procedure that given SOG and course trajectories, $\bar{U}(\cdot)$ and $\bar{\chi}(\cdot)$, in $\bar{\mathcal{U}}_d$ computes the vessel pose. See Eriksen and Breivik (2019) and Eriksen et al. (2019) for more details on the trajectory generation procedure.

In order to converge toward the trajectory specified by the mid-level algorithm, a desired acceleration is computed based on a line-of-sight guidance scheme. In Eriksen and Breivik (2019) and Eriksen et al. (2019), the samples closest to the desired acceleration in (35) are replaced with the desired acceleration, given that this is vessel-feasible. A problem with this, is that when operating at high speeds, the possible acceleration may not be symmetric, resulting in that zero acceleration (hence keeping a constant speed and course), may not be part of the search space. This can cause undesirable behavior, since the BC-MPC algorithm will be unable to keep the speed and course constant, which can cause oscillatory behavior. In this paper, we therefore propose to move the acceleration samples closest to zero, and adding the desired acceleration as a separate sample, given that it is vessel feasible. This will make sure that keeping a constant speed and course, as well as a trajectory converging toward the desired trajectory is included in the search space.

Given the predicted trajectories, the algorithm finds the optimal desired SOG and course trajectory for the vessel controller $\boldsymbol{u}_d^*(\cdot) = [U_d(\cdot)^*, \chi_d(\cdot)^*]$ as

$$\boldsymbol{u}_d^*(\cdot) = \underset{(\bar{\boldsymbol{\eta}}_k(\cdot), \boldsymbol{u}_{d,k}(\cdot)) \in (\bar{\mathcal{H}}, \mathcal{U}_d)}{\operatorname{argmin}} G(\bar{\boldsymbol{\eta}}_k(\cdot), \boldsymbol{u}_{d,k}(\cdot); \boldsymbol{p}_d^{\text{mid}}(\cdot)), \tag{37}$$

where the objective function is given as

$$\begin{aligned} G(\bar{\boldsymbol{\eta}}(\cdot), \boldsymbol{u}_d(\cdot); \boldsymbol{p}_d^{\text{mid}}(\cdot)) \;=\; & w_{\text{al}} \text{align}(\bar{\boldsymbol{\eta}}(\cdot); \boldsymbol{p}_d^{\text{mid}}(\cdot)) \\ & + w_{\text{av,m}} \text{avoid}_{\text{m}}(\bar{\boldsymbol{\eta}}(\cdot)) + w_{\text{av,s}} \text{avoid}_{\text{s}}(\bar{\boldsymbol{\eta}}(\cdot)) \\ & + w_{\text{t},U} \text{tran}_U(\boldsymbol{u}_d(\cdot)) + w_{\text{t},\chi} \text{tran}_\chi(\boldsymbol{u}_d(\cdot)). \end{aligned} \tag{38}$$

The variables $w_{\text{al}}, w_{\text{av,m}}, w_{\text{av,s}}, w_{\text{t},U}, w_{\text{t},\chi} > 0$ are tuning parameters, while $\text{align}(\bar{\boldsymbol{\eta}}(\cdot); \boldsymbol{p}_d^{\text{mid}}(\cdot))$ measures the alignment between a candidate trajectory $\bar{\boldsymbol{\eta}}(\cdot)$ and the desired trajectory from the mid-level algorithm $\boldsymbol{p}_d^{\text{mid}}(\cdot)$. The function $\text{avoid}_{\text{m}}(\bar{\boldsymbol{\eta}}(\cdot))$ ensures COLAV of moving obstacles by penalizing trajectories close to obstacles, using a non-symmetric obstacle ship domain designed with the COLREGs in mind. The function $\text{avoid}_{\text{s}}(\bar{\boldsymbol{\eta}}(\cdot))$ ensures COLAV of static obstacles by introducing an occupancy grid, while $\text{tran}_U(\boldsymbol{u}_d(\cdot))$ and $\text{tran}_\chi(\boldsymbol{u}_d(\cdot))$ introduces transitional costs to avoid shattering. The transitional terms penalize deviations from the planned trajectory of the previous iteration, unless changing to the trajectory corresponding by the desired acceleration. See Eriksen and Breivik (2019) and Eriksen et al. (2019) for more details and descriptions of the terms.

6. SIMULATION RESULTS

The hybrid COLAV system is verified through simulations, which are present in this section. The simulations include ocean current and both static and moving obstacles. We include moving obstacles both acting in compliance with the COLREGs, and violating the COLREGs.

6.1. Simulation Setup

The simulations are performed in MATLAB on a computer with an 2.8 GHz Intel Core i7 processor running macOS Mojave, using CasADi (Andersson et al., 2019) and IPOPT (Wächter and Biegler, 2005) for implementing the high-level and mid-level algorithms. The simulator is built upon the mathematical model of the Telemetron ASV described in section 2, and the model-based speed and course controller in Eriksen and Breivik (2018) is used as the vessel controller.

The parameters of the high-level algorithm are listed in **Table 1**. The number of prediction steps N_{hi} is chosen to achieve a time step length $h = {}^{t_{\max}}/_{N_{\text{hi}}} < 1.5$ s, which seems to be a good compromise between capturing the relevant system dynamics and having a feasible computational requirement.

The mid-level algorithm is implemented using the parameters in **Table 2**.

The slack variable cost K_ξ has five elements, implying that we use five steps in our homotopy scheme. The mid-level NLP is initially warm started with the solution from the previous iteration, while each step in the homotopy scheme is warm started with the solution from the previous step of the homotopy scheme, converging toward the solution without slack on the constraints. To reduce the computational load and increase the predictability of the mid-level algorithm, we utilize six steps of each planned mid-level trajectory, only running the mid-level algorithm every 60 s. This implies that six steps of the predicted

TABLE 1 | Tuning parameters for the high-level algorithm.

Param.	Value	Comment
$t_{\max}$		Maximum trajectory time
Scenario 1	1420 s	
Scenario 2	1420 s	
Scenario 3	725 s	
N_{hi}	1000	Number of prediction steps
K_e	1.0 s^3/m	Energy penalty gain
K_δ	1.0	Quadratic yaw control penalty gain
L_m	4.0 m	Length between control origin and outboard motor

TABLE 2 | Tuning parameters for the mid-level algorithm.

Param.	Value	Comment
$\bar{d}_{\text{CPA}}^{\text{enter}}$	900 m	State machine d_{CPA} entry criteria
$\underline{d}_{\text{CPA}}^{\text{exit}}$	2000 m	State machine d_{CPA} exit criteria
$[\underline{t}_{\text{CPA}}^{\text{enter}}, \bar{t}_{\text{CPA}}^{\text{enter}}]$	[0, 270] s	State machine t_{CPA} entry criteria
$[\underline{t}_{\text{CPA}}^{\text{exit}}, \bar{t}_{\text{CPA}}^{\text{exit}}]$	[-20, 290] s	State machine t_{CPA} exit criteria
$\bar{t}_{\text{crit}}^{\text{EM,enter}}$	20 s	Emergency state t_{crit} entry criteria
$\underline{t}_{\text{crit}}^{\text{EM,exit}}$	25 s	Emergency state t_{crit} exit criteria
h	10 s	Step size
N_p	36	Number of prediction steps
K_p	0.02	Position error scaling
σ	1	Huber loss function threshold
$K_{\dot{U}}$	0.3	SOG-derivative penalty term scaling
$K_{\dot{\chi}}$	2.5	Course-derivative penalty term scaling
K_{HO}	40	Head-on potential function scaling
$[\alpha_{x,\text{HO}}, \alpha_{y,\text{HO}}]$	[1/500, 1/400]	Head-on potential function steepness parameters
$x_{0,\text{HO}}$	1000 m	Head-on potential function attenuation parameter
K_{GW}	40	Give-way potential function scaling
$[\alpha_{x,\text{GW}}, \alpha_{y,\text{GW}}]$	[1/400, 1/500]	Give-way potential function steepness parameters
$y_{0,\text{GW}}$	−500 m	Give-way potential function attenuation parameter
K_{SO}	3	Stand-on function scaling
K_{EM}	3	Emergency function scaling
K_ξ	[0.1, 1, 10, 100, ∞]	Iterative slack variable cost
x_a	600 m	Moving obstacle ellipsis major axis size
y_a	225 m	Moving obstacle ellipsis minor axis size

solution will be implemented before computing a new solution, which further implies that the state machine is also only run every 60 s. If the mid-level algorithm fails in finding a feasible solution, the algorithm will re-use the solution from the last iteration. This may for instance happen if the algorithm tries to compute a solution while being inside a moving obstacle ellipse, which sometimes can be the case when an obstacle is exiting an emergency or stand-on state. The BC-MPC algorithm is run every 5 s, with parameters as described in Eriksen and Breivik (2019). An update rate of 5 s is considered sufficient due the typically large maneuvering margins at sea. It is also worth noting that the detection and tracking system can represent a significant time delay, especially for radar-based systems (Eriksen et al., 2019). For confined and congested areas the BC-MPC algorithm may need to be run at a higher rate, which also imposes requirements for high-bandwidth obstacle estimates. Static obstacles are padded with a safety margin of 150 m for the high-level and mid-level algorithms, while the BC-MPC algorithm uses a safety margin of 100 m for static obstacles. The reason for having a smaller static obstacle safety margin for the BC-MPC algorithm is that it tends to struggle with following trajectories on the static obstacle boundaries. The BC-MPC algorithm would hence not be able to follow the nominal trajectory if the static obstacle safety margin was the same as for the mid-level and high-level algorithms.

The simulations are performed without any noise on the obstacle estimates, providing the algorithms with exact information about the obstacles position, course, and speed. The BC-MPC algorithm has previously been shown to perform well with noisy and uncertain obstacle estimates in full-scale experiments using radar-based detection and tracking of obstacles (Eriksen and Breivik, 2019; Eriksen et al., 2019). The mid-level algorithm is likely to have a larger requirement to low noise levels on the obstacle estimates, since the state machine in the mid-level algorithm depends on logic and discrete switching. However, the algorithm is also run less frequently, reducing the required bandwidth of the obstacle estimates, possibly allowing using smoothing or tracking filters with a lower process noise if necessary. It may also be feasible to make the mid-level algorithm depend on data from the automatic identification system, which typically have much lower noise levels than radar-based tracking systems, while being subject to robustness issues (Harati-Mokhtari et al., 2007).

We present three scenarios, which demonstrate different important properties of the hybrid COLAV system:

Scenario 1 This scenario contains two static obstacles, and four moving obstacles of which all comply with the COLREGs. The moving obstacles demonstrate stand-on, give-way and head-on situations.

Scenario 2 This scenario contains one static and five moving obstacles. The moving obstacles demonstrate stand-on with an obstacle ignoring the COLREGs, an overtaking and a simultaneous head-on, give-way and stand-on situation with obstacles complying with the COLREGs.

Scenario 3 This scenario contains two moving obstacles, which suddenly perform dangerous maneuvers close to the ownship, displaying the use of the emergency state.

6.2. Scenario 1

Scenario 1 contains two static obstacles, four moving obstacles, an ocean current of $[-2, 0]^\top$ m/s and is shown in **Figure 7**. The high-level planner plans a nominal trajectory between the

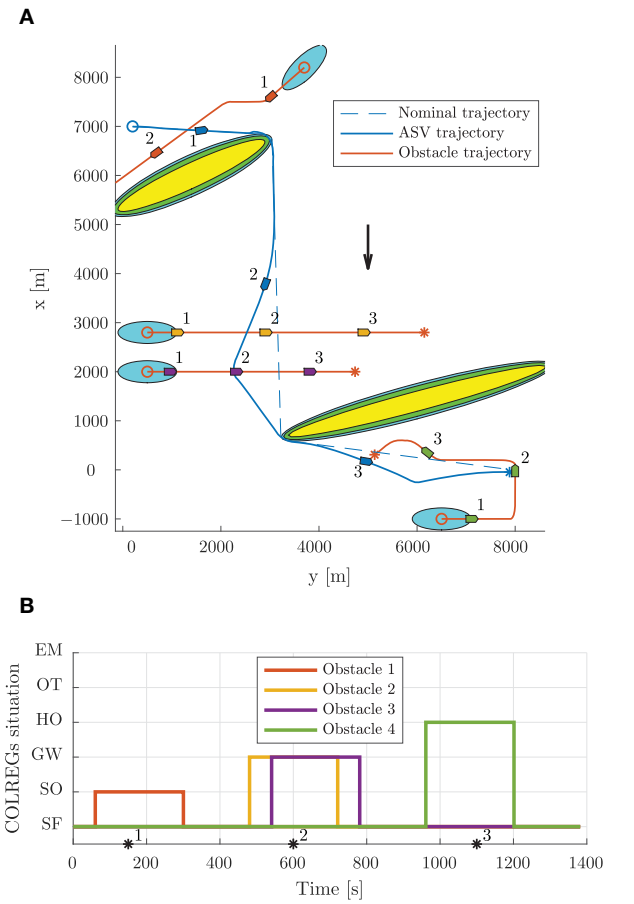


FIGURE 7 | Scenario 1: trajectory and COLREGs interpretation. The text marks denote the time steps [150, 600, 1100] s. **(A)** Trajectory plot. The initial position of the ownship and obstacles are shown with circles, with the blue ellipses illustrating the moving obstacle ellipse size. The vessel patches, which are overexaggerated for visualization, mark the ownship and obstacle poses at given time stamps. The static obstacles are shown in yellow, with the BC-MPC and mid-level safety margins enclosed around. The black arrow indicates the ocean current direction. **(B)** Output from the state machine for each obstacle. The asterisks mark time stamps, and the colors correspond to the obstacle patch colors in the trajectory plot.

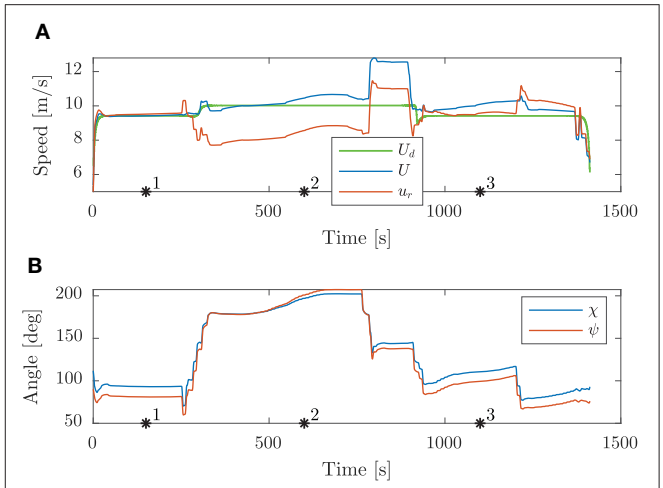


FIGURE 8 | Scenario 1: speed and angular trajectories. The asterisks mark the same time samples as in **Figure 7**. **(A)** Speed trajectories. **(B)** Angular trajectories.

initial and goal positions at $[7000, 200]^\top$ m and $[0, 7900]^\top$ m, respectively. The first obstacle is in a stand-on situation, where it is required to maneuver in order to avoid collision with the ownship, which is required to stand on. As shown in **Figure 7B**, the first obstacle is quickly considered as a stand-on situation, at which the mid-level algorithm disregards the obstacle and continues with the current speed and course. Following this, the obstacle maneuvers in accordance to the COLREGs, and we avoid collision. After the first static obstacle, we encounter two crossing vessels where the ownship is deemed the give-way vessel. In accordance with the COLREGs, we maneuver to starboard in order to pass behind both obstacles. Notice that the second give-way obstacle is detected as a give-way situation later than the first, since the entry criteria in the state machine includes the time to CPA, which is higher for the second give-way obstacle. After avoiding the two give-way obstacles, we converge toward the nominal trajectory and encounter a head-on situation. This is correctly identified by the state machine as head on, and we maneuver to starboard in order to avoid collision. Notice that even though the obstacle maneuvers, we keep the obstacle in the head-on state until we have passed it.

Figure 8 shows the speed and angular trajectories during Scenario 1, where the desired speed is calculated as the nominal speed at the closest point on the nominal trajectory given the ownship position. From this, we see that the mid-level and BC-MPC algorithms manage to track the desired nominal speed before and after the first static obstacle, where no obstacles require maneuvering away from the nominal trajectory. Notice that when encountering the two crossing obstacles, the mid-level algorithm chooses to slowly change the course, which is due to the attenuation of the give-way potential function and the large distance between the vessels. It would be better to make a clear course change, which is a subject of tuning. After passing the two crossing obstacles, the mid-level algorithm increases the speed in order to get back to the nominal trajectory, which is due to the algorithm attempting to keep the speed projected on the nominal trajectory equal as the desired nominal speed. Furthermore, notice that the mid-level algorithm actively controls the relative surge speed in order achieve the desired SOG, which is clearly seen when passing the first static obstacle.

6.3. Scenario 2

Scenario 2, shown in **Figure 9**, is more complex than Scenario 1, with a total of five moving obstacles, and has an ocean current of $[-1, 1]^\top$ m/s. The high-level planner plans a nominal trajectory between the initial and goal positions at $[200, 200]^\top$ m and $[5500, 7000]^\top$ m, respectively. The first obstacle is a crossing vessel, which similarly as in Scenario 1 is deemed to give way for the ownship, which should keep the current speed and course. However, in this scenario, the obstacle violates the COLREGs by

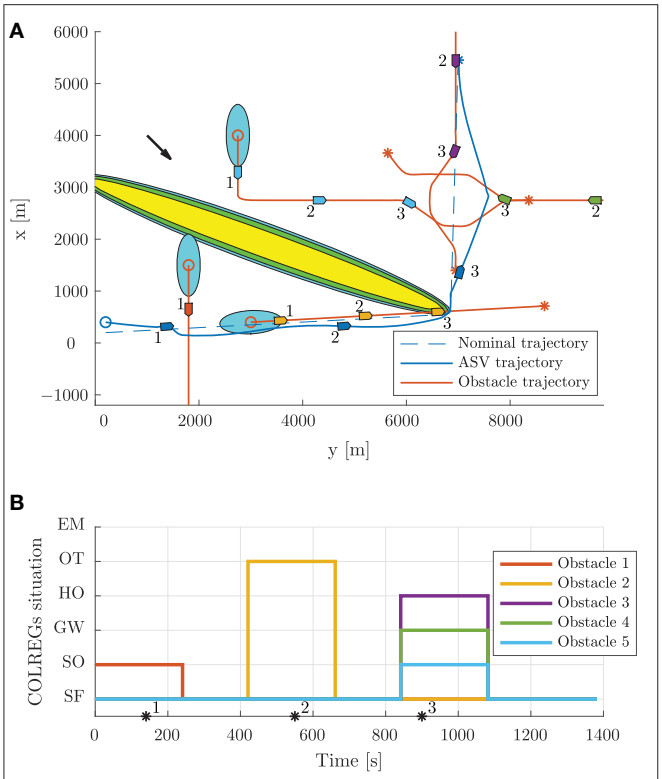


FIGURE 9 | Scenario 2: trajectory and COLREGs interpretation. The text marks denote the time steps [140, 550, 900] s. **(A)** Trajectory plot. The initial position of the ownship and obstacles are shown with circles, with the blue ellipses illustrating the moving obstacle ellipse size. The vessel patches, which are overexaggerated for visualization, mark the ownship and obstacle poses at given time stamps. The static obstacles are shown in yellow, with the BC-MPC and mid-level safety margins enclosed around. The black arrow indicates the ocean current direction. **(B)** Output from the state machine for each obstacle. The asterisks mark time stamps, and the colors correspond to the obstacle patch colors in the trajectory plot.

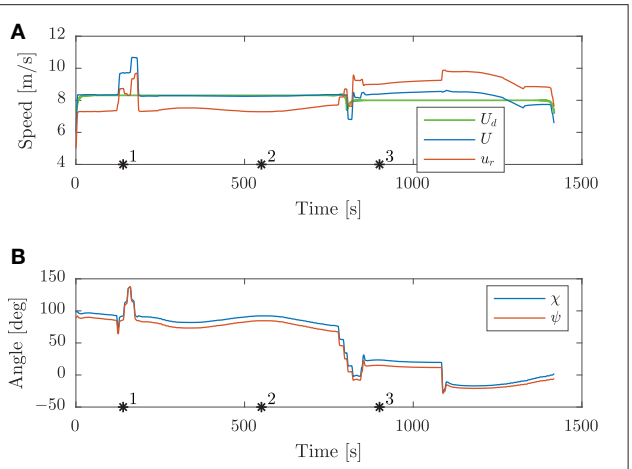


FIGURE 10 | Scenario 2: speed and angular trajectories. The asterisks mark the same time samples as in **Figure 9**. **(A)** Speed trajectories. **(B)** Angular trajectories.

not maneuvering in order to avoid collision. Therefore, the BC-MPC algorithm maneuvers to avoid collision when the obstacle gets so close that the safety margins of the BC-MPC algorithms is violated. The BC-MPC algorithm maneuvers to port, as advised by COLREGs Rule 17 for crossing situations where the stand-on vessel has to maneuver, and safely avoid the first obstacle. The second obstacle is overtaken by the ownship, and correctly considered as an overtaking situation by the state machine. For such an situation, there is no requirement on how the ownship should maneuver, except keeping clear from the overtaken vessel. After passing the second obstacle, we encounter a complex situation with simultaneous head-on, give-way and stand-on obligations. In this situation, each vessel, including the ownship, finds itself in a situation where a head-on and a give-way situation require starboard maneuvers, while a stand-on situation requires the vessel to keep the current speed and course. However, head-on and give-way obligations should be prioritized higher than stand-on situations, and the situation is quite easily solved by each vessel maneuvering to starboard and passing behind the vessel crossing from starboard. The mid-level algorithm solves this situation with the desirable behavior, and converges toward the nominal trajectory after the situation is resolved. As shown in **Figure 9B**, the state machine interprets the situations correctly.

From the speed trajectory in **Figure 10** it is clear that the mid-level algorithm follows the desired nominal speed also when overtaking the second obstacle.

6.4. Scenario 3

Scenario 3, shown in **Figure 11**, contains two moving obstacles on parallel courses with the ownship, and has an ocean current of $[-1, 1]^\top$ m/s. The high-level planner plans a nominal trajectory between the initial and goal positions at $[500, 500]^\top$ m and $[3328, 5399]^\top$ m, respectively, which results in a straight line trajectory with a course angle of 60°. The first obstacle travels at a higher speed than the ownship, while the second one travels at a lower speed and will be overtaken by the ownship. Since the obstacles are on parallel paths with the obstacle, the time to CPA is sufficiently high such that the obstacles are in the safe state, even though the vessels are quite close. However, both obstacles make sudden maneuvers to port dangerously close to the ownship and enters on a crossing course with the ownship. With respect to the COLREGs, the ownship is required to give way to both obstacles since they are crossing from the ownship's starboard side. One can, however, argue that the maneuvers displayed by the obstacles are dangerous and displays poor seamanship, such that the ownship should not be held accountable if a collision occurred. Nevertheless, the hybrid COLAV system manages to avoid both obstacles. As seen in **Figure 11B**, the first obstacle is sufficiently far away from the ownship to be considered as a give-way situation when the state machine interprets the situation, and the mid-level algorithm plans a trajectory passing behind the first obstacle. The second obstacle maneuvers to port even closer to the ownship, resulting in the distance to the critical point being within the threshold

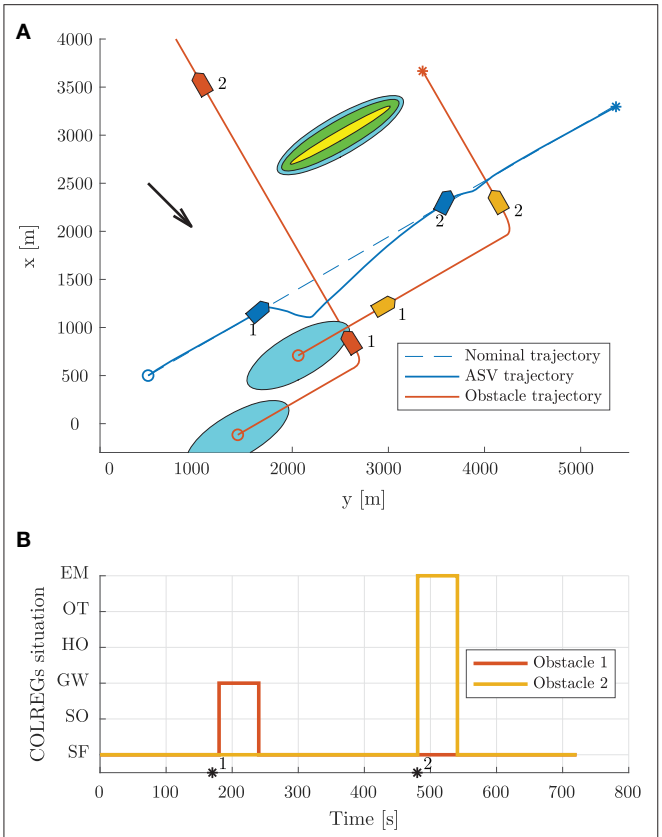


FIGURE 11 | Scenario 3: trajectory and COLREGs interpretation. The text marks denote the time steps [170, 480] s. **(A)** Trajectory plot. The initial position of the ownship and obstacles are shown with circles, with the blue ellipses illustrating the moving obstacle ellipse size. The vessel patches, which are overexaggerated for visualization, mark the ownship and obstacle poses at given time stamps. The static obstacles are shown in yellow, with the BC-MPC and mid-level safety margins enclosed around. The black arrow indicates the ocean current direction. **(B)** Output from the state machine for each obstacle. The asterisks mark time stamps, and the colors correspond to the obstacle patch colors in the trajectory plot.

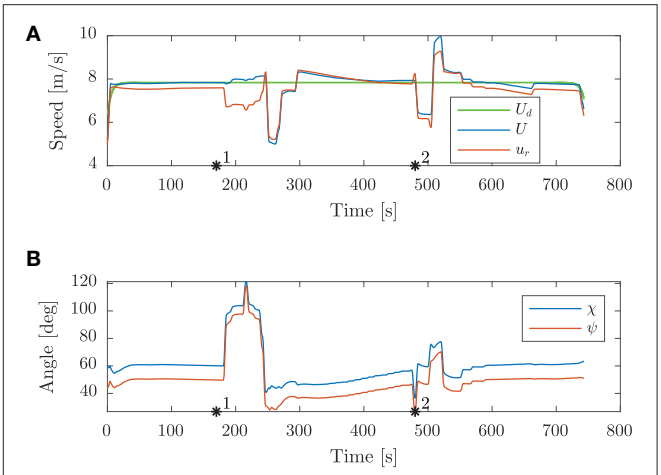


FIGURE 12 | Scenario 3: speed and angular trajectories. The asterisks mark the same time samples as in **Figure 11**. **(A)** Speed trajectories. **(B)** Angular trajectories.

TABLE 3 | Minimum distance to static and moving obstacles for the simulation scenarios.

Scenario	Minimum distance to static obstacles (m)	Minimum distance to moving obstacle number (m)				
		1	2	3	4	5
Scenario 1	93.7	634.3	596.3	522.7	726.8	–
Scenario 2	118.2	185.5	228.3	1,097.2	575.6	842.3
Scenario 3	1123.8	326.4	106.6	–	–	–

for entering the emergency situation when the state machine interprets the situation. In this situation, the mid-level algorithm disregards the obstacle and leaves it to the BC-MPC algorithm to avoid collision.

As seen in **Figure 12**, the mid-level algorithm both reduces the speed and changes the course to avoid the first obstacle. When approaching the second obstacle, the BC-MPC algorithm initiates a speed reduction, and after some time also maneuver to starboard in order to pass behind the obstacle and resolve the situation.

6.5. Simulation Summary

The simulation results show that the hybrid COLAV system is able to handle a wide range of situations, while also behaving in an energy-optimal way when moving obstacles are not interfering with the ownship trajectory. **Table 3** shows the minimum distance to static and moving obstacles for the scenarios.

The minimum distance to static obstacles is in Scenario 1 below the safety region size of the BC-MPC algorithm, which is intentional and caused by the algorithm using a smooth penalty function for interpreting static obstacles. The penalty function value increases linearly when moving further into the safety region, see Eriksen and Breivik (2019) for more details. The minimum distance to moving obstacles is a bit difficult to interpret, since the obstacle ship domains are non-circular, implying that the required clearance depends on relative position of the ownship with respect to the moving obstacles. However, we see that we have a larger clearance in head-on, give-way and stand-on situations where the obstacles comply with the COLREGs, and do not perform dangerous maneuvers (as in Scenario 3), compared to overtaking situations. The reason for this is that when overtaking (obstacle 2 in Scenario 2), we pass the obstacle on a parallel course, resulting in the minor axis of the moving obstacle ellipsis indicating the required clearance. Furthermore, we see that obstacle 1 in Scenario 2, which ignores its give-way obligation, comes significantly closer than other crossing obstacles except for those in Scenario 3. The reason for this is that the BC-MPC algorithm, which handles this situation, has a lower clearance requirement than the mid-level algorithm, which still should be considered as safe. In Scenario 3, the two obstacles display poor seamanship, and behave dangerously. Obstacle 1 is handled by the mid-level algorithm and passed

with a clearance lower than the major axis of the mid-level algorithm, which is caused by the BC-MPC algorithm "cutting the corner." The clearance should still be considered safe since we are behind the obstacle, and the clearance requirements of the BC-MPC algorithm is enforced. Obstacle 2, which is placed in the emergency state and handled by the BC-MPC algorithm, is passed with a clearance of only 106.6 m. This is lower than the clearance to Obstacle 1 in Scenario 2 (which violated its stand-on requirement), and is due to the BC-MPC algorithm having a non-symmetric obstacle ship domain function allowing for a smaller clearance when passing behind an obstacle than in front.

For the three scenarios, the high-level planner used an average of 67 s with a maximum of 93 s to compute the solution. Since the high-level planner is intended to be run off-line, this is well within reasonable limits. The mid-level algorithm used 0.60 s on average, and a maximum of 2.1 s, which we consider to be computationally feasible since the mid-level algorithm only is run every 60 s. The BC-MPC algorithm used 0.29 s on average, and a maximum of 0.63 s, which we also consider to be real-time feasible when the BC-MPC algorithm is run every 5 s. The BC-MPC algorithm is highly parallelizable, which could reduce the BC-MPC runtime by a large magnitude if required. The mid- and high-level algorithms may not return solutions as they are non-convex optimization problems, but the BC-MPC algorithm makes the hybrid COLAV system real-time feasible since it always will find a (potentially sub-optimal) solution.

7. CONCLUSION

In this paper, we have presented a three-layered hybrid COLAV system, compliant with COLREGs rules 8 and 13–17. As part of this, we have further developed the MPC-based mid-level COLAV algorithm in Eriksen and Breivik (2017b) and Bitar et al. (2019a) to comply with COLREGs rules 13–16 and parts of Rule 17, which includes developing a state machine for COLREGs interpretation. The hybrid COLAV system has a well-defined division of labor, including an inherent understanding of COLREGs Rule 17, where the mid-level algorithm obeys stand-on situations, while the BC-MPC algorithm handles situations where give-way vessels do not maneuver.

The hybrid COLAV system is verified through simulations, where we in three scenarios challenge the system with a number of different situations. The scenarios include multi-obstacle situations with multiple simultaneously active COLREGs rules, and situations where obstacles violate the COLREGs. Collision is avoided in all the scenarios, and we show that the ownship follows an energy-optimized trajectory generated by the high-level planner when moving obstacles do not interfere with this trajectory.

For further work, we suggest to:

- Investigate if using situation-dependent entry and exit criteria parameters in the state machine improves the performance.
- Expand the state machine with the possibility of transitioning from head-on, give-way and overtaking states to the emergency state for situations where obstacles behave dangerously or hostile.
- Develop a methodology for deciding tuning parameters.
- Perform simulations with noisy obstacle estimates to investigate how the state machine and mid-level algorithm respond to this.
- Explore the possibilities for integrating the COLREGs interpretation in the mid-level NLP, relaxing the assumption of the current COLREGs situation being valid for the entire prediction horizon.
- Investigate the possibility of including static obstacles from e.g., ENCs in the high- and mid-level algorithms.
- Simulate scenarios where multiple vessels running the hybrid COLAV system interact with each other.
- Validate the hybrid COLAV system in full-scale experiments.

DATA AVAILABILITY STATEMENT

The datasets generated for this study are available on request to the corresponding author.

AUTHOR CONTRIBUTIONS

The work in this article was the result of a collaboration between B-OE and GB, supervised by MB and AL. The contributions to the mid-level and short-term algorithms are made by B-OE, while the COLREGs interpreter was developed in collaboration between GB and B-OE. GB has implemented the high-level planner, and prepared this for integration with the developed simulator. B-OE has taken lead on writing the paper, in collaboration with GB. MB and AL have provided valuable feedback in the writing process.

FUNDING

This work was supported by the Research Council of Norway through project number 269116, project number 244116, as well as the Centers of Excellence funding scheme with project number 223254.

REFERENCES

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

Benjamin, M. R., Leonard, J. J., Curcio, J. A., and Newman, P. M. (2006). A method for protocol-based collision avoidance between autonomous marine surface craft. *J. Field Robot.* 23, 333–346. doi: 10.1002/rob.20121

Bitar, G., Breivik, M., and Lekkas, A. M. (2018). "Energy-optimized path planning for autonomous ferries," in *Proceedings of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)* (Opatija), 389–394.

Bitar, G., Eriksen, B.-O. H., Lekkas, A. M., and Breivik, M. (2019a). "Energy-optimized hybrid collision avoidance for ASVs," in *Proceedings of the 17th IEEE European Control Conference (ECC)* (Naples), 2522–2529.

Bitar, G., Vestad, V. N., Lekkas, A. M., and Breivik, M. (2019b). "Warm-started optimized trajectory planning for ASVs," in *Proceedings of the 12th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles (CAMS)* (Daejeon).

Campbell, S., Abu-Tair, M., and Naeem, W. (2014). An automatic COLREGs-compliant obstacle avoidance system for an unmanned surface vehicle. *Proc. Inst. Mech. Eng. M J. Eng. Marit. Environ.* 228, 108–121. doi: 10.1177/1475090213498229

Casalino, G., Turetta, A., and Simetti, E. (2009). "A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields," in *Proceedings of the 2009 IEEE OCEANS-EUROPE Conference* (Bremen).

Chauvin, C. (2011). Human factors and maritime safety. *J. Navig.* 64, 625–632. doi: 10.1017/S0373463311000142

Cockcroft, A. N., and Lameijer, J. N. F. (2004). *A Guide to the Collision Avoidance Rules*. Oxford, UK: Elsevier.

Deuflhard, P. (2011). *Newton Methods for Nonlinear Problems*. Berlin: Springer.

Eriksen, B.-O. H., and Breivik, M. (2017a). *Modeling, Identification and Control of High-Speed ASVs: Theory and Experiments*. Cham: Springer International Publishing, 407–431.

Eriksen, B.-O. H., and Breivik, M. (2017b). "MPC-based mid-level collision avoidance for ASVs using nonlinear programming," in *Proceedings of the 1st IEEE Conference on Control Technology and Applications (CCTA)* (Kohala Coast, HI), 766–772.

Eriksen, B.-O. H., and Breivik, M. (2018). "A model-based speed and course controller for high-speed ASVs," in *Proceedings of the 11th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS)* (Opatija), 317–322.

Eriksen, B.-O. H., and Breivik, M. (2019). Short-term ASV collision avoidance with static and moving obstacles. *Model. Identif. Control* 40, 177–187. doi: 10.4173/mic.2019.3.4

Eriksen, B.-O. H., Breivik, M., Wilthil, E. F., Flåten, A. L., and Brekke, E. F. (2019). The branching-course model predictive control algorithm for maritime collision avoidance. *J. Field Robot.* 36, 1222–1249. doi: 10.1002/rob.21900

Eriksen, B.-O. H., Wilthil, E. F., Flåten, A. L., Brekke, E. F., and Breivik, M. (2018). "Radar-based maritime collision avoidance using dynamic window," in *Proceedings of the 2018 IEEE Aerospace Conference* (Big Sky, MT), 1–9.

Hagen, I. B., Kufoalor, D. K. M., Brekke, E., and Johansen, T. A. (2018). "MPC-based collision avoidance strategy for existing marine vessel guidance systems," in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)* (Brisbane, QLD), 7618–7623.

Harati-Mokhtari, A., Wall, A., Brooks, P., and Wang, J. (2007). Automatic identification system (AIS): data reliability and human error implications. *J. Navig.* 60, 373–389. doi: 10.1017/S0373463307004298

Kufoalor, D. K. M., Brekke, E. F., and Johansen, T. A. (2018). "Proactive collision avoidance for ASVs using a dynamic reciprocal velocity obstacles method," in *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Madrid), 2402–2409.

Kuwata, Y., Wolf, M. T., Zarzhitsky, D., and Huntsberger, T. L. (2014). Safe maritime autonomous navigation with COLREGS, using velocity obstacles. *IEEE J. Ocean. Eng.* 39, 110–119. doi: 10.1109/joe.2013.2254214

Levander, O. (2017). Autonomous ships on the high seas. *IEEE Spectr.* 54, 26–31. doi: 10.1109/MSPEC.2017.7833502

Loe, Ø. A. G. (2008). *Collision avoidance for unmanned surface vehicles* (Master's thesis), Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

Švec, P., Shah, B. C., Bertaska, I. R., Alvarez, J., Sinisterra, A. J., von Ellenrieder, K., et al. (2013). "Dynamics-aware target following for an autonomous surface vehicle operating under COLREGs in civilian traffic," in *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Tokyo), 3871–3878.

Szlapczynski, R., and Szlapczynska, J. (2017). Review of ship safety domains: models and applications. *Ocean Eng.* 145, 277–289. doi: 10.1016/j.oceaneng.2017.09.020

Tam, C., and Bucknall, R. (2010). Collision risk assessment for ships. *J. Mar. Sci. Technol.* 15, 257–270. doi: 10.1007/s00773-010-0089-7

Wächter, A., and Biegler, L. T. (2005). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.* 106, 25–57. doi: 10.1007/s10107-004-0559-y

Wu, D. (2019). *Proactive maritime collision avoidance based on historical AIS data* (Master's thesis), Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

Zhang, X., Liniger, A., Sakai, A., and Borrelli, F. (2018). "Autonomous parking using optimization-based collision avoidance," in *Proceedings of the 57th IEEE Conference on Decision and Control (CDC)* (Miami, FL), 4327–4332.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Robotics and AI

METHODS
published: 20 March 2020
doi: 10.3389/frobt.2020.00032

Reinforcement Learning-Based Tracking Control of USVs in Varying Operational Conditions

Andreas B. Martinsen[1]*, *Anastasios M. Lekkas*[1,2], *Sébastien Gros*[1], *Jon Arne Glomsrud*[3] and *Tom Arne Pedersen*[3]

[1] Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway, [2] Centre for Autonomous Marine Operations and Systems, Norwegian University of Science and Technology, Trondheim, Norway, [3] Digital Assurance Program, Group Technology and Research, DNV GL, Trondheim, Norway

We present a reinforcement learning-based (RL) control scheme for trajectory tracking of fully-actuated surface vessels. The proposed method learns online both a model-based feedforward controller, as well an optimizing feedback policy in order to follow a desired trajectory under the influence of environmental forces. The method's efficiency is evaluated via simulations and sea trials, with the unmanned surface vehicle (USV) *ReVolt* performing three different tracking tasks: The four corner DP test, straight-path tracking and curved-path tracking. The results demonstrate the method's ability to accomplish the control objectives and a good agreement between the performance achieved in the Revolt Digital Twin and the sea trials. Finally, we include an section with considerations about assurance for RL-based methods and where our approach stands in terms of the main challenges.

Keywords: reinforcement learning, trajectory tracking, optimal control, model-based adaptive control, approximate dynamic programming (ADP), dynamic positioning (DP), autonomous ships, system identification

OPEN ACCESS

Edited by:
Marco Bibuli,
Italian National Research Council, Italy

Reviewed by:
Ning Wang,
Dalian Maritime University, China
Farah Bouakrif,
University of Jijel, Algeria

***Correspondence:**
Andreas B. Martinsen
andreas.b.martinsen@ntnu.no

Specialty section:
This article was submitted to Robotic Control Systems, a section of the journal Frontiers in Robotics and AI

Received: *06 November 2019*
Accepted: *20 February 2020*
Published: *20 March 2020*

Citation:
Martinsen AB, Lekkas AM, Gros S, Glomsrud JA and Pedersen TA (2020) Reinforcement Learning-Based Tracking Control of USVs in Varying Operational Conditions. Front. Robot. AI 7:32. doi: 10.3389/frobt.2020.00032

1. INTRODUCTION

Control of marine vehicles is a challenging problem, mostly due to the unpredictable nature of the sea and the difficulty in developing accurate mathematical models to represent the varying marine vehicle dynamics. As a result, considerable research effort has been dedicated to the topic since the early 90's (Fossen, 1994), resulting in a vast literature utilizing ideas from virtually every branch of control engineering: Linear, non-linear, adaptive, intelligent, optimal, fuzzy, and stochastic control approaches, to name a few, have been developed and tested over the years, and many of their properties are well-understood (Hasegawa et al., 1989; Pettersen and Egeland, 1996; Katebi et al., 1997; Fossen, 2000; McGookin et al., 2000; Soetanto et al., 2003; Wang et al., 2015; Do, 2016). Due to the fact that the hydrodynamic coefficients, and consequently the behavior, of a marine vehicle can vary significantly in different speed regimes, a common approach has been to design controllers for specific motion control scenarios. This approach simplifies the vessel modeling process and has led to dynamic positioning (DP) and station keeping controllers for speeds close to zero, and trajectory tracking or path following (depending on whether temporal constraints are considered) controllers when a vessel is in transit mode. Naturally, the main drawback is that, when moving from one speed regime to another, controllers and/or models with different properties are needed. Two well-researched

ways to achieve such performance diversity with conventional methods are to design numerous controllers and switch among them when needed, or to use adaptive approaches. To this end, research effort has been dedicated to developing flexible methods for updating the model parameters by, for instance, using system identification methods or parameter estimation via neural networks (Källström and Åström, 1981; Kallstrom, 1982; Fossen et al., 1996; Sutton et al., 1997; Mišković et al., 2011; Dai et al., 2012; Wang et al., 2017). In the majority of the aforementioned works, model-based approaches exploiting human knowledge on hydrodynamics and the laws of motion were considered.

Reinforcement learning (RL), also known as neuro-dynamic programming or approximate dynamic programming, is a field of research developed by the Artificial Intelligence (AI) community for achieving optimal sequential decision making under system and environment uncertainty. The roots of RL can be traced back to the 60's and a thorough overview of its evolution can be found in Sutton and Barto (2018) and Bertsekas (2019). Contrary to optimal control theory, RL is based on *evaluative*, rather than instructive, feedback and comes in different forms, which may or may not include partial knowledge of the environment or the system. The process typically involves hand-engineering a reward function, which assigns a reward, or penalty, to the actions that induce desired, or undesired, outcomes, respectively. An RL algorithm is then assigned to find a policy (or controller, in control engineering terminology) that solves the control objective optimally, given the problem constraints and uncertainties. To sum up, RL algorithms use the reward function as a guide, and through trial and error, learn to model the system and its environment, which then leads to a policy that provides an optimal solution to the assigned problem.

Despite a number of successes for RL on simple problems, including algorithms, such as *Q-learning* and *REINFORCE*, the field has seen limited interest. In recent years there has however been a resurgence of interest due to the development of Deep Reinforcement Learning (DRL), starting with Deep Mind developing the *Deep Q-Network* (DQN) algorithm that achieved superhuman performance in several Atari games (Mnih et al., 2013), followed by Deep Mind's *AlphaGo* algorithm becoming the first computer program to beat a human champion in the game of *Go* (Silver et al., 2016). Since then, DRL has been successful in surpassing all previous computer programs in chess and learning how to accomplish complex robotic tasks (Silver et al., 2017; Andrychowicz et al., 2018). Given DRL's ability to tackle problems with high uncertainty, implementations to motion control scenarios involving marine vessels have been presented recently (Shen and Guo, 2016; Zhang et al., 2016; Pham Tuyen et al., 2017; Yu et al., 2017; Cheng and Zhang, 2018; Martinsen and Lekkas, 2018a,b). In most of these works the authors implemented algorithms pertaining to the class of *actor-critic* RL methods, which involves two parts (Konda and Tsitsiklis, 2000): The *actor*, where the gradient of the performance is estimated and the policy parameters are directly updated in a direction of improvement. The main drawbacks of the actor are that it is prone to variance and the new gradient is estimated independently of past estimates. The *critic*, learns an approximation of the value function, leading to an approximate solution to the Bellman or Hamilton-Jacobi-Bellman equation, which then is expected to prescribe a near-optimal policy. The critic's main drawback is that it lacks reliable guarantees in terms of near-optimality of the resulting policy. The actor-critic approach involves the actor improving the policy parameters' estimation based on the approximations learned by the critic. In the case of DRL, one main novelty was the use of two DNNs as function approximators of the policy and the value function, which resulted in considerably improved performance compared to previous approaches. However, DNNs have drawbacks, with some of the most important being lack of transparency and interpretability, lack of robustness, and inability to generalize to situations beyond their past experiences.

In this paper, we follow and extend the work by Kamalapurkar et al. (2018) and Walters et al. (2018) in order to build a trajectory tracking control system for a fully-actuated unmanned surface vehicle (USV). Conceptually, the approach is quite similar to dynamic positioning (DP) (Sørensen, 2011), but extends to higher velocity operational domains, while also trying to optimize tracking performance and compensate for environmental forces (Lekkas and Fossen, 2014). The method combines elements from reinforcement learning, Lyapunov stability theory and system identification: We assume the structure of the vessel model is known but all of its parameters are unknown and have to be estimated online, as well as updated accordingly when the operational conditions change. Then we derive the tracking error dynamics for a generic reference trajectory and a stabilizing parametric control law (the *actor*), whose parameters are estimated during operation.

In order to validate the control scheme, the proposed method was tested in both in simulations, and on a physical model of DNV GL's *ReVolt* platform.

2. REINFORCEMENT LEARNING-BASED TRAJECTORY TRACKING

In this section we will derive a trajectory tracking control system for fully-actuated USVs. Since the approach is a model based reinforcement learning approach, we will start by looking at how ASVs can be modeled, and how the models can be approximated online using system identification. We will derive a feedforward control law for tracking the desired trajectory, and a feedback control law based on reinforcement learning, for controlling the drift of the vessel in a way that minimizes a given cost function.

2.1. Vessel Model

The mathematical model used to describe the system can then be kept reasonably simple by limiting it to the planar position and orientation of the vessel. 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, (x, y) 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 1**. Using the notation in Fossen (2011) we can describe a 3-DOF vessel model as follows

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu}, \\ \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} &= \boldsymbol{\tau}_{\text{Thrust}} + \boldsymbol{\tau}_{\text{Environment}} \end{aligned} \tag{1}$$

where $\boldsymbol{M} \in \mathbb{R}^{3\times3}$, $\boldsymbol{D}(\boldsymbol{\nu}) \in \mathbb{R}^{3\times3}$, $\boldsymbol{C}(\boldsymbol{\nu}) \in \mathbb{R}^{3\times3}$, $\boldsymbol{\tau}_{\text{Thrust}}, \boldsymbol{\tau}_{\text{Environment}} \in \mathbb{R}^3$ and $\boldsymbol{J}(\boldsymbol{\eta}) \in SO(3)$ are the inertia matrix, damping matrix, coriolis matrix, control input vector, environmental forces, and rotation matrix, respectively. The rotational matrix $\boldsymbol{J}(\boldsymbol{\eta}) \in SO(3)$ is given by

$$\boldsymbol{J}(\boldsymbol{\eta}) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

and is the rotation from the body frame to the earth-fixed North East Down (NED) reference frame.

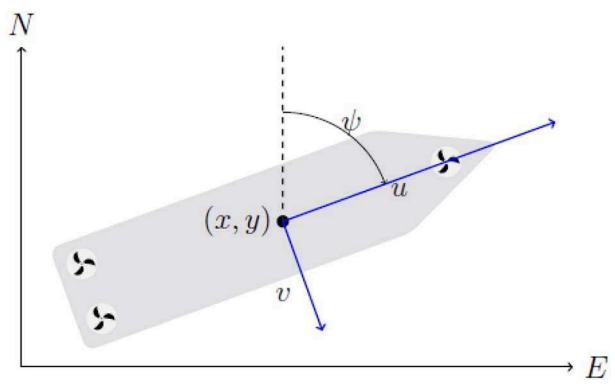


FIGURE 1 | 3-DOF vessel centered at (x, y), with surge velocity u, sway velocity v, heading ψ in a North-East-Down (NED) reference frame.

2.2. Model Approximation

While the structure of a vessel model, as given above, is well-known, the model parameters are often difficult to find. For our approach we wish to make as few assumptions on the parameters of the vessel model as possible, and use online system identification in order to model the vessel based on gathered data. For this we assume that we know the model structure as given in (1), but that the model parameters are unknown. Splitting the model into a known and unknown part, we get the following:

$$\dot{\boldsymbol{x}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}) + f_1(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u} \tag{3}$$

where $f_1(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are known, and $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ is unknown. For the vessel model in (1), with the state vector $\boldsymbol{x} = [\boldsymbol{\eta}, \boldsymbol{\nu}]^\top$ and the control vector $\boldsymbol{u} = \boldsymbol{\tau}_{\text{Thrust}}$. We have the following:

$$\begin{aligned} f_{\boldsymbol{\theta}}(\boldsymbol{x}) &= \begin{bmatrix} \boldsymbol{0}_{3\times1} \\ -\boldsymbol{M}^{-1}\left(\boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{\tau}_{\text{Environment}}\right) \end{bmatrix} \\ f_1(\boldsymbol{x}) &= \begin{bmatrix} \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu} \\ \boldsymbol{0}_{3\times1} \end{bmatrix} \\ g(\boldsymbol{x}) &= \begin{bmatrix} \boldsymbol{0}_{3\times3} \\ \boldsymbol{M}^{-1} \end{bmatrix} \end{aligned}$$

hence we assume the mass matrix is known, but the damping and coriolis matrix are unknown. For the damping and coriolis matrices we assume the vessel has port starboard symmetry, from Fossen (2011) this gives the following structure.

$$\boldsymbol{C}(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & Y_{\dot{v}} \cdot v + Y_{\dot{r}} \cdot r \\ 0 & 0 & -X_{\dot{u}} \cdot u \\ -Y_{\dot{v}} \cdot v + Y_{\dot{r}} \cdot r & X_{\dot{u}} \cdot u & 0 \end{bmatrix} \tag{4}$$

$$\boldsymbol{D}(\boldsymbol{\nu}) = \begin{bmatrix} -X_u - X_{|u|u} \cdot |u| & 0 & 0 \\ 0 & -Y_v - Y_{|v|v} \cdot |v| - Y_{|r|v} \cdot |r| & -Y_r - Y_{|v|r} \cdot |v| - Y_{|r|r} \cdot |r| \\ 0 & -N_v - N_{|v|v} \cdot |v| - N_{|r|v} \cdot |r| & -N_r - N_{|v|r} \cdot |v| - N_{|r|r} \cdot |r| \end{bmatrix} \tag{5}$$

For the damping matrix $\boldsymbol{D}(\boldsymbol{\nu})$, both linear and non-linear terms are included. The linear terms are important for low speed maneuvering and station keeping, while ensuring the velocity converges exponentially to zero. The non-linear terms are required as they dominate at higher velocities. This ensures that the model is able to handle a large range of velocities, i.e., it can be used for both high speed trajectory tracking and low speed station keeping and dynamic positioning. For the coriolis matrix, we use only the added mass terms. Since the structure of the rigid body, and added mass is the same for the coriolis matrix, the coriolis matrix given above will be able to capture both the added mass and rigid body dynamics.

In addition to learning the vessel dynamics, we also wanted to be able to compensate for environmental forces. In order to allow for the environmental forces to be learned, they are modeled as an additional unknown pressure vector $\boldsymbol{p}_{\text{env}}^{\text{NED}} = [p_{\text{North}}, p_{\text{East}}, 0]^\top$ assumed constant in the NED frame. The resulting force in the body frame is then assumed to be proportional to the cross sectional area of the vessel times the pressure in the body frame, giving the following relationship.

$$\boldsymbol{\tau}_{\text{Environment}}^{\text{body}} = \text{diag}([w, l, 0])\boldsymbol{J}^\top(\boldsymbol{\nu})\boldsymbol{p}_{\text{Environment}}^{\text{NED}} \tag{6}$$

where w and l are the width and length of the vessel, respectively, note that for better accuracy calculated pressure coefficients based on the design of the hull may be used instead of the width and length. The unknown parameters are

$$\begin{aligned} \boldsymbol{\theta} = [&X_{\dot{u}}, Y_{\dot{v}}, Y_{\dot{r}}, X_u, Y_v, Y_r, N_v, N_r, X_{|u|u}, Y_{|v|v}, Y_{|v|r}, Y_{|r|v}, Y_{|r|r}, \\ &N_{|v|v}, N_{|v|r}, N_{|r|v}, N_{|r|r}, p_{\text{North}}, p_{\text{East}}]^\top \end{aligned} \tag{7}$$

and the function $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ can be written as a linear function in $\boldsymbol{\theta}$:

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = Y(\boldsymbol{x})\boldsymbol{\theta} \tag{8}$$

where $Y(\boldsymbol{x})$ is:

$$Y(\boldsymbol{x}) = \begin{bmatrix} \mathbf{0}_{3\times 3} \\ -\boldsymbol{M}^{-1} \end{bmatrix} \begin{bmatrix} 0 & v\cdot r & r^2 & -u & 0 & 0 & 0 & 0 & -|u|u & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w\cos\psi & w\sin\psi \\ -u\cdot r & 0 & 0 & 0 & -v & -r & 0 & 0 & 0 & -|v|v & -|v|r & -|r|v & -|r|r & 0 & 0 & 0 & 0 & -l\sin\psi & l\cos\psi \\ u\cdot v & -v\cdot u & -r\cdot u & 0 & 0 & 0 & -v & -r & 0 & 0 & 0 & 0 & 0 & -|v|v & -|v|r & -|r|v & -|r|r & 0 & 0 \end{bmatrix} \tag{9}$$

We therefore obtain the following parametric model:

$$\dot{\boldsymbol{x}} = Y(\boldsymbol{x})\boldsymbol{\theta} + f_1(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u}, \tag{10}$$

which is linear in the parameters $\boldsymbol{\theta}$.

2.2.1. Model Assumptions

- The vessel is port starboard symmetric, with a structure as given as in (**Figure 1**).
- The vessel dampening is linear and quadratic with respect to the linear and angular velocity.
- Environmental forces are constant in the NED frame, and proportional to vessel cross section.
- The vessel is fully actuated.

2.3. Trajectory Tracking

In this section we will develop an adaptive feedforward control law which given a time-varying trajectory, finds the control inputs required to follow the trajectory, given the model approximation found in the previous section.

When the control objective is to track a bounded continuously differentiable signal $\boldsymbol{x}_d$, the dynamics of the tracking error $\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{x}_d$ can be written as

$$\dot{\boldsymbol{e}} = f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u} - \dot{\boldsymbol{x}}_d \tag{11}$$

Assuming $g(\boldsymbol{x})$ is bounded and has full column rank for all $\boldsymbol{x}$ (Kamalapurkar et al., 2018), then the system is controllable, which in this case holds as the vessel is fully actuated. This gives the feedforward control for the reference trajectory as :

$$\boldsymbol{u}_d(\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d) = g^+(\boldsymbol{x}_d)(\dot{\boldsymbol{x}}_d - f(\boldsymbol{x}_d)) \tag{12}$$

where g^+ is the left Moore–Penrose pseudo-inverse, given as $g^+ = (g^\top g)^{-1} g^\top$. Using a reference model $\dot{\boldsymbol{x}}_d = h_d(\boldsymbol{x}_d)$, the feedforward control for the reference trajectory can be written as:

$$\boldsymbol{u}_d(\boldsymbol{x}_d) = g^+(\boldsymbol{x}_d)(h_d(\boldsymbol{x}_d) - f(\boldsymbol{x}_d)) \tag{13}$$

We can then formulate the tracking problem as the following time-invariant optimal control problem.

$$\underbrace{\begin{bmatrix} \dot{\boldsymbol{e}} \\ \dot{\boldsymbol{x}}_d \end{bmatrix}}_{\dot{\boldsymbol{\zeta}}} = \underbrace{\begin{bmatrix} f(\boldsymbol{e}+\boldsymbol{x}_d) + g(\boldsymbol{e}+\boldsymbol{x}_d)\boldsymbol{u}_d(\boldsymbol{x}_d) \\ h_d(\boldsymbol{x}_d) \end{bmatrix}}_{F(\boldsymbol{\zeta})} + \underbrace{\begin{bmatrix} g(\boldsymbol{e}+\boldsymbol{x}_d) \\ 0 \end{bmatrix}}_{G(\boldsymbol{\zeta})} \boldsymbol{\pi} \tag{14}$$

Where $\boldsymbol{\pi}$ is an input correction for the drift dynamics, which we will define in the next section. Given the parametric model in (10), the parametric version of the tracking problem is given as:

$$\underbrace{\begin{bmatrix} \dot{\boldsymbol{e}} \\ \dot{\boldsymbol{x}}_d \end{bmatrix}}_{\dot{\boldsymbol{\zeta}}} = \underbrace{\begin{bmatrix} Y(\boldsymbol{e}+\boldsymbol{x}_d)\boldsymbol{\theta} + f_1(\boldsymbol{e}+\boldsymbol{x}_d) + g(\boldsymbol{e}+\boldsymbol{x}_d)\boldsymbol{u}_d(\boldsymbol{x}_d;\boldsymbol{\theta}) \\ h_d(\boldsymbol{x}_d) \end{bmatrix}}_{F(\boldsymbol{\zeta};\boldsymbol{\theta})} + \underbrace{\begin{bmatrix} g(\boldsymbol{e}+\boldsymbol{x}_d) \\ 0 \end{bmatrix}}_{G(\boldsymbol{\zeta})} \boldsymbol{\pi} \tag{15}$$

where the parametric feedforward control for the reference trajectory $\boldsymbol{u}_d(\boldsymbol{x}_d;\boldsymbol{\theta})$ is given as:

$$\boldsymbol{u}_d(\boldsymbol{x}_d;\boldsymbol{\theta}) = g^+(\boldsymbol{x}_d)(h_d(\boldsymbol{x}_d) - Y(\boldsymbol{x}_d)\boldsymbol{\theta} - f_1(\boldsymbol{x}_d)) \tag{16}$$

Given the formulation above, with the feedforward control for the reference trajectory $\boldsymbol{u}_d(\boldsymbol{x}_d)$, and the optimal model parameters $\boldsymbol{\theta}^*$, the exact feedforward control for the reference trajectory is possible to compute. The dynamics above guarantee trajectory tracking when $\dot{\boldsymbol{e}} = 0$, i.e., when the tracking error is zero. When the tracking error is not zero however, we need to control the drift dynamics in order to ensure convergence to the desired trajectory by designing the feedback control $\boldsymbol{\pi}(t)$ such that $\lim_{t\to\infty} e(t) = 0$. The objective of the optimal control problem is to design the feedback control law $\boldsymbol{\pi}(t)$ such that it minimizes a given cost function.

2.4. Approximate Optimal Control of Drift Dynamics

In the previous section we developed a feedforward control law $\boldsymbol{u}_d(\boldsymbol{x}_d;\boldsymbol{\theta})$ for tracking a desired trajectory. Due to inaccuracies in model approximation and disturbances, using only the feedforward control law, the vessel will experience drift. In order to compensate for the inevitable drift, we will in this section develop a feedback control law $\boldsymbol{\pi}(\cdot)$, which controls the drift dynamics in a way that optimizes a given cost function. We will additionally show how the parameters of the feedback control law can be learned by using reinforcement learning.

The optimal control problem we wish to solve is that of minimizing the cost function:

$$J(\boldsymbol{\zeta}, \boldsymbol{\pi}) = \int_{t_0}^{\infty} r(\boldsymbol{\zeta}(\tau), \boldsymbol{\pi}(\tau))d\tau \tag{17}$$

Where $r(\cdot)$ is scalar function defining the local cost, and should not be confused with the yaw rate. The cost function is defined as:

$$r(\boldsymbol{\zeta}, \boldsymbol{\pi}) = Q(\boldsymbol{\zeta}) + \boldsymbol{\pi}^\top \boldsymbol{R}\boldsymbol{\pi} \tag{18}$$

where $\boldsymbol{R} \succ 0$ is a positive definite symmetric matrix. And $Q(\boldsymbol{\zeta})$ is a positive definite function. Assuming that a minimizing control policy $\boldsymbol{\pi}(\cdot)$ exists, the optimal value function is given as:

$$V^*(\boldsymbol{\zeta}) = \min_{\boldsymbol{\pi}(\tau),\, \tau\in[t_0,\infty)} \int_{t_0}^{\infty} r(\boldsymbol{\zeta}(\tau), \boldsymbol{\pi}(\tau))d\tau \tag{19}$$

We can now note that for a small time step Δt, the above expression can be formulated as:

$$V^*(\boldsymbol{\zeta}(t)) = \min_{\boldsymbol{\pi}(\tau),\, \tau\in[t,t+\Delta t)} \int_{t}^{t+\Delta t} r(\boldsymbol{\zeta}(\tau), \boldsymbol{\pi}(\tau))d\tau + V^*(\boldsymbol{\zeta}(t+\Delta t))$$

Taking the limit of this as $\Delta t \to 0$, for the optimal value function under the optimal policy, we get (Doya, 2000):

$$V^*(\boldsymbol{\zeta}(t)) = \min_{\boldsymbol{\pi}(t)} r(\boldsymbol{\zeta}(t), \boldsymbol{\pi}(t)) + V^*(\boldsymbol{\zeta}(t)) + \dot{V}^*(\boldsymbol{\zeta}(t))$$

Simplifying this we get the Hamilton-Jacobi-Bellman (HJB) equation for the optimal control problem as follows:

$$\begin{aligned} H^* &= \dot{V}^*(\boldsymbol{\zeta}) + r(\boldsymbol{\zeta}, \boldsymbol{\pi}^*(\boldsymbol{\zeta})) \\ &= \nabla_{\boldsymbol{\zeta}} V^*(\boldsymbol{\zeta})^\top \dot{\boldsymbol{\zeta}} + r(\boldsymbol{\zeta}, \boldsymbol{\pi}^*(\boldsymbol{\zeta})) \\ &= \nabla_{\boldsymbol{\zeta}} V^*(\boldsymbol{\zeta})^\top \left(F(\boldsymbol{\zeta}) + G(\boldsymbol{\zeta})\boldsymbol{\pi}^*(\boldsymbol{\zeta})\right) + r(\boldsymbol{\zeta}, \boldsymbol{\pi}^*(\boldsymbol{\zeta})) = 0 \end{aligned} \tag{20}$$

Where H^*, $\boldsymbol{\pi}^*$ and V^* is the optimal hamiltonian, policy and value function, respectively. From calculus of variation (Liberzon, 2011) we have the Hamiltonian minimization condition, which states that a value function V is the optimal Value function if and only if there exists a controller $\boldsymbol{\pi}(\cdot)$ and trajectory $\boldsymbol{\zeta}(\cdot)$ under $\boldsymbol{\pi}(\cdot)$ satisfy the equation:

$$\begin{aligned} &\nabla_{\boldsymbol{\zeta}} V(\boldsymbol{\zeta})^\top \left(F(\boldsymbol{\zeta}) + G(\boldsymbol{\zeta})\boldsymbol{\pi}(\boldsymbol{\zeta})\right) + r(\boldsymbol{\zeta}, \boldsymbol{\pi}(\boldsymbol{\zeta})) \\ &= \min_{\hat{\pi}\in U} \{\nabla_{\boldsymbol{\zeta}} V(\boldsymbol{\zeta})^\top \left(F(\boldsymbol{\zeta}) + G(\boldsymbol{\zeta})\hat{\boldsymbol{\pi}}(\boldsymbol{\zeta})\right) + r(\boldsymbol{\zeta}, \hat{\boldsymbol{\pi}}(\boldsymbol{\zeta}))\} \end{aligned} \tag{21}$$

The necessary conditions for this to hold are:

$$\nabla_{\boldsymbol{\pi}} \left(\nabla_{\boldsymbol{\zeta}} V(\boldsymbol{\zeta})^\top \left(F(\boldsymbol{\zeta}) + G(\boldsymbol{\zeta})\boldsymbol{\pi}(\boldsymbol{\zeta})\right) + r(\boldsymbol{\zeta}, \boldsymbol{\pi}(\boldsymbol{\zeta}))\right) = 0 \tag{22}$$

which gives the closed form solution of the optimal controller as:

$$\begin{aligned} G^\top(\boldsymbol{\zeta})\left(\nabla_{\boldsymbol{\zeta}} V(\boldsymbol{\zeta})\right) + \nabla_{\boldsymbol{\pi}} r(\boldsymbol{\zeta}, \boldsymbol{\pi}) = 0 \Leftrightarrow \\ 2\boldsymbol{R}\boldsymbol{\pi} = -G^\top(\boldsymbol{\zeta})\left(\nabla_{\boldsymbol{\zeta}} V(\boldsymbol{\zeta})\right) \Leftrightarrow \end{aligned}$$

$$\boldsymbol{\pi}^*(\boldsymbol{\zeta}) = -\frac{1}{2}\boldsymbol{R}^{-1}G^\top(\boldsymbol{\zeta})\left(\nabla_{\boldsymbol{\zeta}} V(\boldsymbol{\zeta})\right) \tag{23}$$

Hence assuming that an optimal controller exists, the closed form solution given by the HJB equation is given by (23). Note that the value function is assumed time independent, and hence we are looking for a stationary solution of the HJB equation. This holds true, as the reformulation into a trajectory tracking problem (15) gives a time independent system.

The Universal Approximation theorem (Kamalapurkar et al., 2018, Property 2.3) states that a single layer neural network can simultaneously approximate a function and its derivative given a sufficiently large number of basis functions. Using this, we can approximate any continuous function as:

$$V(\boldsymbol{x}) = \boldsymbol{W}^\top \sigma(\boldsymbol{x}) + \epsilon(\boldsymbol{x}) \tag{24}$$

where $\boldsymbol{W}$ is the weighting matrix, $\sigma(\boldsymbol{x})$ is the vector of basis functions, and $\epsilon(\boldsymbol{x})$ is the approximation error, which can be made arbitrarily small by increasing the number of basis functions. Note that the basis functions can here be chosen to be any parameterization, such as Radial-Basis functions, polynomials or even a Fourier series. Using this we can represent the value function as a neural network which is linear in the parameters, giving the optimal value function:

$$V^*(\boldsymbol{\zeta}) = \boldsymbol{W}^\top \sigma(\boldsymbol{\zeta}) + \epsilon(\boldsymbol{\zeta}) \tag{25}$$

and the optimal policy as a feedback control law on the form:

$$\boldsymbol{\pi}^*(\boldsymbol{\zeta}) = -\frac{1}{2}\boldsymbol{R}^{-1}G^\top(\boldsymbol{\zeta})\left(\nabla_{\boldsymbol{\zeta}}\sigma(\boldsymbol{\zeta})^\top \boldsymbol{W} + \nabla_{\boldsymbol{\zeta}}\epsilon^\top(\boldsymbol{\zeta})\right) \tag{26}$$

By making the parameterizations sufficiently rich, we make the approximation error small. We can then use the approximations given below, for the value function and control policy, respectively.

$$\hat{V}(\boldsymbol{\zeta}; \hat{\boldsymbol{W}}_c) = \hat{\boldsymbol{W}}_c^\top \sigma(\boldsymbol{\zeta}) \tag{27}$$

$$\hat{\boldsymbol{\pi}}(\boldsymbol{\zeta}; \hat{\boldsymbol{W}}_a) = -\frac{1}{2}\boldsymbol{R}^{-1}G^\top(\boldsymbol{\zeta})\nabla_{\boldsymbol{\zeta}}\sigma(\boldsymbol{\zeta})^\top \hat{\boldsymbol{W}}_a \tag{28}$$

In order to find the parameters $\hat{\boldsymbol{W}}_c$ and $\hat{\boldsymbol{W}}_a$, we will in the next section find update laws, based on reinforcement learning, to be able to optimize performance online.

Unfortunately, policy (28) does not account for the saturating constraints, such as the maximum force the actuators of the physical vessel can produce. In order to account for the actuator limitations, we propose a different control policy which uses a saturating function (Doya, 2000) in order to avoid this problem. Using the following cost function:

$$r(\boldsymbol{\zeta}, \boldsymbol{\pi}) = Q(\boldsymbol{\zeta}) + 2\sum_{i=1}^{m} r_i \int_0^{\pi_i} \tanh^{-1}(\xi)d\xi \tag{29}$$

where r_i is the ith entry of the diagonal of $\boldsymbol{R}$, i.e., $\boldsymbol{R} = \text{diag}([r_1, r_2 \dots r_m])$. **Figure 2** shows a comparison of the saturating input cost, and a pure quadratic cost. Performing the same analysis as for the quadratic penalty, we can get the following saturating control law:

$$\boldsymbol{\pi}^*(\boldsymbol{\zeta}) = -\tanh\left(\frac{1}{2}\boldsymbol{R}^{-1}G^\top(\boldsymbol{\zeta})\left(\nabla_{\boldsymbol{\zeta}} V(\boldsymbol{\zeta})\right)\right) \tag{30}$$

Since $\tanh(\cdot)$ saturates at ± 1, this means that the feedback control law $\boldsymbol{\pi}$ will saturate at ± 1, the outputs can then be easily scaled to fit other bounds. It can be shown that since $\tanh(\cdot)$ is a monotonically increasing continuously differentiable function,

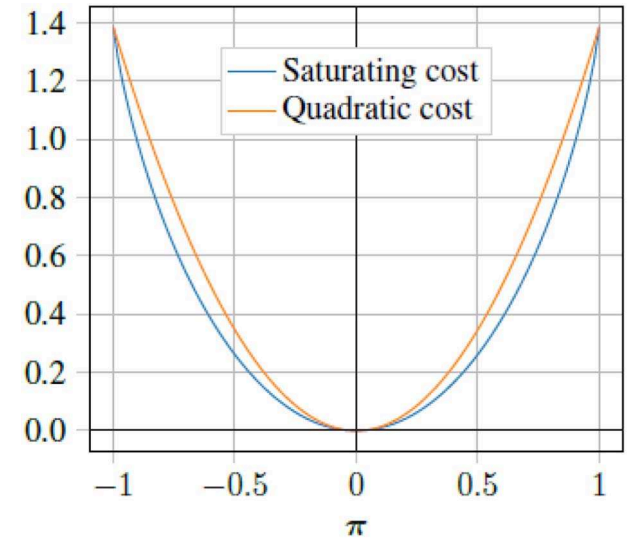


FIGURE 2 | Comparison between saturating cost (29) and quadratic cost (18). Here the quadratic cost is scaled by ln(4).

the control law satisfies the first order necessary conditions, and the second order sufficient conditions of the Hamiltonian minimization condition. This means that if an optimal controller exists the closed form solution is given by (30). Using an approximation we get the following approximate optimal policy

$$\hat{\boldsymbol{\pi}}(\boldsymbol{\zeta};\hat{\boldsymbol{W}}_a) = -\tanh\left(\frac{1}{2}\boldsymbol{R}^{-1}G^\top(\boldsymbol{\zeta})\nabla_{\boldsymbol{\zeta}}\sigma(\boldsymbol{\zeta})^\top\hat{\boldsymbol{W}}_a\right) \tag{31}$$

It should be noted, that while the policy in (31) uses a value function approximation in order to approximate the optimal policy, the parameters $\hat{\boldsymbol{W}}_a$ are not the same as the parameters $\hat{\boldsymbol{W}}_c$ in the value function approximation in (27). In this way we can separate the learning of the policy and value function, this is known as an actor critic method, where the value function is known as the critic, and the policy is known as an actor. The intuitive reason for doing this, is that it allows the critic to learn the value function resulting from the behavior of the policy, and in this way it ca critique the policy. Similarly, the policy or actor, can learn to improve its performance based on the criticism of the critic. How the learning is performed is further discussed in the next section.

2.5. Update Laws

Now that we have expressed the control laws $\boldsymbol{u}_d(\boldsymbol{x}_d;\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\pi}}(\boldsymbol{\zeta};\hat{\boldsymbol{W}}_a)$ and value function $\hat{V}(\boldsymbol{\zeta};\hat{\boldsymbol{W}}_c)$, the challenge becomes finding update laws for the parameters of the system identification $\hat{\boldsymbol{\theta}}$, the critic $\hat{\boldsymbol{W}}_c$ and the actor $\hat{\boldsymbol{W}}_a$. For the model parameters $\hat{\boldsymbol{\theta}}$, we will use methods from system identification and adaptive control, to try to optimize the fit between the parameterized model, and the observed vessel states. For the actor and critic parameters $\hat{\boldsymbol{W}}_a$ and $\hat{\boldsymbol{W}}_c$, we will use model based reinforcement learning to find the parameters that gives the optimal value function, and consequently the optimal feedback control policy.

For the system identification parameters $\hat{\boldsymbol{\theta}}$, the goal is to find the parameters for which the model behaves as similarly as possible to the observed behavior. Running our physical system, and collecting observations $(\dot{\boldsymbol{x}}_i, \boldsymbol{x}_i, \boldsymbol{u}_i) \quad i \in 1,2,\ldots,N$, we can formulate a least squares optimization problem for finding the parameters that minimize the difference between the observed state derivative $\dot{\boldsymbol{x}}_i$ and the parametric model (3) as follows.

$$\boldsymbol{\theta}^* = \arg\min_{\hat{\boldsymbol{\theta}}} \underbrace{\sum_{i=1}^{N}\frac{1}{2}||\dot{\boldsymbol{x}}_i - Y(\boldsymbol{x}_i)\hat{\boldsymbol{\theta}} - f_1(\boldsymbol{x}_i) - g(\boldsymbol{x}_i)\boldsymbol{u}_i||_2^2}_{L(\hat{\boldsymbol{\theta}})}$$

This is a linear least squares optimization problem for which there exists a closed form solution, however we can also solve the problem by performing stochastic gradient decent on the parameters $\hat{\boldsymbol{\theta}}$, as follows:

$$\hat{\boldsymbol{\theta}} \leftarrow \hat{\boldsymbol{\theta}} - \nabla_{\hat{\boldsymbol{\theta}}}L(\hat{\boldsymbol{\theta}})$$

The gradient decent law above, works in discrete iteration, however we can reformulate it as a an ordinary differential equation (ODE). Doing some further changes motivated by the stability analysis of the convergence of the parameter estimates, we get the concurrent learning based approach proposed in Chowdhary and Johnson (2011b) as:

$$\dot{\hat{\boldsymbol{\theta}}}(t) = \boldsymbol{\Gamma}_\theta Y^\top(\boldsymbol{x}(t))\tilde{\boldsymbol{x}}(t) + \frac{k_\theta}{N}\boldsymbol{\Gamma}_\theta\sum_{i=1}^{N}Y^\top(\boldsymbol{x}_i)\left(\dot{\boldsymbol{x}}_i - f_1(\boldsymbol{x}_i) - g(\boldsymbol{x}_i)\boldsymbol{u}_i - Y(\boldsymbol{x}_i)\hat{\boldsymbol{\theta}}\right) \tag{32}$$

where $\boldsymbol{\Gamma}_\theta$ is a parameter weight matrix, and k_θ is a scalar weight factor. Assuming that the prerecorded data is sufficiently rich such that the matrix $\sum_{i=1}^{N}Y^\top(\boldsymbol{x}_i)Y(\boldsymbol{x}_i)$ is full rank, the parameter error can be shown to converge. As the convergence rate of the system identifier is proportional to the minimum singular value of $\sum_{i=1}^{N}Y^\top(\boldsymbol{x}_i)Y(\boldsymbol{x}_i)$, replacing data in the data stack can be done by using a singular value maximizing algorithm (Chowdhary and Johnson, 2011a) in order to get faster convergence. Note, that since we are assuming a sufficiently rich prerecorded data set, we no longer need persistence of excitation (PE), in order to guarantee parameter convergence.

In order to find the update laws for the critic or value function parameters $\hat{\boldsymbol{W}}_c$, we need a way of evaluating the optimality of the value function given the current parameters. For this we look back at the HJB Equation (20) given as:

$$0 = r(\boldsymbol{\zeta},\boldsymbol{\pi}^*(\boldsymbol{\zeta})) + \nabla_{\boldsymbol{\zeta}}V^*(\boldsymbol{\zeta})^\top\left(F(\boldsymbol{\zeta}) + G(\boldsymbol{\zeta})\boldsymbol{\pi}^*(\boldsymbol{\zeta})\right)$$

Substituting the estimates $\hat{V}$ and $\hat{\boldsymbol{\pi}}$ for the optimal value function V^* and optimal policy $\boldsymbol{\pi}$, we can formulate the Bellman error as the error in the HJB equation as follows:

$$\delta(\boldsymbol{\zeta};\hat{\boldsymbol{\theta}},\hat{\boldsymbol{W}}_c,\hat{\boldsymbol{W}}_a) = \underbrace{Q(\boldsymbol{\zeta}) + \hat{\boldsymbol{\pi}}^\top(\boldsymbol{\zeta};\hat{\boldsymbol{W}}_a)\boldsymbol{R}\hat{\boldsymbol{\pi}}(\boldsymbol{\zeta};\hat{\boldsymbol{W}}_a)}_{r(\boldsymbol{\zeta},\hat{\boldsymbol{\pi}}(\boldsymbol{\zeta};\hat{\boldsymbol{W}}_a))} + \nabla_{\boldsymbol{\zeta}}\hat{V}(\boldsymbol{\zeta};\hat{\boldsymbol{W}}_c)^\top\left(F(\boldsymbol{\zeta};\hat{\boldsymbol{\theta}}) + G(\boldsymbol{\zeta})\hat{\boldsymbol{\pi}}(\boldsymbol{\zeta};\hat{\boldsymbol{W}}_a)\right) \tag{33}$$

The Bellman error can intuitively be thought of as the error between the optimal value function under the policy, and the estimates. Since the goal for the value function or critic is to find the parameters $\boldsymbol{W}_c$ that best approximates the value function, a natural choice becomes to find the parameters that minimize the bellman error. With reinforcement learning we can use a data stack of prerecorded state transitions $\boldsymbol{\zeta}_i(t) = [\boldsymbol{x}_i - \boldsymbol{x}_{d,i}, \boldsymbol{x}_{d,i}]^\top \quad i \in 1, 2, \dots N$, to formulate the following optimization problem:

$$\min_{\hat{\boldsymbol{W}}_c} \sum_{i=1}^{N} \frac{1}{2} \delta(\boldsymbol{\zeta}_i; \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{W}}_c, \hat{\boldsymbol{W}}_a)^2$$

This is a non-linear optimization problem, but we may again use a methods like gradient decent in order to iteratively learn parameters that improve the optimization problem given above. Writing the gradient decent in terms of an ODE, and making some changes motivated by a stability analysis (Kamalapurkar et al., 2018). A least-squares update law with forgetting factor (Ioannou and Sun, 2012) can be formulated for the critic as follows:

$$\begin{aligned}\dot{\hat{\boldsymbol{W}}}_c(t) = &-k_{c,1}\boldsymbol{\Gamma}(t)\frac{\boldsymbol{\omega}(\boldsymbol{\zeta}(t),t)}{\rho(\boldsymbol{\zeta}(t),t)}\hat{\delta}(\boldsymbol{\zeta}(t),t) - \frac{k_{c,2}}{N}\boldsymbol{\Gamma}(t) \\ &\sum_{i=1}^{N}\frac{\boldsymbol{\omega}(\boldsymbol{\zeta}_i(t),t)}{\rho_i(\boldsymbol{\zeta}_i(t),t)}\hat{\delta}(\boldsymbol{\zeta}_i(t),t)\end{aligned} \tag{34}$$

$$\dot{\boldsymbol{\Gamma}}(t) = \begin{cases}\beta\boldsymbol{\Gamma}(t) - k_{c,1}\boldsymbol{\Gamma}(t)\frac{\boldsymbol{\omega}(\boldsymbol{\zeta}(t),t)\boldsymbol{\omega}^\top(\boldsymbol{\zeta}(t),t)}{\rho^2(\boldsymbol{\zeta}(t),t)}\boldsymbol{\Gamma}(t) & \text{If } ||\boldsymbol{\Gamma}|| \leq \bar{\boldsymbol{\Gamma}} \\ 0 & \text{Otherwise}\end{cases} \tag{35}$$

In critic update law above $k_{c,1}$ and $k_{c,2}$ are scalar learning rates, while $\boldsymbol{\Gamma}$ is an adaptive weight matrix, and β is a scalar forgetting factor, which controls how previous data samples are discounted. For brevity of notation we used the functions $\boldsymbol{\omega}(\cdot)$, $\rho(\cdot)$, and $\hat{\delta}(\cdot)$ defined as:

$$\begin{aligned}\boldsymbol{\omega}(\boldsymbol{\zeta}, t) &= \nabla_{\boldsymbol{\zeta}}\sigma(\boldsymbol{\zeta})\left(F(\boldsymbol{\zeta}; \hat{\boldsymbol{\theta}}(t)) + G(\boldsymbol{\zeta})\hat{\boldsymbol{\pi}}(\boldsymbol{\zeta}; \hat{\boldsymbol{W}}_a(t))\right) \\ \rho(\boldsymbol{\zeta}, t) &= 1 + \boldsymbol{\omega}^\top(\boldsymbol{\zeta}, t)\boldsymbol{\Gamma}(t)\boldsymbol{\omega}(\boldsymbol{\zeta}, t) \\ \hat{\delta}(\boldsymbol{\zeta}, t) &= \delta(\boldsymbol{\zeta}; \hat{\boldsymbol{\theta}}(t), \hat{\boldsymbol{W}}_c(t), \hat{\boldsymbol{W}}_a(t))\end{aligned}$$

Here, $\boldsymbol{\omega}$ can be considered a regressor vector, while ρ is a normalization factor, and $\hat{\delta}$ the Bellman error.

The actor update law (36) is chosen such that it learns from the critic, while at the same time trying to stay close to the initial control law.

$$\dot{\hat{\boldsymbol{W}}}_a(t) = \text{proj}\left(-k_{a,1}\left(\hat{\boldsymbol{W}}_a(t) - \hat{\boldsymbol{W}}_c(t)\right) - k_{a,2}\left(\hat{\boldsymbol{W}}_a(t) - \boldsymbol{W}_0\right)\right) \tag{36}$$

In the actor update law above, the first term will make the actor parameters follow the critic parameters, while the second term will try to keep the actor parameters close to the initial parameters $\boldsymbol{W}_0$. $k_{a,1}$ and $k_{a,2}$ are scalar learning rates for the two terms. A smooth projection (Ioannou and Sun, 2012) is added such that the actor weights are within a predefined region, for which the control law is stable. Any smooth projection can be chosen, however we chose a projection ensuring the actor weights were bounded within a region of the initial weights $\boldsymbol{W}_0$.

2.6. Stability Analysis

For the system identification parameters $\boldsymbol{\theta}$, we consider the candidate Lyapunov function:

$$V_p(\boldsymbol{x}) = \tilde{\boldsymbol{\theta}}^\top \boldsymbol{\Gamma}_{\boldsymbol{\theta}}^{-1}\tilde{\boldsymbol{\theta}}, \tag{37}$$

where $\tilde{\boldsymbol{\theta}} = \hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*$ is the difference between the predicted and optimal model parameters. Assuming the system is time invariant (including time invariant environmental forces in the NED frame), and given a positive definite weighting matrix $\boldsymbol{\Gamma}_{\boldsymbol{\theta}}$. The time derivative of the candidate Lyapunov function is:

$$\begin{aligned}\dot{V}_P(\boldsymbol{x}) &= 2\tilde{\boldsymbol{\theta}}^\top\boldsymbol{\Gamma}_{\boldsymbol{\theta}}^{-1}\dot{\boldsymbol{\theta}} \\ &= 2\tilde{\boldsymbol{\theta}}^\top\boldsymbol{\Gamma}_{\boldsymbol{\theta}}^{-1}\boldsymbol{\Gamma}_{\boldsymbol{\theta}}Y(\boldsymbol{x})^\top\tilde{\boldsymbol{x}} + \frac{2k_{\boldsymbol{\theta}}}{N}\tilde{\boldsymbol{\theta}}^\top\boldsymbol{\Gamma}_{\boldsymbol{\theta}}^{-1}\boldsymbol{\Gamma}_{\boldsymbol{\theta}}\sum_{i=1}^{N}Y^\top(\boldsymbol{x}_i)\tilde{\boldsymbol{x}}_i\end{aligned}. \tag{38}$$

Using the fact that: $\tilde{\boldsymbol{x}} = \dot{\boldsymbol{x}} - f_1(\boldsymbol{x}) - Y(\boldsymbol{x})\hat{\boldsymbol{\theta}} - g(\boldsymbol{x})\boldsymbol{\tau} = -Y(\boldsymbol{x})\tilde{\boldsymbol{\theta}}$ we get:

$$\dot{V}_P(\boldsymbol{x}) = -2\tilde{\boldsymbol{x}}^\top\tilde{\boldsymbol{x}} - \frac{2k_{\boldsymbol{\theta}}}{N}\tilde{\boldsymbol{\theta}}^\top\sum_{i=1}^{N}\left(Y^\top(\boldsymbol{x}_i)Y(\boldsymbol{x}_i)\right)\tilde{\boldsymbol{\theta}} \leq 0, \tag{39}$$

hence the model error $\tilde{\boldsymbol{x}}$ and parameter error $\tilde{\boldsymbol{\theta}}$ converge exponentially to zero as $t \to \infty$. We can also note that the rate of the parameter convergence is given by the singular values of $\sum_{i=1}^{N}\left(Y^\top(\boldsymbol{x}_i)Y(\boldsymbol{x}_i)\right)$.

For the RL update laws in (34)–(36), it can be shown that under a number of strict assumptions, a system on the form given in (15), with an unconstrained policy, is uniformly ultimately bounded in terms of the error dynamics $\boldsymbol{e}$, as well as the weights and parameters $\boldsymbol{W}_a$, $\boldsymbol{W}_c$, and $\boldsymbol{\theta}$. The stability analysis can be found in Kamalapurkar et al. (2018). For our purposes, we further constrain the parameters $\boldsymbol{W}_a$ of the feedback control law by projecting them into a region close to a known stable initial parameterization. Closed loop stability is important for assurance of the control system, this is further discussed in section 4.

2.7. Reference Model

When generating a reference path, we must ensure that it is sufficiently smooth in order to be able to say something about the convergence to the path. In practice however, we may have a signal which is discrete, defining the desired pose only at certain times. In order to smooth the trajectory we therefor use a reference model, which tracks the discrete reference pose, and generates a continuous reference trajectory pose $\boldsymbol{\eta}_d = [x_d, y_d, \psi_d]^\top$ and velocity vector $\boldsymbol{\nu}_d = [u_d, v_d, r_d]^\top$. For the pose we can make a reference model on the following form:

$$\begin{bmatrix}\dot{\boldsymbol{\eta}}_d \\ \ddot{\boldsymbol{\eta}}_d \\ \dddot{\boldsymbol{\eta}}_d\end{bmatrix} = \begin{bmatrix}0 & \boldsymbol{I} & 0 \\ 0 & 0 & \boldsymbol{I} \\ -\boldsymbol{\Omega}^3 & -(2\boldsymbol{\Delta} + \boldsymbol{I})\boldsymbol{\Omega}^2 & -(2\boldsymbol{\Delta} + \boldsymbol{I})\boldsymbol{\Omega}\end{bmatrix}\begin{bmatrix}\boldsymbol{\eta}_d \\ \dot{\boldsymbol{\eta}}_d \\ \ddot{\boldsymbol{\eta}}_d\end{bmatrix}$$

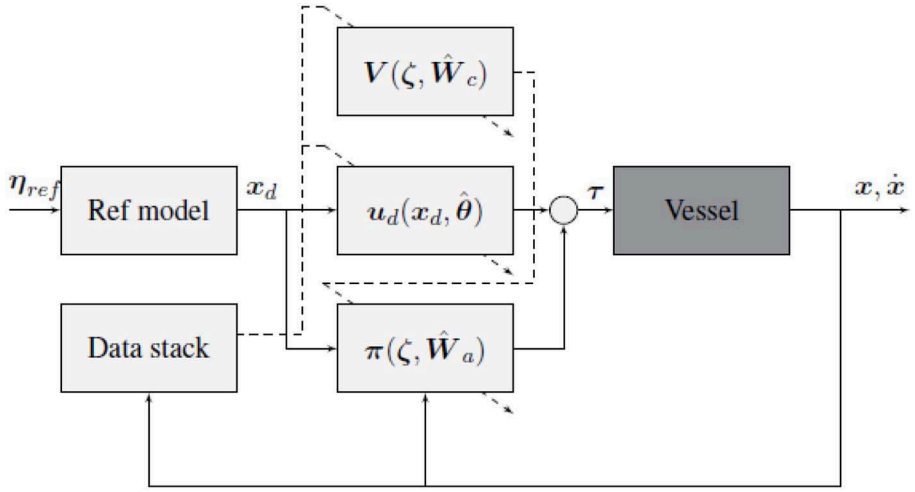


FIGURE 3 | Block diagram of the proposed control scheme, solid lines represent state and control signals, while dashed lines represent adaption signals. Note that the underlying vessel dynamics considered are unknown, and includes thrust allocation and state estimation.

$$+ \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{\Omega}^3 \end{bmatrix} \boldsymbol{\eta}_{ref} \tag{40}$$

Where $\boldsymbol{\Omega} = \text{diag}([\omega_1, \ldots, \omega_n])$ and $\boldsymbol{\Delta} = \text{diag}([\delta_1, \ldots, \delta_n])$. Choosing $\boldsymbol{\Delta} = \boldsymbol{I}$ ensures the reference model is critically damped, while $\boldsymbol{\Omega}$ controls the rate of convergence of the states. We must also generate the velocity vector, however based on the pose, the velocity can be calculated as:

$$\begin{aligned} \boldsymbol{\nu}_d &= \boldsymbol{J}^\top(\boldsymbol{\eta}_d)\dot{\boldsymbol{\eta}}_d \\ \dot{\boldsymbol{\nu}}_d &= -\boldsymbol{S}([0, 0, r_d]^\top)\boldsymbol{J}^\top(\boldsymbol{\eta}_d)\dot{\boldsymbol{\eta}}_d + \boldsymbol{J}^\top(\boldsymbol{\eta}_d)\ddot{\boldsymbol{\eta}}_d \end{aligned} \tag{41}$$

where $-\boldsymbol{S}([0, 0, r_d]^\top)\boldsymbol{J}^\top(\boldsymbol{\eta}_d) = \dot{\boldsymbol{J}}^\top(\boldsymbol{\eta}_d)$, and $\boldsymbol{S}(\boldsymbol{\omega})$ is the skew symmetric matrix:

$$\boldsymbol{S}([\omega_1, \omega_2, \omega_3]^\top) = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}$$

The reason we here use a third order filter for the reference model, is to ensure a smooth pose, velocity, and acceleration, even when a step in the reference is observed. This ensures that the feedforward control for the reference trajectory (16) can track the reference.

A block diagram of the final control structure is given in **Figure 3**. The diagram shows how the controller is split into a feedback control law $\boldsymbol{\pi}$, and a feedforward control law $\boldsymbol{u}_d$. Where the Reference filter is used to generate the pose and velocity reference $\boldsymbol{x}_d$, and the data stack collected from observing vessel transitions, is used to update the parameters of the control laws.

3. EXPERIMENTS

In this section we present the results form simulations, and sea trials on the *ReVolt* test platform (see **Figure 4**), when using the control scheme proposed in the previous section. We will first present the implementation details for the for the control algorithm. After that we will briefly present the experimental platform, before finally presenting the simulation, and sea trial results for varying operational conditions. The experiments include both low speed dynamic positioning, and high speed trajectory tracking.

3.1. Implementation Details

For the implementation the parameter update laws (32), (36), (35), and (34) were implemented with a 4th order Runge-Kutta integration scheme, with a timestep of 0.1 s. Additionally the reference model in (40) and (41) were implemented, also using a 4th order Runge-Kutta scheme, in order to generate the reference trajectory $\boldsymbol{x}_d = [\boldsymbol{\eta}_d, \boldsymbol{\nu}_d]^\top$ and its derivative $\dot{\boldsymbol{x}}_d = h_d = [\dot{\boldsymbol{\eta}}_d, \dot{\boldsymbol{\nu}}_d]^\top$.

For the parameterization of the system identifier, the $\boldsymbol{\theta}$ and $Y(\boldsymbol{x})$ were chosen as in (7) and (9), while for the actor and critic, the parameterization $\sigma(\boldsymbol{\zeta})$ was chosen as the vector of all the second order cross terms of the position and velocity error in the body frame $\boldsymbol{e}^{\text{body}} = [\tilde{\boldsymbol{\eta}}^{\text{body}}, \tilde{\boldsymbol{\nu}}]$ where $\tilde{\boldsymbol{\eta}}^{\text{body}} = \boldsymbol{J}^\top(\boldsymbol{\eta})\tilde{\boldsymbol{\eta}}$, giving the following expression:

$$W\sigma(\boldsymbol{\zeta}) = \sum_{x_i \in \boldsymbol{e}^{\text{body}}} \sum_{x_j \in \boldsymbol{e}^{\text{body}}} w_{i,j} x_i x_j \tag{42}$$

The reason that we use the error in the body frame, is the assumption that the cost is invariant to rotations when in the body frame, as this is the same frame the dynamics of the system are given in. The initial conditions for the actor and critic weights were chosen such that they matched the continuous time algebraic Riccati equation for a simplified linear model of the vessel.

For the control law, the constrained closed form controller (30) was used. And the output was scaled to fit the max thrust and torque $\bar{\boldsymbol{\tau}} = \frac{1}{\sqrt{3}}[50.0, 20.0, 32.0]^\top$ the vessel is able to

FIGURE 4 | *ReVolt* test platform courtesy of DNV GL.

produce. While $[50.0, 20.0, 32.0]^\top$ is the max force the vessel is able to produce in each direction individually, due to the coupling between thrusters, we assume the maximum thrust in any given coupled direction can be approximated by the an ellipse with axis lengths 50.0, 20.0, and 32.0. Since the proposed method only allows us to constrain thrust in each individual direction, we use the largest inner approximation of the ellipse as our thrust bound, giving the max thrust and torque as $\bar{\boldsymbol{\tau}}$ given above. It should be noted, that while this constrains the thrust, we can still not guarantee that the vessel is able to produce the desired amount of thrust as $\bar{\boldsymbol{\tau}}$ is only an inner approximation of the elliptic approximation, whereas the true thrust bound may be much more complex. It should also be noted that using the inner approximation $\bar{\boldsymbol{\tau}}$ as a bound, means we are not able to fully utilize the full thrust that the vessel has to offer. One way of solving these issues would be to include the thrust allocation as part of the problem formulation, however this is beyond the scope of this paper. It should also be noted that the desired thrust vector includes both the path tracking control law and drift correction $\boldsymbol{\tau}_{\text{Thrust}} = \boldsymbol{u}_d(\boldsymbol{x}_d; \hat{\boldsymbol{\theta}}) + \hat{\boldsymbol{\pi}}(\boldsymbol{\zeta}; \hat{\boldsymbol{W}}_a)$ where the saturation is only considered in the drift controller and not the path tracking control law. This means the desired path should be generated in a way that satisfies the thrust constraints.

For the state cost function $Q(\boldsymbol{\zeta})$ a quadratic cost on the form $Q(\boldsymbol{\zeta}) = [\tilde{\boldsymbol{\eta}}^{\text{body}}, \tilde{\boldsymbol{\nu}}]^\top \boldsymbol{Q} [\tilde{\boldsymbol{\eta}}^{\text{body}}, \tilde{\boldsymbol{\nu}}]$ was chosen, where $\boldsymbol{Q}$ is a positive definite weight matrix. Given as:

$$\boldsymbol{Q} = \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 10.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 10.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 10.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 10.0 \end{bmatrix}$$

The weight matrix is given as a mostly diagonal matrix, with a small cross term between the position error in sway direction, and heading error. The cross term is added in order to encourage the vessel to travel in the surge direction when there is a large position error, as this is the most efficient direction of travel.

The data stack that was used consisted of 100 samples, and a singular value maximization scheme was implemented in order to increase the convergence rate. Using a purely singular value maximization based data selection scheme, while giving good performance on a stationary system, does not work for time varying system, and hence does not allow for estimating the slowly varying environmental forces. In order to account for this, weighting of the singular value maximization, and data sample age was used in order to save recent samples with high singular values.

TABLE 1 | *ReVolt* hardware and software specifications.

Onboard computer:	Tank-720
Sensors:	Xsens MTI-G-710 IMU
	Vector VS330 GNSS Receiver
Software:	Linux Ubuntu LTS 16.04
	ROS Kinetic Kame

3.2. Experimental Platform

The *ReVolt*, shown in **Figure 4**, is a 1:20 scale model of a autonomous concept vessel developed by DNV GL in collaboration with NTNU. The model is 3 m long, 0.72 m wide, and weighs 257 kg. *ReVolt* has a top speed of 2 knots (∼1 m/s) with a total combined engine power of 360 W. The thrust configuration is given as in **Figure 1**, with two identical stern thrusters, and one slightly less powerful bow thruster, all of which are fully rotatable azimuth thrusters, and are controlled by an optimization based thrust allocation (TA) algorithm. The vessel state is estimated using a non-linear observer consisting of an Extended Kalman Filter (EKF), and combines measurements from a Global Navigation Satellite System (GNSS) with Real-Time Kinematic (RTK) correction data, on board accelerometer, gyroscope, and compass. This provides accurate heading and position down to $\pm 0.2^\circ$ and ± 1 cm. A description of the *ReVolt* hardware and software is given in **Table 1**.

While the physical vessel was used for the sea trials, a high fidelity Digital Twin of *ReVolt*, developed by DNV GL, was used for simulation. The Digital Twin is based on a full 6DOF model, with parameters identified through tow-tank experiments, as well as frequency domain analysis of a 3D model of the vessel hull. The Digital Twin allowed for rapidly testing how the proposed control scheme performed under ideal conditions, as well as under different sea states, ocean currents and wind conditions.

3.3. Simulations and Sea Trials

In order to test the proposed control scheme, a number of experiments were devised. As the control scheme was build to

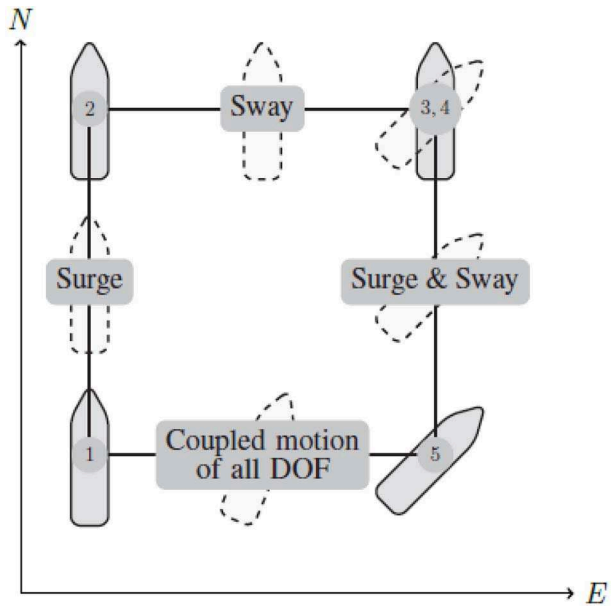


FIGURE 5 | Four corner DP test, for testing trajectory tracking in individual, and coupled degrees of freedom.

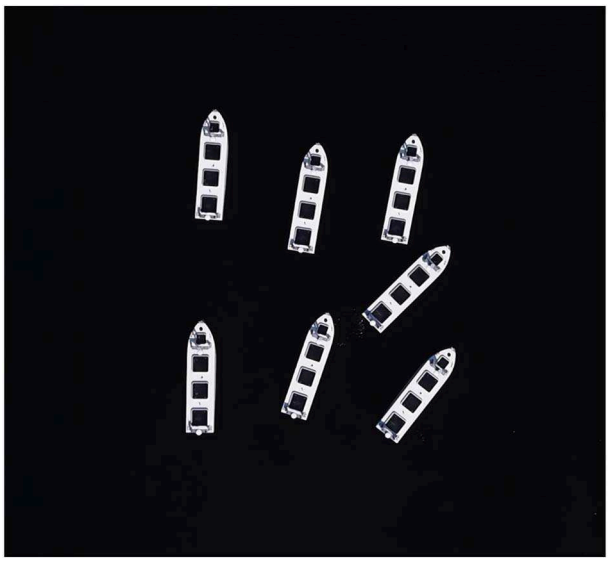


FIGURE 6 | Time-lapse drone photo of four corner DP test. It should be noted that the time-lapse above is of an early test, where errors in the navigation system resulted in poor performance.

be able to handle both high speed and low speed maneuvering, we wanted to test both, by doing low speed Dynamic Positioning (DP), as well as higher speed path tracking.

3.3.1. Dynamic Positioning (DP)

In order to test the dynamic positioning capabilities of the control method, the four corner test seen in **Figures 5**, **6** is used. The four corner DP test is used, as it shows the tracking capabilities of the vessel for individual degrees of freedom, as well as the coupled motion of all degrees of freedom, it is also worth noting that the vessel returns to the initial pose, meaning the test can easily be repeated. The four corner test starts with the vessel pointing north 0°, then performs the following commands:

1. Change position l meters due north, and come to a complete stop. This tests the surge motion of the vessel.
2. Change position l meters due east, and come to a complete stop. This tests the sway motion of the vessel.
3. Change heading 45°, and come to a complete stop. This tests the yaw motion of the vessel.
4. Change position l meters due south, and come to a complete stop. This tests the coupled surge and sway motion of the vessel.
5. Change position l meters due west, and heading to 0° and come to a complete stop. This tests coupled motion of all degrees of freedom.

For the box test we chose the box side length l to be 5 m, and the reference path was generated by linearly interpolating the pose between the commands, with 55 s to execute each command and a 5 s pause between commands in order for the reference filter to catch up to the reference, and ensure that the vessel comes to a stop. The reference poses used for the experiments are given in **Table 2**.

TABLE 2 | Reference pose for four corner DP test, note that the reference that was used was a linear interpolation of the poses in the table.

Time [s]	0	55	60	115	120	175	180	235	240	295	300
x_r [m]	0	5	5	5	5	5	5	0	0	0	0
y_r [m]	0	0	0	5	5	5	5	5	5	0	0
ψ_r [deg]	0	0	0	0	0	45	45	45	45	0	0

In order to evaluate the performance of the dynamic positioning, The Integral Absolute Error (IAE) given in (43) was used.

$$IAE(t) = \int_0^t \sqrt{(\bar{\boldsymbol{\eta}} - \bar{\boldsymbol{\eta}}_d)^\top (\bar{\boldsymbol{\eta}} - \bar{\boldsymbol{\eta}}_d)} dt \tag{43}$$

Where $\bar{\boldsymbol{\eta}}$ and $\bar{\boldsymbol{\eta}}_d$ are the normalized pose vectors, normalized between ± 5 m in north and east direction, and $\pm 50°$ in heading, giving the following.

$$\bar{\boldsymbol{\eta}} = \left[\frac{x}{5}, \frac{y}{5}, \frac{\psi}{50}\right]^\top, \quad \bar{\boldsymbol{\eta}}_d = \left[\frac{x_d}{5}, \frac{y_d}{5}, \frac{\psi_d}{50}\right]^\top$$

Running the proposed control scheme in simulations on the Digital Twin of the *ReVolt* vessel, we got the trajectory and errors seen in **Figure 7**. For the same test performed on the physical vessel during the sea trials, we got the trajectory and errors seen in **Figure 8**. The IAE for the tests are shown in **Figure 9**.

3.3.2. Path Tracking

For both straight line path tracking and curved path tracking, the way-points in **Table 3** were used to generate a linearly, and

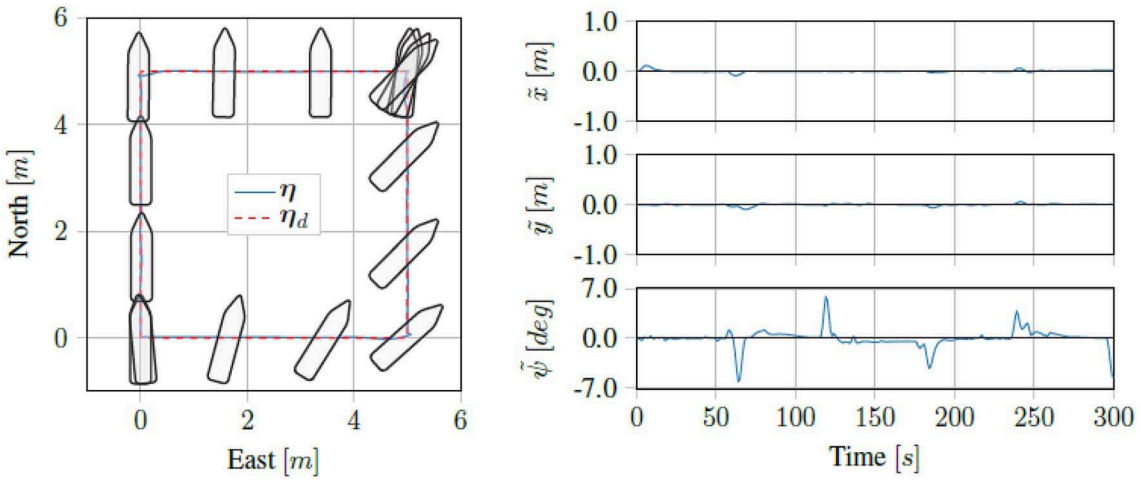


FIGURE 7 | Simulation results for four corner DP tests.

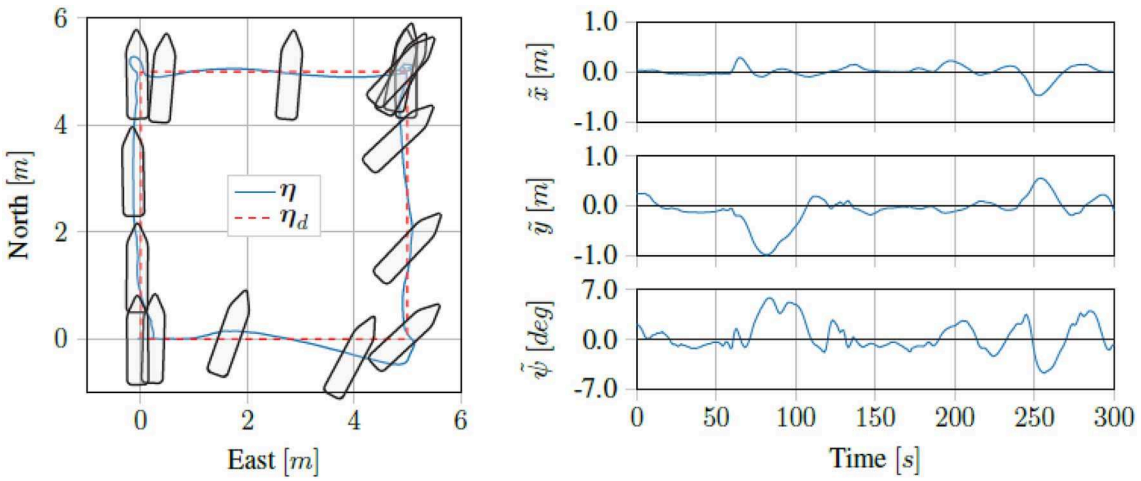


FIGURE 8 | Sea trial results for four corner DP tests.

quadratically interpolated path, respectively. For the heading, the path direction was used to generate the desired heading, giving the following reference heading.

$$\psi_r = \text{atan2}(\dot{y}_r, \dot{x}_r) \tag{44}$$

In order to encourage the vessel to converge to path in the surge direction, a small cross term was added in the state cost function $Q(\boldsymbol{\zeta})$ between the heading error, and the position error in the surge direction of the body frame. The key insight here, is that the for large errors in surge, this term will encourage the vessel to turn the bow toward the desired position, meaning the vessel is encouraged to travel in the surge direction, which is the most efficient direction of travel, due to the design of the hull. For our implementation, where pose error is given in the body frame of the vessel, and the state penalty is given as a quadratic function $Q(\boldsymbol{\zeta}) = \boldsymbol{\zeta}^\top \boldsymbol{Q}\boldsymbol{\zeta}$, this penalty is added by simply adding a term to the off-diagonals of $\boldsymbol{Q}$ corresponding to the cross terms between position error in the y direction, and the heading error.

Running the straight line path tracking on the Digital Twin of the *ReVolt* vessel we got the results seen in **Figure 10**. Running the same tests on the physical vessel, we got the results seen in **Figure 11**. As we can see, the proposed control scheme is able to follow the path quite well.

3.4. Results

Based on the results, the proposed method seems to work very well, in both simulations and the physical platform. While the simulator has been designed to perform as closely as possible to the physical platform, there are slight discrepancies that may explain the performance drop. The main factors of the performance drop is however most likely due to the measurement and observation noise that is present on the physical vessel. While the RTK GNSS is able to give a good

measurement for the pose of the vessel, the estimated vessel velocities that the algorithm is dependent on become very inaccurate, especially at low speeds when the signal to noise ration becomes small. Another error source is likely the thruster dynamics. While the algorithm above assumes the desired thrust is produced immediately, in reality producing the desired thrust vector takes time, as the thrust allocation involves rotating the thrusters to a given angle, as well as spinning up to a desired motor RPM. An additional source of error may also have been a vertical stabilizer, which had recently been added to the vessel between the two rear thrusters, but had not been taken into account in the thrust allocation algorithm. Overall, the results are quite good, especially considering the size of the vessel, the relatively low thrust capability, and the precision to which the maneuvers are performed, even under the uncertainty created by the sensor noise, and environmental forces.

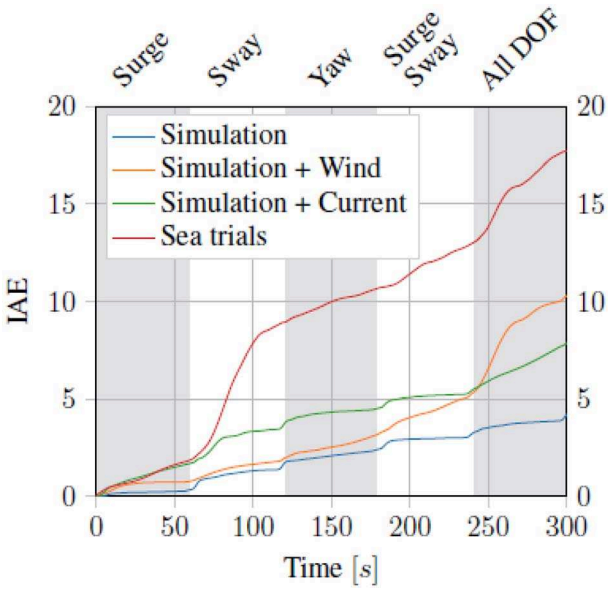


FIGURE 9 | Integral absolute error (IAE) for the dynamic positioning task, the gray and white bands mark the different commands/phases of the four corner test.

4. ASSURANCE OF RL-BASED CONTROLLERS

Assurance is the structured collection of evidence supporting claims and arguments that a system is safe or fit for its intended purpose. Assurance is required to develop trustworthy systems and solutions for use in real-world applications. Principles of assurance can be found in any certification or verification framework, where claims and arguments most often can be considered requirements of verification, while evidence is the result from this verification. Two types of verification are used: (1) *Product verification*, which performs direct verification of the developed product or system and produces *primary evidence*; (2) *process verification*, which performs verification of some part of the development process and produces *circumstantial evidence*. Using established verification frameworks applied to conventional marine control systems, experience has shown what requirements and evidence are most important when verifying these conventional control systems. With novel technology, such as data-driven methods, the verification requirements and evidence that is needed for assurance are still unknown, as they pose a new set of challenges when assuring the system.

Data-driven approaches are not new, but with increasing computational power and abundance of data there has been an increasing interest in these methods. Within control theory, the field of system identification has been a key part of control engineering for many years (Åström et al., 1965; Ho and Kálmán, 1966), and data-driven modeling for control purposes has been practiced since. Such models are typically based on the physical properties that govern the system, and hence the parameters estimated by such methods may reflect measurable properties of the system, thus providing benchmarks for verification. However, most models represent the physical system only within certain operational limits, e.g., weather or sea states, or for certain vessel speed ranges, which restricts the validity of the models accordingly.

TABLE 3 | Reference pose for the straight line path and curved path, note that the reference that was used was for straight line path tracking was a linear interpolation of the poses, and the reference pose for the curved path, was a quadratic interpolation of the poses.

Time [s]	**0**	**100**	**200**	**300**
x_r [m]	0	50	100	150
y_r [m]	0	0	50	50

Contrary to the more static nature of classical data-driven approaches, where tuning the control parameters relies on offline estimation of the model parameters, in this paper the key difference is that model based-RL is used for online tuning of both the vessel model (including an estimation of unknown disturbances) and the control policy parameters. The vessel model and the control policy are based on proven methods used in the maritime industry for vessel station keeping and guidance, but there are still some key issues that must be considered. For instance, the control policy is highly dependent on an instantaneously valid vessel model, which in turn means the behavior of the vessel is highly dependant on the validity of the learned model. Both the vessel model and the control policy parameters are all continuously learned, but it is critical that all allowed parameter combinations give a sufficiently safe behavior. The proposed control scheme in this paper continuously learns and updates the parameters in order to optimize the tracking behavior. In terms of safety, the main concern is whether the learned model and policy parameters lead to a safe and acceptable behavior. Verifying this in a setting where the parameters are learned online is still an open problem.

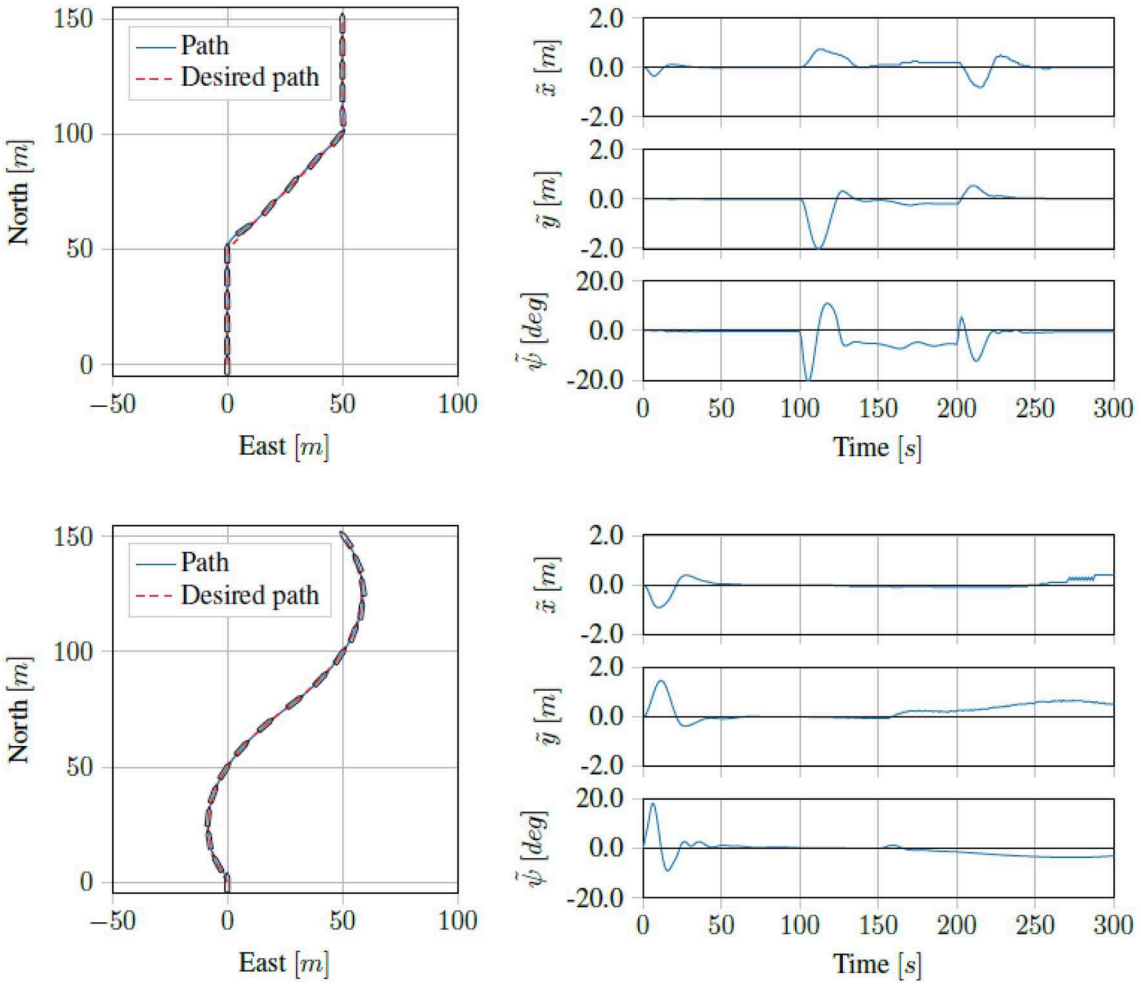


FIGURE 10 | Simulation results for straight line and curved path tracking.

Amodei et al. (2016) discusses five basic concrete problem areas related to RL and safety, which must be taken into account for any application of RL.

1. Avoiding reward hacking: the first problem, is that of hacking or gaming the cost function. For the tracking problem in this paper, a positive definite quadratic penalty on the error dynamics is used. From control theory these methods are known to converge to the origin, i.e., where the error is zero. This means the intended behavior is guaranteed when the policy converges to the optimal policy.
2. Avoiding negative side effects: the second problem of avoiding negative side effects, is similar to the first, but addresses the issue of choosing the cost function such that the optimal policy does not give bad or unintended behavior. For the method proposed in this paper, making such guarantees is quite difficult, as tuning the parameters of the quadratic cost function will still have an effect on the vessel behavior when converging to the origin. One example of this is that we typically want the vessel to approach the path head on if we have a large deviation between the position we are at, and the desired position. Tuning the parameters of cost function in order to get this behavior is not trivial.
3. Scalable oversight: this pertains to how we can ensure that the RL agent respects aspects of the objective that are encountered infrequently. In terms of the trajectory tracking problem, the environment is quite limited, and the objective is clearly defined, hence the problem of scalable oversight is of limited relevance to the work presented in this paper.
4. Safe exploration: exploration is necessary in order to improve performance, but bears risk, and thus performing exploration in a safe manner is not trivial. Safe exploration also encompasses the evaluation of the quality of the training data that is gathered. For a real world application, this means accounting for faulty hardware, and noisy measurements, which may lead to problematic training data. For the method proposed in this paper, where the system is learning continuously online, the problem of safe exploration and learning is highly relevant. Some measures are taken, such

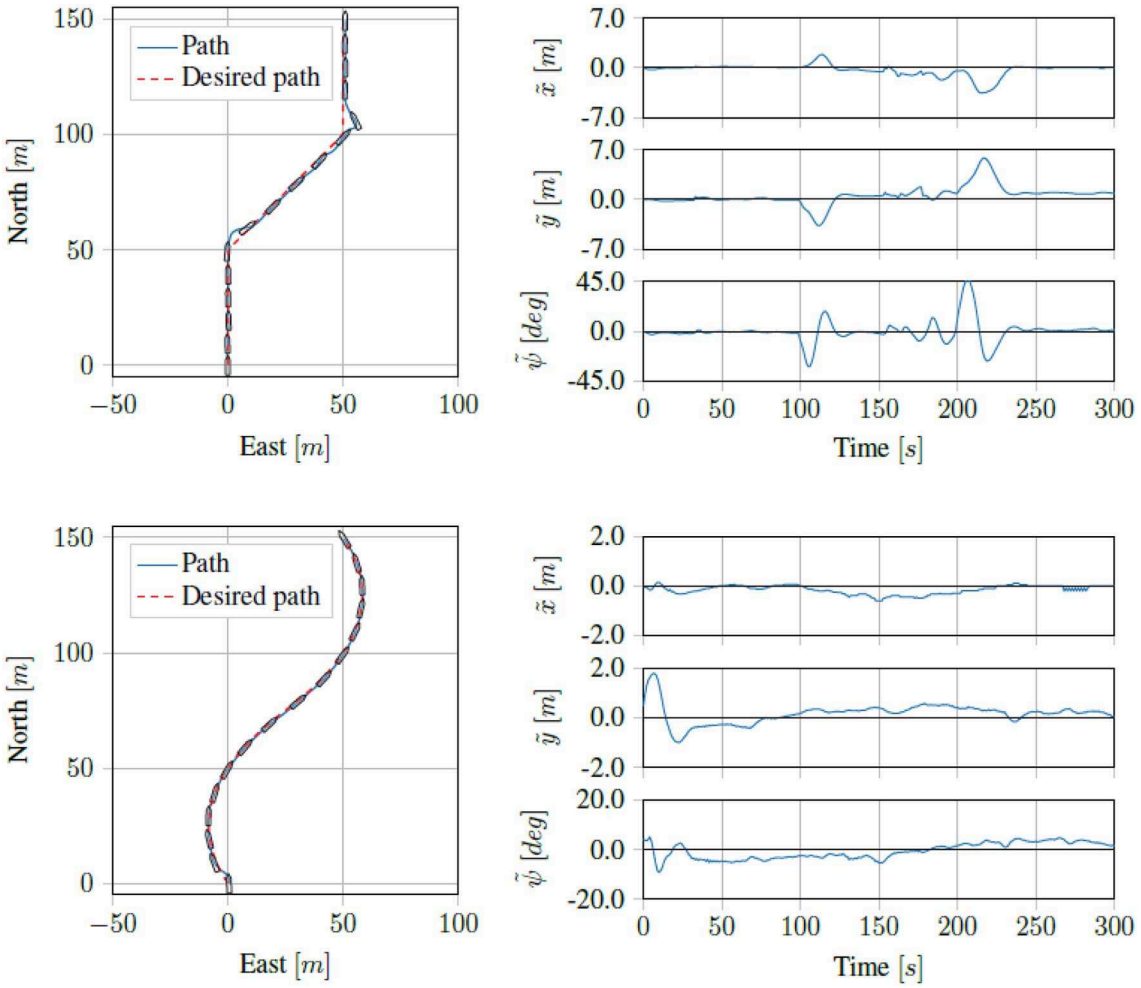


FIGURE 11 | Sea trial results for straight line and curved path tracking.

as using batches of training data and restricting the values that the policy parameterization may take. However, these measures only serve to mitigate potential problems, hence safe exploration and learning is still an open problem.

5. Robustness to distributional shift: this refers to how we can ensure the agent is robust to changes in the operating environment. For the proposed method, this is mostly solved by continuously learning online, which ensures that the agent learns the distributional shift when the environment changes. However, as discussed in the fourth problem of safe exploration, learning online complicates the matter of ensuring data quality.

In order to produce the evidence needed for verification of data driven methods, there are two main approaches, namely *scenario based verification*, and *theoretical verification*. *Scenario based verification* would be to conduct extensive testing in representative scenarios, which in practice would mean simulation-based testing as this would be the only feasible online solution. More limited real testing should also be used, but targeted toward validating the simulation accuracy. Many RL solutions are in practice not viable without simulation-based training or development and this would mean the same tool can be utilized both for testing, and offline training. The challenge in this case would be to induce a representative set of scenarios to prove the safety or validity of the solution, and such scenario selection is an open question for testing AI or systems operating in complex environments in general. The second approach *theoretical verification*, would be to impose constraints on the RL algorithm in order to avoid unwanted behavior. This would entail combining methods from control theory, a physical or mathematical understanding of the system, and experience or insights of the control scheme, in order to express and implement various constraints on the learnable model and policy parameters. This may not conceptually be a new approach since similar methods already exist, but this approach is difficult to use in practice, as finding parameter constraints that ensure safe operations is non-trivial.

In conclusion, an assurance framework for technologies, such as the one presented in this paper is an open research question.

However, one can with confidence state that it should include both process and product verification, i.e., considering not only what is developed, but also how it is developed. This would mean that adequate development and assurance processes should be developed, including verification methods that can produce the required evidence both for efficient development, as well as assurance. In addition, novel data-driven methods should be combined with prior knowledge, verified solutions and proven physical or mathematical relations (Eldevik, 2018). This, in order to be able to explain the behavior, and in turn guarantee that the methods are safe and fit for the intended purpose.

5. CONCLUSION

The proposed method performed very well in all three tested tracking scenarios both in simulations and in sea trials. The method is also versatile, as using it on different vessels only requires knowledge of the inertia matrix, with the update laws providing a tool for learning the other model parameters, and a control policy. For future work, it may be interesting to improve and update the thrust allocation algorithm to get a smaller error between produced and desired thrust, and investigate whether this results in better accuracy. Alternatively, feedback from the thrusters can be used to get better estimates of the thrust vector for use in the data stack, and model estimation. For the value function, polynomial basis functions were used, and the estimator was linear in the parameters, which leads to a limited estimation capability. Deep learning methods could lead to more accurate value function approximation, albeit at the expense of transparency and interpretability. Procedures for implementing online assurance would add great value to current research practices. One possible way to do this is by using a new parameterization of the control law once it has been verified, either by simulation, or by constraining the parameterization to a set of parameters that is known to be safe.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation, to any qualified researcher.

AUTHOR CONTRIBUTIONS

AM and AL conceived the presented idea. AM developed the theory and performed the simulations, while AM, TP, and AL carried out the experiments. JG and TP contributed with the assurance section from a classification society perspective. AM, AL, and SG discussed the results and methods. All authors contributed to the final manuscript.

REFERENCES

Åström, K. J., Bohlin, T., and Wensmark, S. (1965). *Automatic Construction of Linear Stochastic Dynamic Models for Stationary Industrial Processes With Random Disturbances Using Operating Records*. Technical Paper TP 18, IBM Nordic Laboratory, Stockholm.

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. (2016). Concrete problems in AI safety. *arxiv [Preprint] arXiv:1606.06565*.

Andrychowicz, M., Baker, B., Chociej, M., Jozefowicz, R., McGrew, B., Pachocki, J., et al. (2018). Learning dexterous in-hand manipulation. *arXiv* 1808.00177. doi: 10.1177/0278364919887447

Bertsekas, D. P. (2019). *Reinforcement Learning and Optimal Control*. Belmont, MA: Athena Scientific.

Cheng, Y., and Zhang, W. (2018). Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels. *Neurocomputing* 272, 63–73. doi: 10.1016/j.neucom.2017.06.066

Chowdhary, G., and Johnson, E. (2011a). "A singular value maximizing data recording algorithm for concurrent learning," in *Proceedings of the 2011 American Control Conference* (San Francisco, CA: IEEE), 3547–3552. doi: 10.1109/ACC.2011.5991481

Chowdhary, G. V., and Johnson, E. N. (2011b). Theory and flight-test validation of a concurrent-learning adaptive controller. *J. Guid. Control Dyn.* 34, 592–607. doi: 10.2514/1.46866

Dai, S.-L., Wang, C., and Luo, F. (2012). Identification and learning control of ocean surface ship using neural networks. *IEEE Trans. Ind. Inform.* 8, 801–810. doi: 10.1109/TII.2012.2205584

Do, K. D. (2016). Global path-following control of underactuated ships under deterministic and stochastic sea loads. *Robotica* 34, 2566–2591. doi: 10.1017/S0263574715000211

Doya, K. (2000). Reinforcement learning in continuous time and space. *Neural Comput.* 12, 219–245. doi: 10.1162/089976600300015961

Eldevik, S. (2018). *AI + Safety*. Available online at: https://ai-and-safety.dnvgl.com/

Fossen, T. I. (1994). *Guidance and Control of Ocean Vehicles*. New York, NY: John Wiley & Sons Inc.

Fossen, T. I. (2000). A survey on nonlinear ship control: from theory to practice. *IFAC Proc. Vol.* 33, 1–16. doi: 10.1016/S1474-6670(17)37044-1

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

Fossen, T. I., Sagatun, S. I., and Sørensen, A. J. (1996). Identification of dynamically positioned ships. *Control Eng. Pract.* 4, 369–376. doi: 10.1016/0967-0661(96)00014-7

Hasegawa, K., Kouzuki, A., Muramatsu, T., Komine, H., and Watabe, Y. (1989). Ship auto-navigation fuzzy expert system (safes). *J. Soc. Naval Archit. Jpn.* 1989, 445–452. doi: 10.2534/jjasnaoe1968.1989.166_445

Ho, B., and Kálmán, R. E. (1966). Effective construction of linear state-variable models from input/output functions. *AT-Automatisierungstechnik* 14, 545–548. doi: 10.1524/auto.1966.14.112.545

Ioannou, P. A., and Sun, J. (2012). *Robust Adaptive Control*. Mineola, NY: Courier Corporation.

Kallstrom, C. G. (1982). *Identification and Adaptive Control Applied to Ship Steering*. Technical report, Department of Automatic Control, Lund Institute of Technology (LTH), Lund.

Källström, C. G., and Åström, K. J. (1981). Experiences of system identification applied to ship steering. *Automatica* 17, 187–198. doi: 10.1016/0005-1098(81)90094-7

Kamalapurkar, R., Walters, P., Rosenfeld, J., and Dixon, W. (2018). *Reinforcement Learning for Optimal Feedback Control: A Lyapunov-Based Approach*. Springer.

Katebi, M., Grimble, M., and Zhang, Y. (1997). H-∞ robust control design for dynamic ship positioning. *IEEE Proc. Control Theory Appl.* 144, 110–120. doi: 10.1049/ip-cta:19971030

Konda, V. R., and Tsitsiklis, J. N. (2000). "Actor-critic algorithms," in *Advances in Neural Information Processing Systems*, eds S. A. Solla, T. K. Leen, and K. Müller (Denver, CO: MIT Press), 1008–1014.

Lekkas, A. M., and Fossen, T. I. (2014). "Trajectory tracking and ocean current estimation for marine underactuated vehicles," in *2014 IEEE Conference on Control Applications (CCA)* (Juan Les Antibes: IEEE), 905–910. doi: 10.1109/CCA.2014.6981451

Liberzon, D. (2011). *Calculus of Variations and Optimal Control Theory: A Concise Introduction*. Princeton, NJ; Oxford: Princeton University Press.

Martinsen, A. B., and Lekkas, A. M. (2018a). "Curved path following with deep reinforcement learning: results from three vessel models," in *OCEANS 2018 MTS/IEEE* (Charleston, SC: IEEE), 1–8. doi: 10.1109/OCEANS.2018.8604829

Martinsen, A. B., and Lekkas, A. M. (2018b). "Straight-path following for underactuated marine vessels using deep reinforcement learning," in *11th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles (CAMS)* (Opatija). doi: 10.1016/j.ifacol.2018.09.502

McGookin, E. W., Murray-Smith, D. J., Li, Y., and Fossen, T. I. (2000). Ship steering control system optimisation using genetic algorithms. *Control Eng. Pract*. 8, 429–443. doi: 10.1016/S0967-0661(99)00159-8

Mišković, N., Vukić, Z., Bibuli, M., Bruzzone, G., and Caccia, M. (2011). Fast in-field identification of unmanned marine vehicles. *J. Field Robot*. 28, 101–120. doi: 10.1002/rob.20374

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., et al. (2013). Playing Atari with deep reinforcement learning. *arxiv [Preprint] arXiv:1312.5602*.

Pettersen, K. Y., and Egeland, O. (1996). "Exponential stabilization of an underactuated surface vessel," in *Proceedings of 35th IEEE Conference on Decision and Control*, Vol. 1 (Kobe: IEEE), 967–972. doi: 10.1109/CDC.1996.574602

Pham Tuyen, L., Layek, A., Vien, N., and Chung, T. (2017). "Deep reinforcement learning algorithms for steering an underactuated ship," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)* (Daegu: IEEE), 602–607. doi: 10.1109/MFI.2017.8170388

Shen, H., and Guo, C. (2016). "Path-following control of underactuated ships using actor-critic reinforcement learning with mlp neural networks," in *Sixth International Conference on Information Science and Technology (ICIST)* (Dalian: IEEE), 317–321. doi: 10.1109/ICIST.2016.7483431

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature* 529:484. doi: 10.1038/nature16961

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., et al. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arxiv [Preprint] arXiv:1712.01815*.

Soetanto, D., Lapierre, L., and Pascoal, A. (2003). "Adaptive, non-singular path-following control of dynamic wheeled robots," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, Vol. 2 (IEEE), 1765–1770. doi: 10.1109/CDC.2003.1272868

Sørensen, A. J. (2011). A survey of dynamic positioning control systems. *Annu. Rev. Control* 35, 123–136. doi: 10.1016/j.arcontrol.2011.03.008

Sutton, R., Roberts, G., and Taylor, S. (1997). Tuning fuzzy ship autopilots using artificial neural networks. *Trans. Inst. Meas. Control* 19, 94–106. doi: 10.1177/014233129701900204

Sutton, R. S., and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Walters, P., Kamalapurkar, R., Voight, F., Schwartz, E. M., and Dixon, W. E. (2018). Online approximate optimal station keeping of a marine craft in the presence of an irrotational current. *IEEE Trans. Robot*. 34, 486–496. doi: 10.1109/TRO.2018.2791600

Wang, N., Qian, C., Sun, J.-C., and Liu, Y.-C. (2015). Adaptive robust finite-time trajectory tracking control of fully actuated marine surface vehicles. *IEEE Trans. Control Syst. Technol.* 24, 1454–1462. doi: 10.1109/TCST.2015.2496585

Wang, N., Su, S.-F., Yin, J., Zheng, Z., and Er, M. J. (2017). Global asymptotic model-free trajectory-independent tracking control of an uncertain marine vehicle: an adaptive universe-based fuzzy control approach. *IEEE Trans Fuzzy Syst*. 26, 1613–1625. doi: 10.1109/TFUZZ.2017.2737405

Yu, R., Shi, Z., Huang, C., Li, T., and Ma, Q. (2017). "Deep reinforcement learning based optimal trajectory tracking control of autonomous underwater vehicle," in *36th Chinese Control Conference (CCC)* (Dalian: IEEE), 4958–4965. doi: 10.23919/ChiCC.2017.8028138

Zhang, L., Qiao, L., Chen, J., and Zhang, W. (2016). "Neural-network-based reinforcement learning control for path following of underactuated ships," in *35th Chinese Control Conference (CCC)* (Chengdu: IEEE), 5786–5791. doi: 10.1109/ChiCC.2016.7554262

Conflict of Interest: JG and TP were employed by the company DNV GL.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Robotics and AI

TECHNOLOGY AND CODE
published: 28 May 2020
doi: 10.3389/frobt.2020.00070

Trustable Environmental Monitoring by Means of Sensors Networks on Swarming Autonomous Marine Vessels and Distributed Ledger Technology

Ivan Berman[1], Enrica Zereik[2], Aleksandr Kapitonov[1], Fabio Bonsignorio[3], Alisher Khassanov[4], Aziza Oripova[5], Sergei Lonshakov[4] and Vitaly Bulatov[6]*

[1] Faculty of Control Systems and Robotics, ITMO University, Saint Petersburg, Russia, [2] Institute of Marine Engineering, Italian National Research Council, Genova, Italy, [3] Heron Robots, Genova, Italy, [4] Airalab, Tolyatti, Russia, [5] Faculty of Food Biotechnologies and Engineering, ITMO University, Saint Petersburg, Russia, [6] M2M Economy, Inc. ("Merklebot"), San Francisco, CA, United States

OPEN ACCESS

Edited by:
Sheri Marina Markose, University of Essex, United Kingdom

Reviewed by:
Alan Gregory Millard, University of Lincoln, United Kingdom
Luis J. Manso, Aston University, United Kingdom

***Correspondence:**
Ivan Berman
iaberman@itmo.ru

Specialty section:
This article was submitted to Computational Intelligence in Robotics, a section of the journal Frontiers in Robotics and AI

Received: *12 December 2019*
Accepted: *29 April 2020*
Published: *28 May 2020*

Citation:
Berman I, Zereik E, Kapitonov A, Bonsignorio F, Khassanov A, Oripova A, Lonshakov S and Bulatov V (2020) Trustable Environmental Monitoring by Means of Sensors Networks on Swarming Autonomous Marine Vessels and Distributed Ledger Technology. Front. Robot. AI 7:70. doi: 10.3389/frobt.2020.00070

The article describes a highly trustable environmental monitoring system employing a small scalable swarm of small-sized marine vessels equipped with compact sensors and intended for the monitoring of water resources and infrastructures. The technological foundation of the process which guarantees that any third party can not alter the samples taken by the robot swarm is based on the Robonomics platform. This platform provides encrypted decentralized technologies based on distributed ledger tools, and market mechanisms for organizing the work of heterogeneous multi-vendor cyber-physical systems when automated economical transactions are needed. A small swarm of robots follows the autonomous ship, which is in charge of maintaining the secure transactions. The swarm implements a version of Reynolds' Boids model based on the Belief Space Planning approach. The main contributions of our work consist of: (1) the deployment of a secure sample certification and logging platform based on the blockchain with a small-sized swarm of autonomous vessels performing maneuvers to measure chemical parameters of water in automatic mode; (2) the coordination of a leader-follower framework for the small platoon of robots by means of a Reynolds' Boids model based on a Belief Space Planning approach. In addition, the article describes the process of measuring the chemical parameters of water by using sensors located on the vessels. Both technology testing on experimental vessel and environmental measurements are detailed. The results have been obtained through real world experiments of an autonomous vessel, which was integrated as the "leader" into a mixed reality simulation of a swarm of simulated smaller vessels.The design of the experimental vessel physically deployed in the Volga river to demonstrate the practical viability of the proposed methods is shortly described.

Keywords: unmanned surface vessel, robonomics, environmental monitoring, belief space planning, boids model, blockchain, mobile sensors, water quality

1. INTRODUCTION

Water resources are crucial for the maintenance of human life. Natural water is an exhaustible, partially renewable resource. Fresh water is used both for drinking water supply and in industry, agriculture, transport—in almost all human activities. Depending on the desired usage, the requirements for water chemical composition and physical properties may be different.

Worldwide water consumption has been increasing at about 1% per year since the 1980s (World Water Assessment Programme, 2019), and it is due to the growing demand for water in developing countries, where population is still increasing, to the acceleration of socio-economic development and to the fact that consumption patterns began to evolve (Alharsha et al., 2018; World Water Assessment Programme, 2019) in a similar way to those of old and new industrialization countries. Agriculture (irrigation, livestock, and aquaculture) is the largest consumer of water, it accounts for 69% of the world's annual water withdrawal. The next places are occupied by industry (19%) and household (12%) (World Water Assessment Programme, 2019). It is estimated that the global demand for water will continue to grow at the same pace until 2050, which will lead to an increase of 20–30% above the current level of water usage due to the development of the industrial and domestic sectors (Burek et al., 2016) of the emerging countries. Thus, the impact of human activities on the world state of water resources will further grow.

The problem of water pollution will become more and more important, turning it into one of the largest potential disasters for humanity in the coming century[1]. The irresponsible and ill-conceived approach to industry, urbanization, agriculture and environmental management, which humankind has adhered to in the recent past, the lack of adequate measures to prevent and eliminate polluting factors, as well as the lack and weakness of the mechanisms for bringing to justice the violators of environmental pollution, have led to the fact that water resources began to decline rapidly.

The UN General Assembly announced (Food and Agriculture Organization of the United Nations, 2011) that more than 1 billion (according to other estimates such as World Water Assessment Programme, 2019, more than 2 billions) people in the world suffer from a lack of safe water for drinking and household needs. Although the global average water deficit is only 11%, in 31 countries the water deficit ranges from 25% (the minimum threshold for water deficit) to 70%, and in 22 countries it exceeds 70% (United Nations Publications, 2018).

The main issues affecting the quality of natural waters (World Health Organization, 2017) include many different points:

- Infection with pathogens is an important factor in high morbidity and mortality from gastrointestinal diseases (Soprani et al., 2017). It is directly dependent on the population density and the level of its socio-economic development. Pollution by pathogens is not fully controlled even in developed countries.
- Contamination with organics (Cai et al., 2019), which enter the water in a dissolved or suspended form, mainly with sewage drains or unregulated household drains. Due to the oxygen dissolved in water coming from the atmosphere because of the turbulent nature of the flow, the rivers have a significant self-purification ability. However, when the supply of organics begins to exceed self-purification capability, water pollution progressively increases. Nowadays about 80% of polluted water resources are dumped back into oceans, rivers, and lakes[2]. In addition, the oxygen content in water is inversely proportional to its temperature; therefore, climatic conditions also play an unfavorable role in reducing the self-purification capability of rivers.
- Acidification is an anthropogenic natural process due to the increasing acidic reaction of the environment (Shi et al., 2016). It is accepted that natural waters are in a state of acidification if the pH is equal to or less than 5.0. Acidification is a consequence of dry and wet acidic deposition, the main components of which are aerosols consisting of sulfur and nitrogen oxides and ammonia, which, when interacting with water, form acids. This leads to a reduction or disappearance of crustacean, fish, insect, algae and zooplankton populations. The reproductive functions of aquatic organisms are also slowing down.
- Eutrophication enhances the biological productivity of water bodies due to the accumulation of biogenic elements in the water (Leaf, 2018). Excessive intake of nitrogen and phosphorus compounds (the main source of which is agriculture and household wastewater) leads to enhanced growth of aquatic plants, especially microscopic algae, which then result in the removal of large amounts of oxygen dissolved in water. This leads to negative consequences: reduction of fish populations, blocking of water intakes and spillways, deterioration of water quality.
- Agricultural fertilizers lead to an increase in the concentration of nitrates (Bouraoui and Grizzetti, 2014). Up to 15% of the initial mass of fertilizers goes into water bodies, mainly in groundwater. Excessive nitrate levels in drinking water can cause health problems, especially blood disorders in children and the risk of cancer in adults.
- Heavy metal pollution (Akpor et al., 2014): small but hazardous concentrations enter the global water supply from wastewater or from industrial waste landfills. Many heavy elements, such as lead, mercury, zinc, chromium, cobalt are toxic to both natural flora and fauna, and to humans.

As a consequence of the increasing risks and issues briefly described above, global water resources need careful control and monitoring. Preventing microbial and chemical contamination of a water source is the first stage of protection against contaminated drinking water and other public health concerns. For governmental and public environmental monitoring

[1]Woodford, C. (2019). *Water Pollution: An Introduction*. Available online at: https://www.explainthatstuff.com/waterpollution.html (accessed October 15, 2019).

[2]Denchak, M. (2018). Water Pollution: Everything You Need to Know. https://www.nrdc.org/stories/water-pollution-everything-you-need-know. [accessed October 15, 2019]

services, this task involves significant costs, the state-of-the-art hardware and software, and the work of highly qualified personnel who regularly maintain environmental monitoring tools.

Modern monitoring systems for water bodies consist of ground-based (stationary observations at hydrological and expeditionary posts) and remote (aviation and satellite) observation methods (World Health Organization, 2017; Sachse et al., 2018). They are also divided into contactless and contact observation methods; at the place of measurement—on portable, transportable, and laboratory; on data processing technology—on manual, automated and automatic procedures. In most cases, the information received from them is presented in a different format, even in terms of use within a single environmental organization, and is not integrated into a single information system (Hajdari, 2015).

Rapidity of data collection is of particular importance, especially for quick response to environmental changes in case of technical accidents and natural disasters whether they are or not of human origin (Wang et al., 2015). For example, oil can originate pollution episodes at all stages of production, transportation, processing until the final stages of consumption and disposal of related products. Tens of petroleum spills over 7 tons occur annually[3], a lot of oil gets into the water due to leakage from pipelines, railways, oil-tankers, storage facilities.

Quick and cost effective ways of registering and logging pollution data are also needed, because in many cases it is necessary to determine the liabilities of the parties legally responsible for pollution episodes (Shimshack, 2014). For example, in most cases of noxious and unlawful waste disposal into waters produced by industrial enterprises or other entities, the analysis of water characteristics is performed manually by experts, often after complaints from citizens (Sebastian et al., 2018). The relative cost and lacking effectiveness of monitoring activities impair the processes of mitigation and control of human originated pollution.

One more example: the problem of eutrophication is often aggravated, among other things, by the unsatisfactory conditions of municipal and industrial wastewater treatment plants; in particular, this is especially true for reservoirs (Assemany et al., 2019). In order to monitor and control the purification infrastructures, it is necessary to monitor the content of nutrients causing abnormal coloration of waters, therefore an effective and reliable way to audit the infrastructure is required.

Another important issue is citizens' confidence in the monitoring systems of state and public environmental organizations (Alkhelaiwi and Grigoras, 2015). Official data, especially in developing countries, are often either insufficient or of dubious quality. Environmental experts point out that this is due to the obsolescence of the instrumental measurement base, the low financing of the environmental monitoring activities, political motives, and lobbying of the interests of polluting companies. The presence of a transparent system of monitoring the ecological state of water resources, in which the data obtained are verified and available for verification by every citizen, will raise the level of civil society engagement in the environmental conservation and contribute to reduce the skepticism about the need to finance this area and prevent the spread of environmental misconceptions (Arias et al., 2016).

As a consequence, an ideal system for monitoring the state of water resources should:

- Be cost-effective;
- Be small;
- Collect as much environmental data as possible;
- Have a high level of automation to minimize human influence;
- Be easily deployable, flexible, and scalable.

Today compact sensory systems are commercially available (for example, from companies like Libelium, Vaisala, Bosh) and are capable of measuring many physical and chemical indicators. They are able to provide researchers with quick results on environmental measurements, and such results are automatically sent to a secure data repository.

The work described in this paper explores the usage of a swarm of mobile platforms for the monitoring of the quality of water resources, capable of performing water quality measurements in automatic mode with minimal human participation (Shafi et al., 2018; Wang et al., 2018a,b). Autonomous water platforms, the so-called Unmanned Surface Vessels or Vehicles (USV), are nowadays hugely exploited, and many of them are commercially available, such as the PowerDolphin[4], Texys Marine[5], and CAT-Surveyor[6] projects. These systems are small-sized, can be non-volatile and incomparably cheaper than the previous generation of equipment.

In this article, we describe the principles of certified collection of environmental samples using a small vessel equipped with sensors and connected to distributed registry for storing the collected data; such vessel leads a group of smaller USVs taking samples of the environment. We have performed our tests of the "leader" vessel at field, while the follower swarm behavior has been assessed by means of mixed reality simulation.

Robotics swarming, consisting in cooperative multi-agent autonomous systems, has a great potential in many field domains, and is especially suitable for marine environment monitoring, lending more flexibility and scalability to the overall system, as well as resulting in a greater effectiveness. Actual robotics swarming is not yet so widespread and exploited, due to many technical challenges that have to be addressed and solved, as very recently surveyed in Arnold et al. (2019)—even if such work is more focused on aerial vehicles, many concepts can be easily extended to other robotics application fields. This work highlights a set of features that a swarm must own, among which the ability to move, cooperate and/or react to occurring events. Another very recent work describes the deployment and exploitation of a heterogeneous robotic swarm for marine monitoring Lončar et al. (2019); however, authors

[3]https://www.statista.com/statistics/268553/number-of-oil-spills-by-oil-tankers-since-1970/

[4]https://www.powervision.me/en/product/powerdolphin

[5]https://www.texysmarine.com/

[6]https://www.subsea-tech.com/cat-surveyor/

present a multi-agent system with many different robots that have very limited motion capabilities: they adopt the concept of distributed communicating sensor networks, rather than implementing a robotic swarm as defined above (robots are mainly still in the neighborhood of their deployment points and do not cooperate to gather information). The topic of marine environmental monitoring by means of a robotics swarm is addressed also in Duarte et al. (2016), where the issue of scalability is faced through large-scale simulation. Finally, the issue of aggregation is addressed in a less recent work (Soysal and Sahin, 2005), proposing probabilistic aggregation strategies to obtain cooperative global behavior for the swarm, by combining basic individual behaviors. This approach traces back to the classical Reynolds flocking model described in Reynolds (1987), and here integrated in a Belief Space Planning strategy.

The core idea of the technological solution we propose is to merge a distributed ledger secure storage of the data with an effective sensor swarm. The goal is to have a shared control network system integrating a bio-inspired swarm management into a secure distributed ledger. Previously some authors have proposed either pure blockchain solutions (Kapitonov et al., 2019) or swarm solutions (Strobel et al., 2018) where each node is a blockchain node (something that makes the swarm operation slow).

We propose a hierarchical approach which is novel and puts together the benefits of both. The application of distributed ledger technologies in robotics applications is an emerging field, Castelló Ferrer (2018). A number of workshops have been organized in the latest few years, see footnotes[7,8], and new publications are planned by the people working in this promising area, see footnote[9]. This work is related to two specific problems: the applications of blockchain technologies in swarm robotics and the application of blockchain technologies to networks robotics. In Strobel et al. (2018) the distributed ledger technology is used with the objective of guaranteeing the security of the swarm: the distributed secure ledger of the B-C ensure that "alien robots" do not join the network. This approach on the one hand allows to exploit the adaptivity of swarm intelligence and its capability to manage large numbers of robots, on the other hand applying blockchain algorithms to all the nodes in a swarm is at present difficult to put in practice for practical performance reasons. In Kapitonov et al. (2019) blockchain technologies are proposed as a tool to manage general ranging from Smart Cities to Citizen Science in multi vendor heterogeneous environments. However, the network robotics approach does not allow to efficiently and effectively manage a very large number of robots or intelligent devices and sensors.

In this work we do some steps to bridge the two approaches. Our platform shares the security features of distributed ledger technologies with the adaptivity and scalability of the swarm robotics approach. There are several benefits coming from the exploitation of a robotics swarm in a formation around the leader vessel:

- Many different measurements (of the same physical parameters) along the chosen path in only one mission. Having a cluster of measures around one point rather than just one is helpful in building up a measurement map more complete and reliable;
- Having more than one vehicle carrying sensors guarantees more robustness to the mission completion in terms of possible failures either of the vehicle or of the sensor;
- The follower USVs can be heterogeneous, i.e., equipped with sensors measuring different physical parameters in the current considered point along the chosen path.

Summarizing, a swarm operating in formation around the leader can provide more measurements (both in terms of quantity and in terms of different types), in a more reliable way, being equal the required mission time. The swarm approach provides higher levels of adaptivity and scalability with respect to other network robotics approaches with a limited pre programming and computational effort. Implementing it by means of an inherently stochastic planning motion method like BSP makes the solution comparatively robust. Moreover, in the foreseen overall system, the leader is the only one (having Internet access) in charge of storing the measures on the trustable platform; hence the vehicles around can share with it their gathered data, to be then integrated and aggregated by the leader to build up a map of the surveyed area.

This paper is organized as follows. Section 2 presents the idea of certified sampling based on distributed ledger technology and the related implementation based on the Robonomics platform. In section 3 the architecture and processes of such platform are detailed. Section 4 describes the "leader" vessel: its design, equipment, sensors, software, and the algorithms governing it. In section 5, the experimental results of water quality measurements obtained by the vessel are presented and analyzed. In section 6, we describe our Belief Space Planning implementation of Reynolds' Boids Model. Section 7 presents the results of the experimental tests of swarm behaviors in mixed reality simulation. Conclusions and discussion on future work are drawn in the last section.

2. CERTIFIED SAMPLING

The term "certified sampling" refers to the quality control of a biological or chemical sample for compliance with certain official criteria (Schreiber et al., 2006). The criteria, as a rule, are established by authorized legal entities or public authorities in the form of standards, regulations and sometimes laws, and the verification procedure itself is carried out by accredited specialists according to established rules and complying with established procedures. In theory, precisely this rigor of the sampling activities and the authority of the bodies performing them should foster public confidence in the obtained data. However, due to the high bureaucratization, the high cost of carrying out inspections and the concentration of control over them in the hands of legal bodies that are not always transparent, the public confidence in environmental data is decreasing (Jacques, 2016).

[7]International Workshop on Blockchain Technologies for Robotic Systems (BCT4ROS'2019), https://researchers.pagesperso-orange.fr/bct4ros2019/.

[8]Proceedings of the First Symposium on Blockchain and Robotics, 2018, Ledger Journal, http://ledgerjournal.org/.

[9]Decentralized AI and Robotics Using Blockchain Technology, Frontiers Research Topic, https://www.frontiersin.org/research-topics/10378/decentralized-ai-and-robotics-using-blockchain-technology.

In addition, the disclosure of serious corporate frauds (such as diesel emissions scandal[10]) have contributed to impair citizen trust in the "official" data.

The main requirements of certified sampling are standardized and registered execution procedures (on which all participants in the process agree) and confidence in the data received. The result of the audit is a formal certificate, whose format and content are legally defined, so that the change or forgery of the sample is often prosecuted as a crime. We propose a solution to reduce and mitigate the issues coming from bureaucracy, complexity and cost of inspections, as well as corruption.

The practical usage of mobile and stationary cyber-physical systems (CPS) that take environmental samples in automatic mode is growing (Mois et al., 2016). However, until recently, a general and reliable mechanism for automatically logging the actions of these devices was not proposed. Such a mechanism should not only save data, that device receives and sends. It should also guarantee their immutability and prevent collected data forgery. To this aim, we need to protect both the device executable code and their datasets (Bijani and Robertson, 2014).

In our setting the leader vessel acts as the centralization hub of the swarms samples and take care of the secure logging of the data by interacting with the Ethereum blockchain. The swarm collective behaviors are governed by our implementation of the BSP based flocking model. This allows to have at the same time a secure and certified log of the samples and an efficient management of the robot swarm, see 6. In this section we focus on the secure certification processes.

The foundations for the development of a distributed certification mechanisms have been actively studied by researchers and developers over the past decade: the blockchain technology. A blockchain provides a sequential chain of blocks built according to certain rules, and protected by cryptographic algorithms (Xu et al., 2016; Castelló Ferrer, 2018). The technology allows to create a peer-to-peer decentralized network of many nodes that exchange secure transactions. The main point of the technology is that it prevents data from being changed in transactions, but at the same time preserves the publicity of relations among the nodes in the network. Such a network of nodes will be protected from incorrect or malicious changes caused by a faulty data source, being this last either one of the nodes or an external attacker. Moreover, the blockchain allows to implement smart contracts—generated by a software program, placed on the blockchain with a guarantee of its implementation (Christidis and Devetsikiotis, 2016). Thanks to smart contracts, the operation of autonomous devices can be organized so that the program logic is executed only under the particular conditions specified in the transactions, and the data can be stored in an practically immutable registry.

The Ethereum blockchain (Dannen, 2017) smart contracts have been used to create the secure Robonomics platform[11] and for the interaction of various autonomous devices.

The central idea of the Robonomics platform is to organize the relationship between nodes in the form of offer and demand requests and to negotiate between them by using an internal currency.

The platform is based on a number of pillars:

- The Robot Operating System (ROS) (Koubaa, 2018). Due to the heterogeneity in term of system architectures and middleware of robotic systems and Internet of Things devices system interfaces and software, we used ROS to facilitate their interoperability and coordination. ROS makes it easier, in our case, to integrate new types of devices into a common network system architecture.
- The InterPlanetary File System (IPFS)[12]. We integrated IPFS into the platform to store the large amount of information that devices collect during their operations.
- Liability Market. This is the part of the platform in charge of matching Offer and Demand among the nodes of the system. The Liability Market transactions are organized through IPFS messages.
- Liability Contracts. They are Ethereum smart contracts made by cyber-physical systems (robots, intelligent agents, IoT devices and other artificial agents) with each other or with humans.
- Tokens. Since the interaction among the agents is based on market mechanisms, we need a "currency" in the network, and this is provided by "Tokens."

The advantages of this approach for trustable measurements by USV are:

- This approach ensures that the data is collected and sent by specific USV.
- Once the data has been collected and sent to the network, it cannot be changed.
- The data remains open for verification by third party.
- There is an exact reference to the time when the digital signature of the data was sent to the distributed registry.
- Easy scaling of system. Thanks to integration with ROS, it's quite simple to add additional agents to the system, no matter what structure and mechanism of measurements.
- The unity of machine-to-machine and human-to-machine interaction in the context of ordering measurements.

These advantages enable the creation of a trusted communication environment which could potentially grow into a united ecological information system with a high level of trust. However, we should indicate the limitations of this approach:

- The need for internet access. IPFS and Ethereum require that the device always has access to these networks. The fourth generation mobile access network is sufficient for the stable operation of the system.
- Sending information to the blockchain requires monetary funds (since blockchain miners and some nodes of Robonomics network require a fee for their work).

[10]Schiermeier, Q. (2015). The science behind the Volkswagen emissions scandal. Available online at: https://www.nature.com/news/the-science-behind-the-volkswagen-emissions-scandal-1.18426

[11]Lonshakov, S., Krupenkin A., Kapitonov, A., Radchenkoet, E., Khassanov, A., and Starostin, A. (2018). *The Robonomics Platform Has Been Designed for the Integration of Cyber Physical Systems into Human Economy*. Available online at: https://robonomics.network/robonomics_white_paper_en.pdf (accessed October 15, 2019).

[12]https://ipfs.io/

It should also be noted that at the measurement stage, it is possible to physically intervene in the process. This problem can be solved by developing the proper USV design. For example, USV can record its activity in a photo or video and similarly save this information into Robonomics network.

In previous work of some of the authors, successful operation of several unmanned aerial vehicles on the Robonomics platform was demonstrated (Kapitonov et al., 2019); in the current research we are extending the approach to monitor the state of the environment, in particular of water quality. In this article we describe the operation of a small swarm of marine surface autonomous vehicles equipped with sensors in charge of performing water quality measuring, guided by a "leader" autonomous marine vessel managing a distributed secure registry of the samples by means of the Robonomics platform.

3. THE ROBONOMICS PLATFORM

The platform is based on a middleware software called AIRA (Autonomous Intelligent Robot Agent)[13] which enables the connection of ROS-based systems and devices with Ethereum and IPFS.

In general, following entities are required in the Robonomics network to complete a task:

- The Promisee, a node that assigns a task. This can be either a human or an artificial agent.
- The Promisor, a node that performs a task. It can be associated with physical or software tasks.
- The Liability Market, as mentioned above, is a platform for offer and demand messages published via IPFS.
- The Provider, a node that monitors the messages of the Liability Market and matches an offer and a demand for a small fee. The Providers of a "message channel" are managed by the "Lighthouse"—a special smart contract, which performs a transaction when the Provider establishes a market match between the Promisor and the Promisee.
- New liabilities in the form of smart contracts are concluded in the Ethereum network only via the Provider.
- The AIRA client is required for the Promisor to have access to the Liability Market and to get information about the task.
- The Validator (not shown), an optional node which may be specified in the demand message. If it is specified, only the Validator node (for a fee) can finalize liability contracts after checking them.

The task performing process is organized in three stages, as depicted in **Figure 1**. At the first stage (Negotiation), the Promisee sends a demand message to the Liability Market in IPFS. In the following, the main message fields are reported (refer to **Table 1**).

- The "model" field—it uniquely identifies the cyber-physical system.
- The "objective" field—it contains dynamic parameters specific for the operation to be performed (as arguments for functions in programming languages). This is an IPFS hash indicating the rosbag file. Such rosbag file contains ROS-topics and their details.
- The "token" field—the token used to pay for the service of the cyber-physical system.
- The "cost" field—the cost in tokens from previous "token" field.
- The "lighthouse" field—the name of the Lighthouse which manages the desired Providers.
- The "validator" field—the address of the Validator.
- The "validatorFee" field—validator fee for its work.
- The "deadline" field—block number until which the demand is valid.
- The "sender" and the "signature" fields—they are automatically filled and identify the Promisee.

The demand goes to the Provider, and then to the agent that is able to perform the task. CPS can accept the offer or submit a counter offer, in the same way, the Promisee can send counter-demand. This stage ends when offer/demand messages are equal in the *model*, *objective*, *token*, *cost*, and *lighthouse* fields. In this case, a new smart contract is created in the Ethereum Blockchain by the Provider.

When the Ethereum smart contract is created the requested task enters the Execution stage, during which the AIRA software waits for a message confirming the liability creation and passes the fields with the information to the agent. The CPS subscribes to the indicated ROS topics to obtain the necessary information; after this, the task execution begins.

In the last stage (Finalization), the CPS notifies the AIRA software of the completion of the requested task, and AIRA collects all the operation logs into the Result message. This message is then sent to the IPFS. If a Validator has been specified, it first checks the Result message and validate it. At the end the Provider sees the notification in the Result message and register the final transaction to Ethereum.

4. DESCRIPTION OF THE AUTONOMOUS MARINE VESSELS

The vessels are solar-powered water surface catamaran, with two hulls and a MPPT (Maximum Power Point Tracker) energy harvesting system with a lithium-ion battery.

Each vessel is equipped with:

- Two Bluerobotics T200 thrusters[14];
- A waterproof case for electronics;
- A battery pack: LiFePO4 300 W·h, ∼2,000 charge cycles;
- Photovoltaic panels: 200 W max, 30-60 W while cloudy weather.

The vessel main characteristics are listed in the following. **Figure 2** illustrates its design.

[13]https://github.com/airalab/aira

[14]https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-thruster/

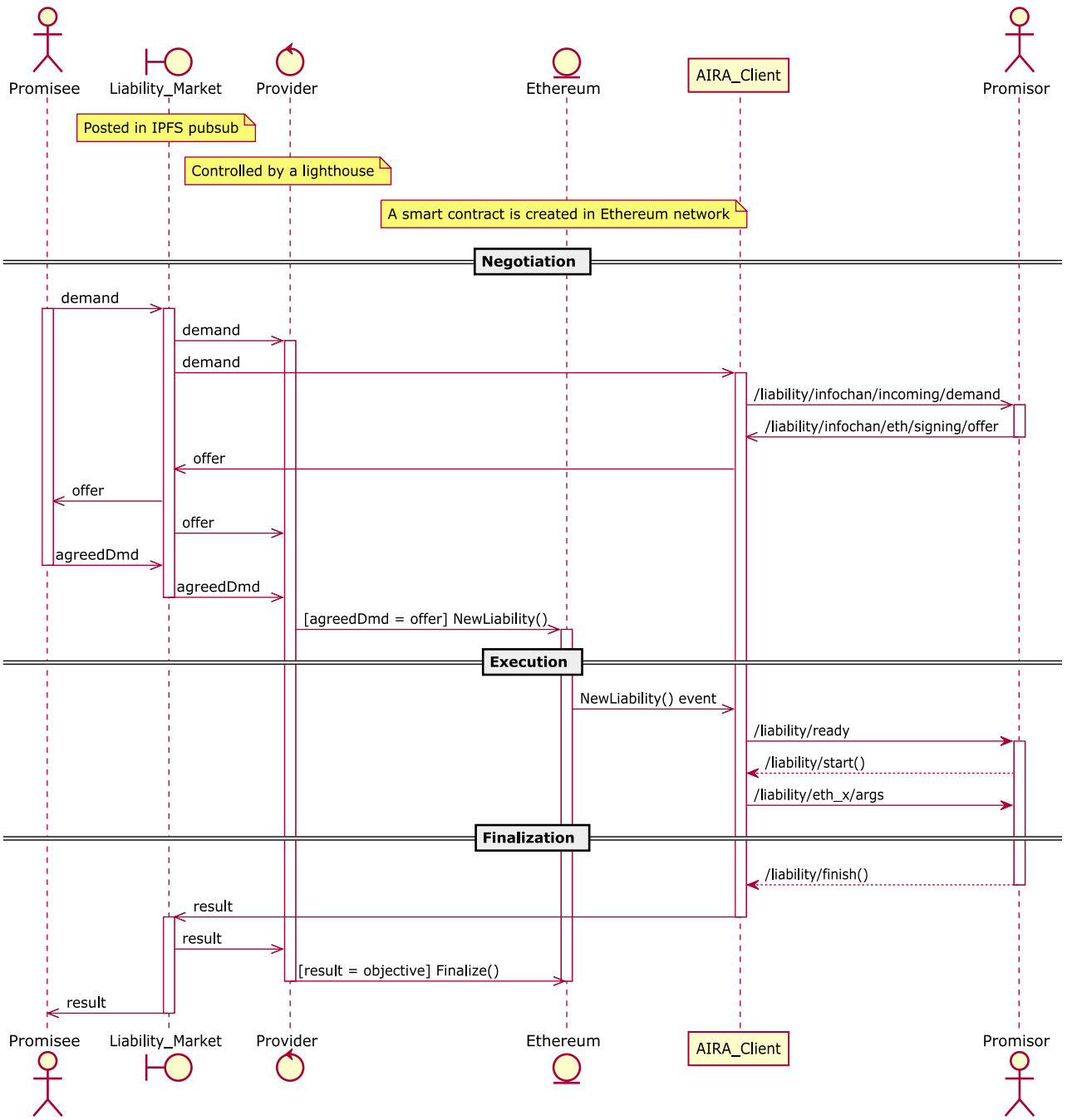


FIGURE 1 | Nodes interaction in Robonomics network. Offer and demand messages between different nodes may vary in format (JSON in IPFS, ROS messages in AIRA client), but the information they provide remains the same.

- Max velocity: 5–7 km/h.
- Cruise velocity: 2–3 km/h.
- Dimensions: 1,200 × 1,200 × 500 mm.
- Weight: up to 38 kg (depending on the number of sensors).

Photovoltaic panels provide enough energy for daytime operations on cruise velocity, as well as 3–5 h of operations without light. The navigation and motion control system is based on Pixhawk and PX4 autopilot[15]. The choice of this controller is due to the fact that Pixhawk and PX4 are among the most popular tools for the navigation controller that natively supported operations with two motors without a steering device. The on-board computational unit

[15]https://pixhawk.org/

TABLE 1 | An example of a typical "Demand" message (without a Validator).

Field	Type	Description	Example
model	ipfs_common/Multihash	CPS behavioral model identifier	QmYb81uDNDHCnu9EZtYV 4eoBDKRBAwJeNy1LT3p5Zb c357
objective	ipfs_common/Multihash	CPS behavioral model parameters in rosbag file	Qmea8XkcSXmvLDKES7D88 6pfimsWh9Vjh1ZJsoHm9MW G4C
token	ethereum_common/Address	Operational token address	0xC02aaA39b223FE8D0A0e5 C4F27eAD9083C756Cc2
cost	ethereum_common/UInt256	CPS behavioral model implementation cost	0,1 WETH
lighthouse	ethereum_common/Address	Lighthouse address	0xa1b60ED40E5A68184b3ce4 f7bEf31521A57eD2dB1
validator	ethereum_common/Address	Validator address	0x00000000000000000000 0000000000000000000 (No)
validatorFee	ethereum_common/UInt256	Validator commission	0
deadline	ethereum_common/UInt256	Deadline block number	6393332
sender	ethereum_common/Address	Message sender address	0xB819d9BC2E665962BCa62 Cd859059875BABB134c
signature	std_msgs/UInt8[]	Sender's digital signature	—

The offer message looks similar. Part of the fields is an IPFS hash on which significant information about the mission is located, another part refers to the node addresses in Ethereum network. Promisee exchanges such messages with Promisor until matching is reached between them. Between the layers, the messages change in the format (in IPFS it is JSON, in the AIRA client — ROS message), but the information remains the same.

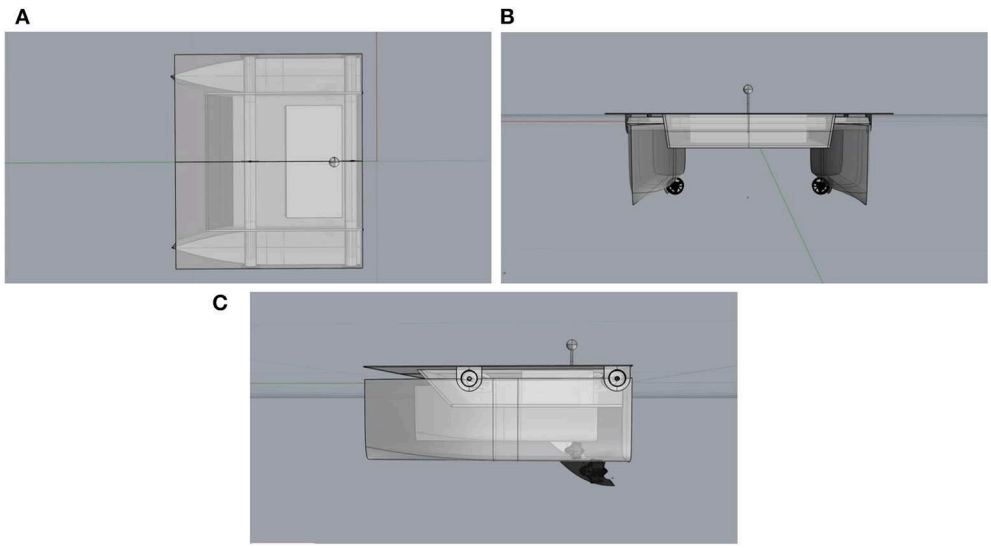


FIGURE 2 | Top **(A)**, front **(B)**, and side **(C)** views of the main vessel.

is an Intel NUC; the water surface vessel has 2 slots for installing Libelium Waspmote Smart Water and Smart Water Ions platforms[16].

Sensors are immersed with a winch. Such system is able to measure:

- pH
- Dissolved oxygen (DO), %
- Oxidation-reduction potential (ORP), mV
- Conductivity, μS/cm
- Temperature, °C
- Turbidity

Moreover, it is possible to detect ions presence, in particular: ammonium (NH4+), bromide (Br−), calcium (Ca2+), chloride (Cl−), cupric (Cu2+), fluoride (F−), iodide (I−), lithium (Li+), magnesium (Mg2+), nitrate (NO3−), nitrite (NO2−), perchlorate (CIO4−), potassium (K+), silver (Ag+), sodium (Na+).

In actual swarm configurations it is possible to integrate smaller and cheaper (sometimes expendable vessels) equipped with different and cheaper sensors.

4.1. Leader Vessel Hardware/Software Architecture

As told above, we plan to sample the environmental data by means of a small fleet of autonomous marine vessels. The "leader" vessel is in charge of the secure logging of the samples and leads the small swarm of vehicles performing the samples. In the

[16]Libelium (2018). *Drones, Sensors and Blockchain for Water Quality Control in the Volga River to Promote Trustworthy Data and Transparency*. Available online at: http://www.libelium.com/drones-sensors-and-blockchain-for-water-quality-control-in-the-volga-river-to-promote-trustworthy-data-and-transparency/ (accessed October 15, 2019).

following, we describe the "leader" vehicle on-board software. That software is based on the AIRA cyber-physical distribution kit running on Intel NUC. AIRA allows developers to implement "robotic tasks as a service" and allows the specification of a number of user parameters to customize the service itself. In the case of the vessel, the required "services" are the measuring and sampling missions. The user parameters of the mission are, in our case, the waypoints defining the vessel path and a list of sensors that should perform measurements during the robot motion along the requested path. The sensor samples are published via the IPFS network and are accessible by hash. This guarantees that the sampled data cannot be tampered and that they can be accessed by authorized persons (in our views the citizens, but in general different data accessibility schemes are possible).

The system software includes:

- General purpose Robonomics communication stack—standard set of components needed for connecting a CPS to the Robonomics platform:
  - Ethereum ROS API—vessel connection with the blockchain via ROS.
  - IPFS ROS API—vessel connection with IPFS via ROS.
  - Liability listener/Message signer—auxiliary services for liability: subscription to the Liability Markets, confirmation of the finalization of the liability.
- Application specific components:
  - Sensor data reader—reading and sending data from Libelium sensors.

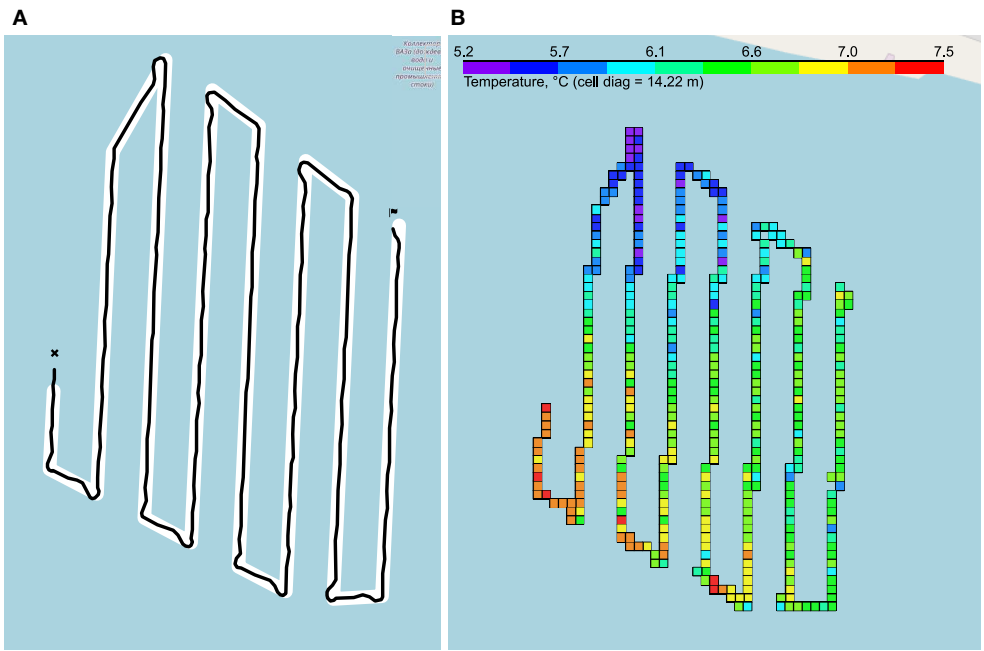


FIGURE 3 | Measurement results. **(A)** Vessel traveled route: planned path in white, real path in black. **(B)** Example of temperature measurements: the colored cells show the averaged measurements of the temperature.

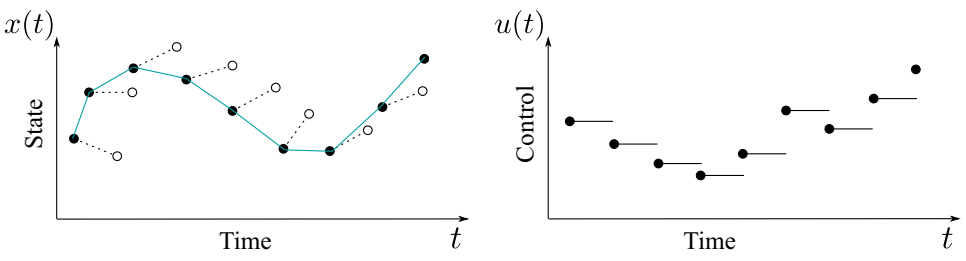


FIGURE 4 | Direct Transcription is a special kind of Non-linear Programming Optimization method. The control variables are discretized as piecewise-constant trajectories while the state variables are represented by linear segments. The optimization is performed by considering the discrete values of the variables representing the control and the state at each segment endpoint.

- Navigation package—motion control based on PX4 autopilot.
- Trader node—the node that is responsible for possible "economic" behaviors (accept the request for measurements or not on the basis of an agreed price of the service). In our tests, the vessel accepts any orders, as we are exploiting the Ethereum distributed ledger as a secure ledger. In general, it is possible to describe any economic behavior (for example, in real world settings, there might be more vessel fleets offering the same service; agents may have to choose which are the most advantageous offers, etc.)

The measurement algorithm works as follows:

1. The user sends the demand for the execution of the measurement mission. The "model" field determines the type of service requested, while the parameters for its execution are transferred in the "objective" field. These parameters are added to the "objective" rosbag-file directly or in a string message containing an IPFS hash link. In our case, the waypoint file with a description of the path and stopping time at each of the points is transmitted in the ROS-topic "`/waypoints`" by a hash reference. The "`/sensors`" topic contains a line with a list of sensors, whose readings should be transmitted according to the result of the mission. The boolean topic "`/virtual`" contains the permission to perform the measurement mission

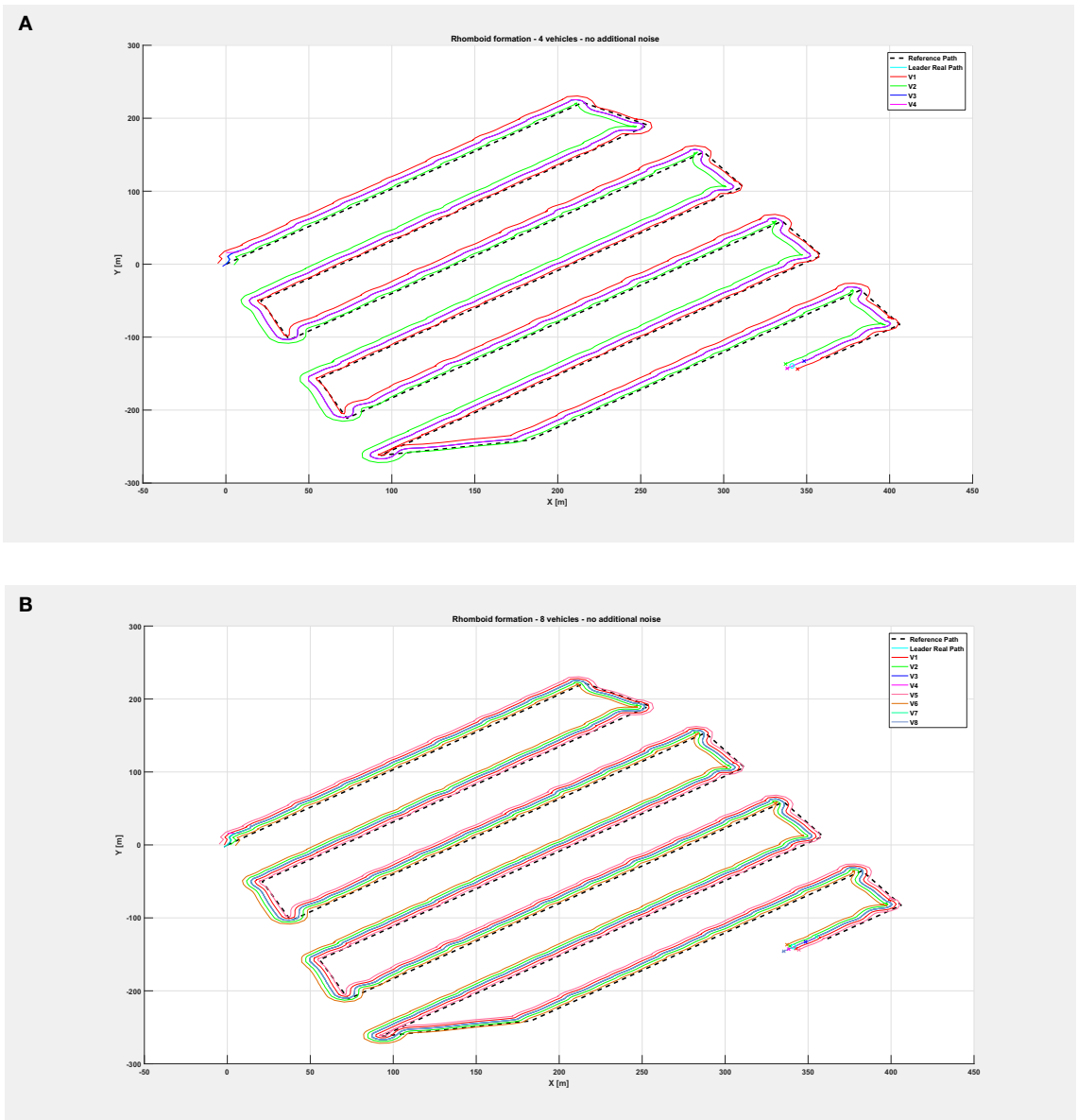


FIGURE 5 | Overall path followed by the vehicle swarm in nominal conditions with **(A)** $n = 4$ companion robots, **(B)** $n = 8$ companion robots.

virtually—to load the sensor readings as a result of the mission, measured previously at the points described in the "`/waypoints`" path. The "`expiration`" topic (Duration type) contains the expiration date of measurements for use in a virtual mission.

2. The demand is broadcast over the Robonomics network, AIRA software checks the parameters and sends an offer with the same parameters for order acceptance: model, objective, payment token, price and validator.
3. Both messages fall into the network and remain in the queues of providers of Robonomics. Matching offer and demand allows one of the providers to create a liability contract based on a delayed transaction mechanism[17]. The appearance of a contract in the blockchain, under which the vessel undertakes to execute a model with parameters from the "objective" field, confirms the appearance of an economically significant transaction (reservation of the customer's funds). This is a signal for the vessel to start working.
4. The vessel executes the contract: it loads the model and the objective from IPFS and starts the extraction of data from the "objective." The navigation and motion planning system

[17]Krupenkin, A. (2018). *Delayed Transactions for Solidity* [Russian]. Available online at: https://github.com/akru/blog/blob/master/posts/2018-02-09-delayed-transaction.markdown (accessed October 15, 2019).

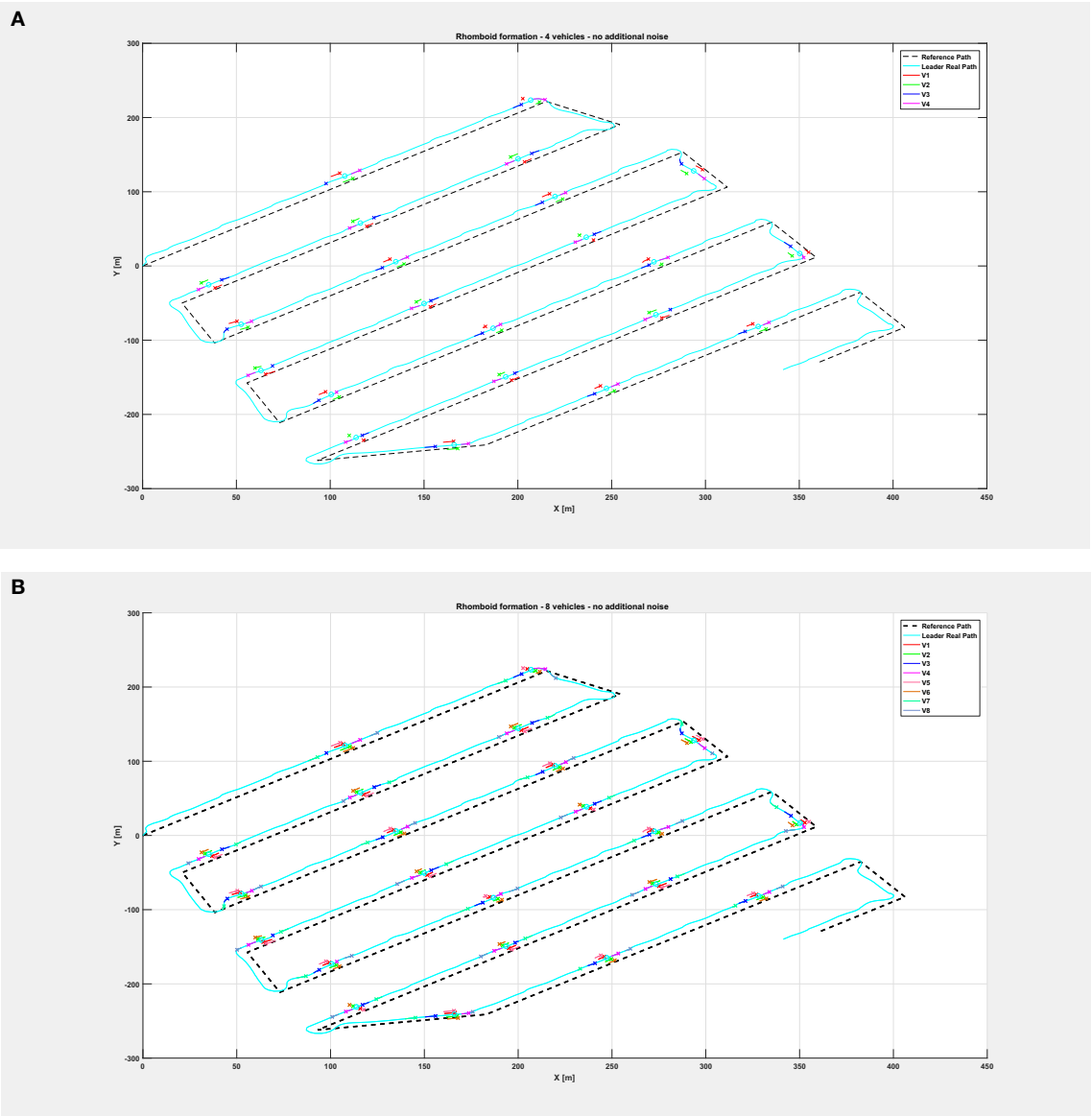


FIGURE 6 | Overall path followed by the vehicle swarm in nominal conditions with **(A)** $n = 4$ companion robots, **(B)** $n = 8$ companion robots. Only selected time instants are depicted.

receives waypoints and stop time intervals for measuring data on each points. The measuring system receives a signal about which sensors should be turned on. During the work, the data is written to a file on the vessel's on-board computer.

5. When the last waypoint is reached and the measurement is taken, the measurement mission is completed. The archive with readings of water quality sensors and geodata are added to IPFS. IPFS hashes are written to the result file in the rosbag format. Its IPFS hash is sent in a transaction to a liability contract with a digital signature.

The distributed ledger implementation protects the monitoring data from counterfeiting or from the hiding the fact itself of having performed the measurements. The location of the hash of the measurement file signed by the private key of the robot in the automatically guaranteed repository (the blockchain) makes the verification of data authenticity simple: we just need to check the IPFS hash of the file and verify it with that recorded in the "result" field of the liability contract. If these hashes do not coincide, it is obvious that the robot received another counterfeited file.

The code of the vessel with Robonomics part is available here, in the footnote.[18]

[18]Water Drone with Robonomics, GitHub repository, URL: https://github.com/Fingerling42/frontiers-vessel-code.

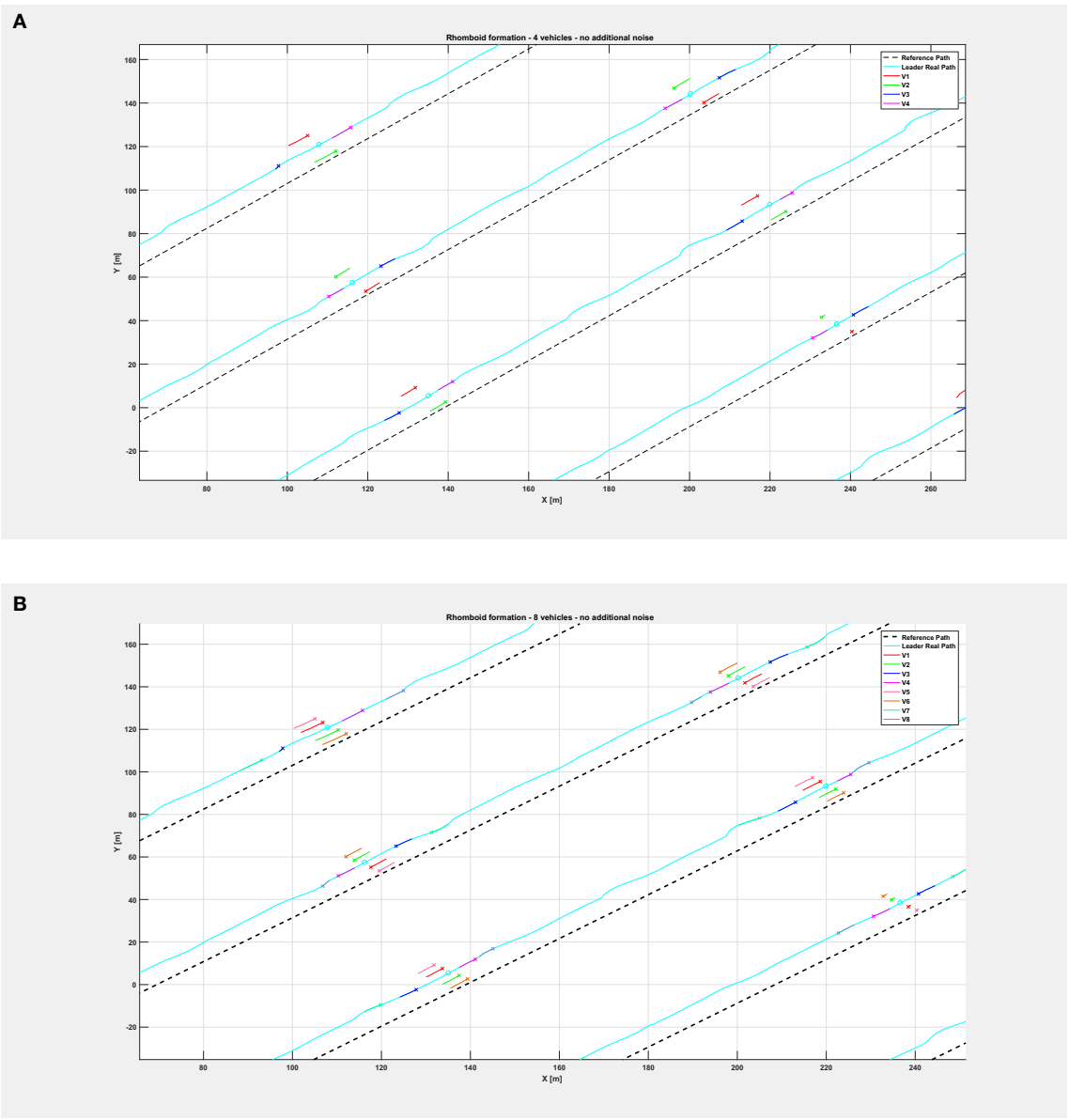


FIGURE 7 | Insight of the formation at selected instants of the path followed by the vehicle swarm in nominal conditions **(A)** with $n = 4$ companion robots, **(B)** with $n = 8$ companion robots. Note the rhomboid and double rhomboid formations kept by the swarm in the two different cases.

The "follower" robots will implement the BSP algorithms in order to be able to follow the leader and to keep the formation.

5. EXPERIMENTS AND ANALYSIS OF VESSEL SAMPLES

In the field experiments, the marine vessel measured dissolved oxygen (%), temperature (°C), pH level, and electrical conductivity (μS/cm) in the surface water layer of the coastal part of the Volga river in Kuibyshev reservoir near the storm drains of Avtozavodsky district, Togliatti, Samara region, Russia. Sensor immersion depth: 1.5–2 m.

Date and time (local time — GMT+4):

- Beginning — 4/25/2019, 7:12:52 PM.
- Ending — 4/25/2019, 9:46:30 PM.
- Total: 154 min.

The route was set in the Ardupilot GUI[19], which formed a file with waypoints. The planned and real routes are presented in **Figure 3A**.

As a result of measurements, the vessel sent data in the form of GPS and sensors data with Unix-timestamp. The Python programming language was chosen for working with data due

[19]http://ardupilot.org/

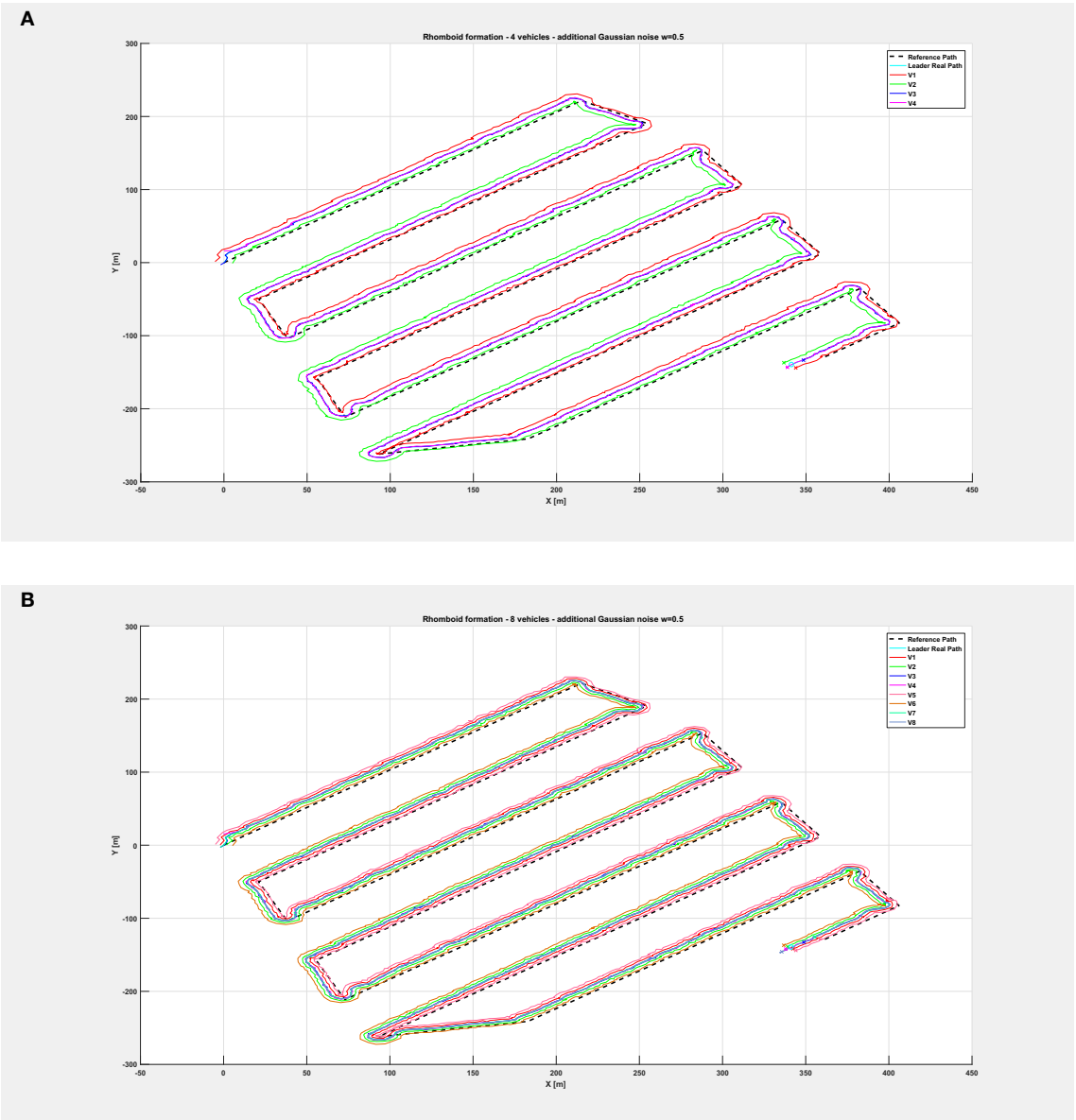


FIGURE 8 | Overall path followed by the vehicle swarm with injected additional noise with $w = 0.5$ and **(A)** $n = 4$ companion robots, **(B)** $n = 8$ companion robots.

to its ease of use, good performance and the presence of a wide range of libraries for file management, and data processing and visualization.

Measurements took place over long periods of time; they, together with GPS coordinates, were recorded discretely. Therefore, it was necessary to reduce such data into one structure, choosing from GPS only those coordinates that corresponded to the logs. The reconciliation was done using Unix timestamps. After extracting the data on the concentration of oxygen in water, it became clear that the representation of oxygen as a percentage is not enough for an adequate analysis of water quality, since in most cases it is necessary to translate the oxygen concentration in mg/l. Such a translation is non-trivial because it requires knowledge of the water temperature, normal oxygen concentration at normal atmospheric pressure at a given temperature, and atmospheric pressure in a given area. For that purpose, a dedicated software was developed. The data were visualized using the Folium Python library[20], as in **Figure 3B**.

All obtained data and an interactive map with the measurement results is available in the footnote link[21]. Also the raw data that was sent to IPFS is available at the following links:

[20]https://python-visualization.github.io/folium/

[21]Marine vessel data processing, GitHub repository, URL: https://github.com/Fingerling42/frontiers-vessel-data-processing

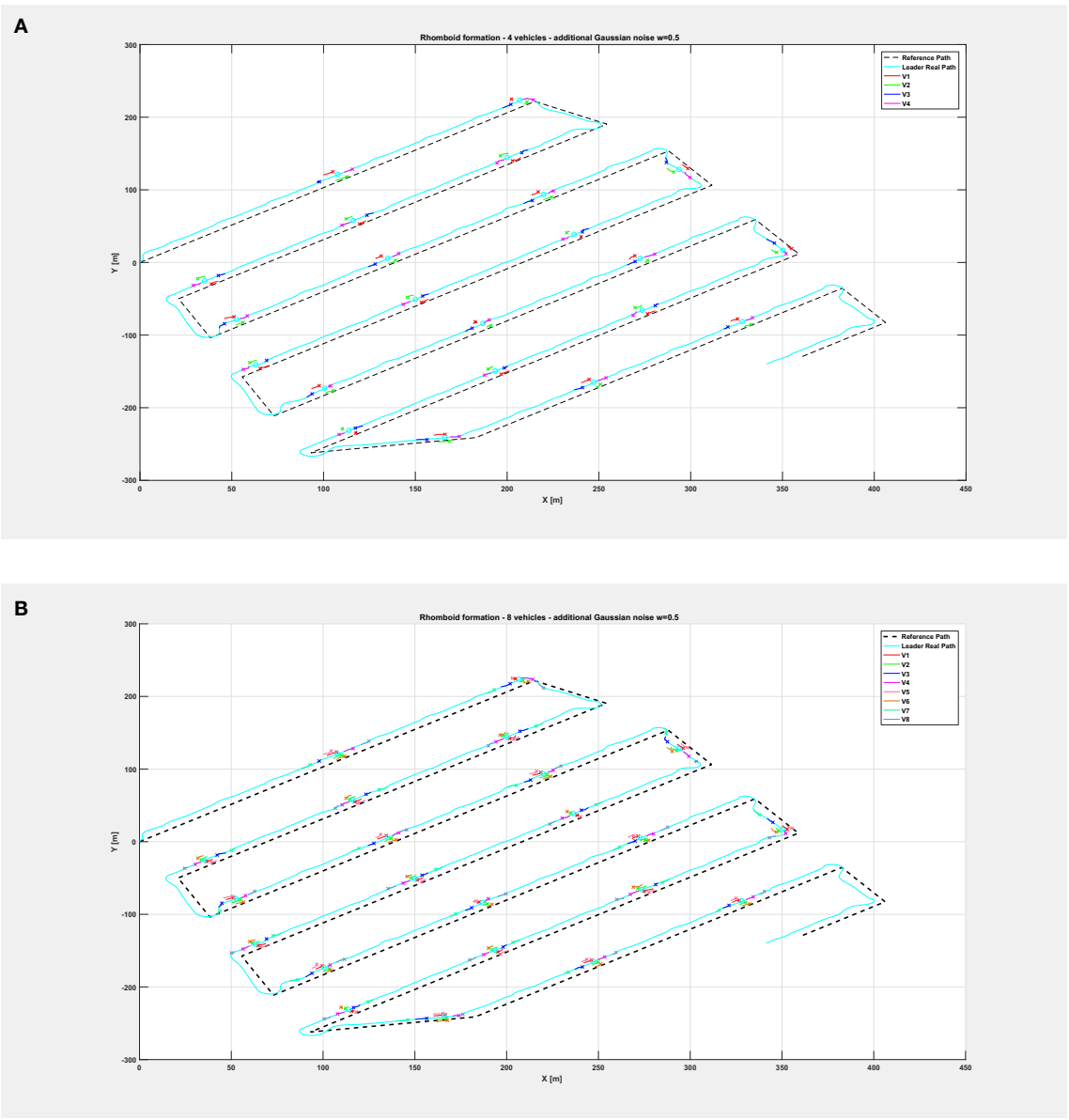


FIGURE 9 | Overall path followed by the vehicle swarm with injected additional noise with $w = 0.5$ and **(A)** $n = 4$ companion robots, **(B)** $n = 8$ companion robots. Only selected time instants are depicted.

- Geodata: https://gateway.ipfs.io/ipfs/QmPvULEGfDE2Roscy4zGpKpBE8s3sBwjiXJVQNS3sBxWDC
- Measurement data from sensors: https://gateway.ipfs.io/ipfs/QmWRjFcQi4Xcisqi8FP3AbGS3PB3gNHgtnfzbcpodKKCBP.

5.1. Analysis of Environmental Data

In this subsection, we provide a short summary of the investigated reservoir collected data.

5.1.1. pH Value

The concentration of hydrogen ions is of great importance for chemical and biological processes occurring in natural waters. In accordance with the requirements for the composition of water bodies in recreation areas and fishery reservoirs, the pH should not go beyond the range of 6.5–8.5. Based on the obtained data, the territory of the reservoir in terms of pH is more related to neutral and slightly alkaline waters and only a few segments are characterized by a high pH (alkaline waters with pH $=$ 8.5 . . . 9.5).

5.1.2. Electrical Conductivity

According to the electrical conductivity level of natural water, we can evaluate the mineralization of water. The conductivity in the studied area does not exceed the standards: the average value of the conductivity is 338.9 μS/cm, which corresponds to 169.45 mg/dm^3 mineralization level. The studied water area can be attributed to the ultra fresh water category. It should be noted that electrical conductivity increases with distance from the coast. The conductivity results correspond to the regular dependence of electrical conductivity and temperature, with a correlation coefficient $r = 0.77$ at $p = 0.05$, which characterizes a strong positive coherence.

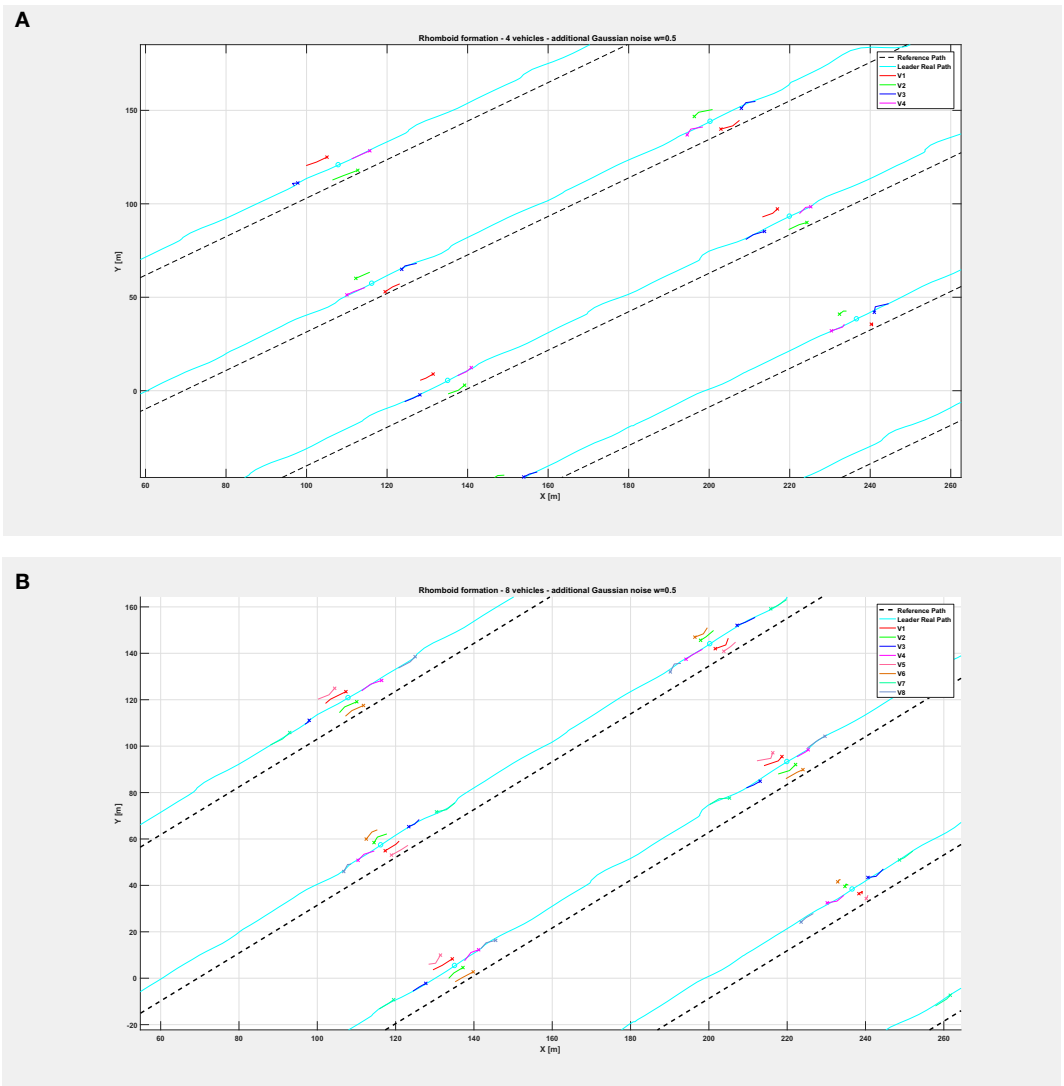


FIGURE 10 | Insight of the formation at selected instants of the path followed by the vehicle swarm with injected additional noise with $w = 0.5$ and **(A)** $n = 4$ companion robots, **(B)** with $n = 8$ companion robots. Note the rhomboid and double rhomboid formations kept by the swarm in the two different cases.

5.1.3. Oxygen

For dissolved oxygen, World Health Organization does not offer any value for indications of its effect on health. However, a sharp decrease in the oxygen content in water indicates its chemical and/or biological pollution. In the obtained data, the amount of dissolved oxygen varies from 0 to 12.9 mg/dm^3. During statistical processing, the data were divided into two groups: in the intervals [0; 1.7] and [8.5; 13.6] with an average value of 6.759 mg/dm^3. The obtained intervals characterize the level of water pollution in the studied reservoir as dirty waters (interval [0; 1.7]) and clean waters (interval [8.5; 13.6]).

Since the content of oxygen dissolved in water depends on the temperature of the water and its mineralization, a pair correlation analysis was performed to determine the relationships: with a sample size of $n = 1194$, the critical value of the Pearson correlation coefficient $r_{xy} = 0.06$ at $p = 0.05$.

Accordingly, the values of the concentrations of oxygen dissolved in water have very weak dependence on temperature (correlation coefficient −0.091), a weak positive dependence on pH (correlation coefficient 0.156), and no dependence on conductivity.

6. REYNOLDS' BOIDS SWARM IMPLEMENTATION BY MEANS OF A BELIEF SPACE PLANNING APPROACH

Swarm behaviors were developed by following the approach proposed by some of the authors in Bonsignorio et al. (2019). Such work extended and applied the approach proposed in Platt et al. (2010) to robotic swarm control. Older work on BSP (Belief Space Planning) by those authors dealt with the trajectory

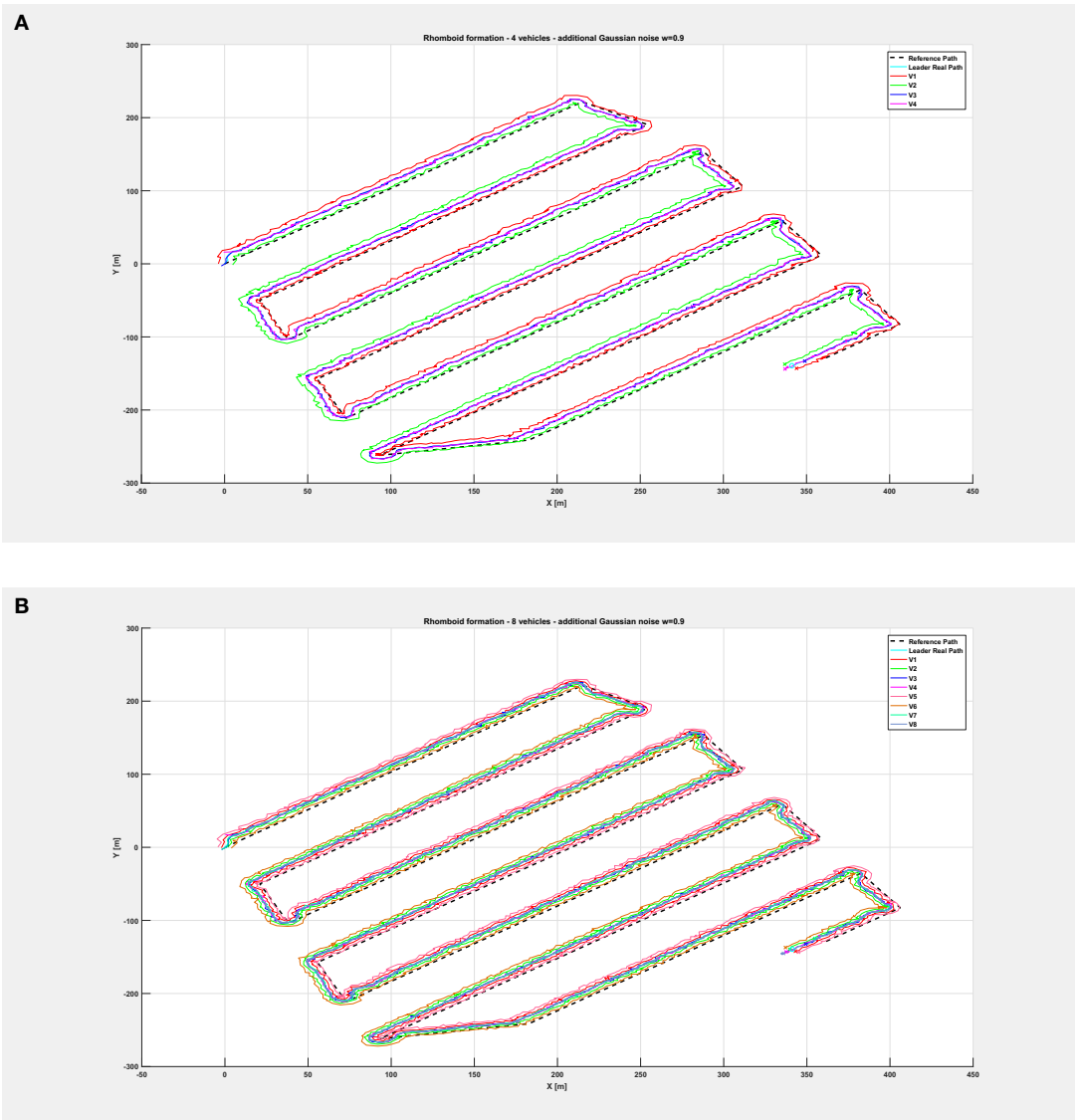


FIGURE 11 | Overall path followed by the vehicle swarm with injected additional noise with $w = 0.9$ and **(A)** $n = 4$ companion robots, **(B)** $n = 8$ companion robots.

planning and control of a robotic 3D-printed manipulator with deliberately poor joint accuracy and actuation with no joint feedback (Zereik et al., 2015), as well as in the motion planning of a marine companion robot for diver assistance and support (Zereik et al., 2014). Belief Space Planning methods allow to concurrently reduce the uncertainty (expressed by the state estimate measure covariance) and reach the desired state. Those features make them very suitable to perform tasks in unstructured environments characterized by significant measure noise. A trajectory in the "Belief Space" for a vehicle moves it from its current state (for example a given position/orientation), represented as a Gaussian PDF (Probability Density Function), to a goal state PDF with the desired mean value and lower covariance. The system state is modeled as the sum of a signal component and a Gaussian noise part.

The trajectory planned in the belief space for the vehicle is linearly approximated by a series of segments in the belief space. The initial and end points of each segment are determined by Direct Transcription. Such discretization method is depicted in **Figure 4**; for further details refer to (Platt et al., 2010) and (Betts, 2010).

The algorithm moves on the Belief Space trajectory piece-wise, segment by segment. The needed control actions to move from one segment extreme to the beginning of the following one are computed through a Linear Quadratic Regulator (LQR). The procedure is iterated on the segments of the linearized

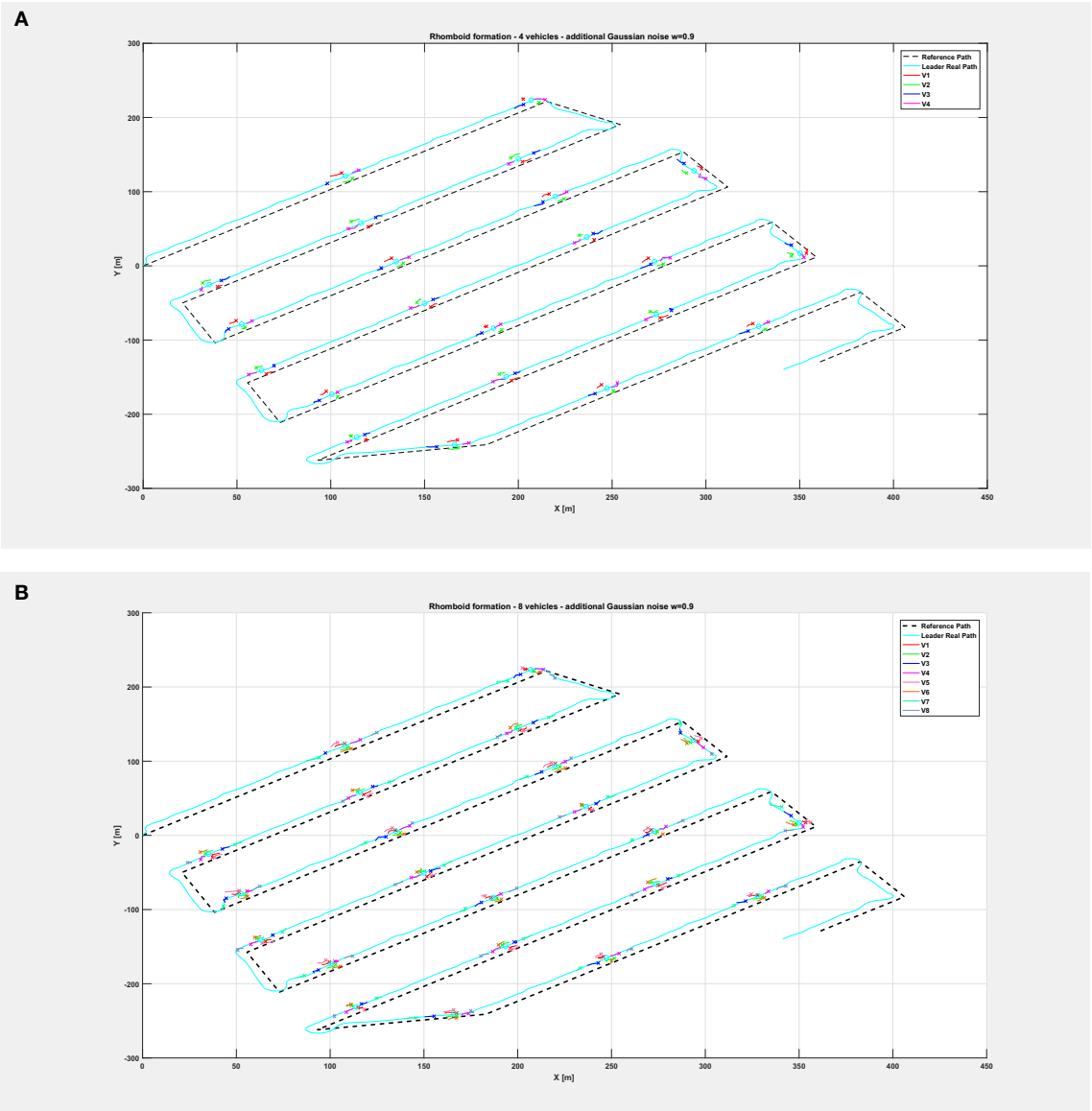


FIGURE 12 | Overall path followed by the vehicle swarm with injected additional noise with $w = 0.9$ and **(A)** $n = 4$ companion robots, **(B)** $n = 8$ companion robots. Only selected time instants are depicted.

trajectory until the vehicle reaches its final goal in the belief space, represented by a vector of Gaussian PDFs with desired mean values for the end point and orientation, as well as a reduced covariance of the point and orientation expected measures. When a single vehicle follows a requested path, at each step, the current reference trajectory point is sent to

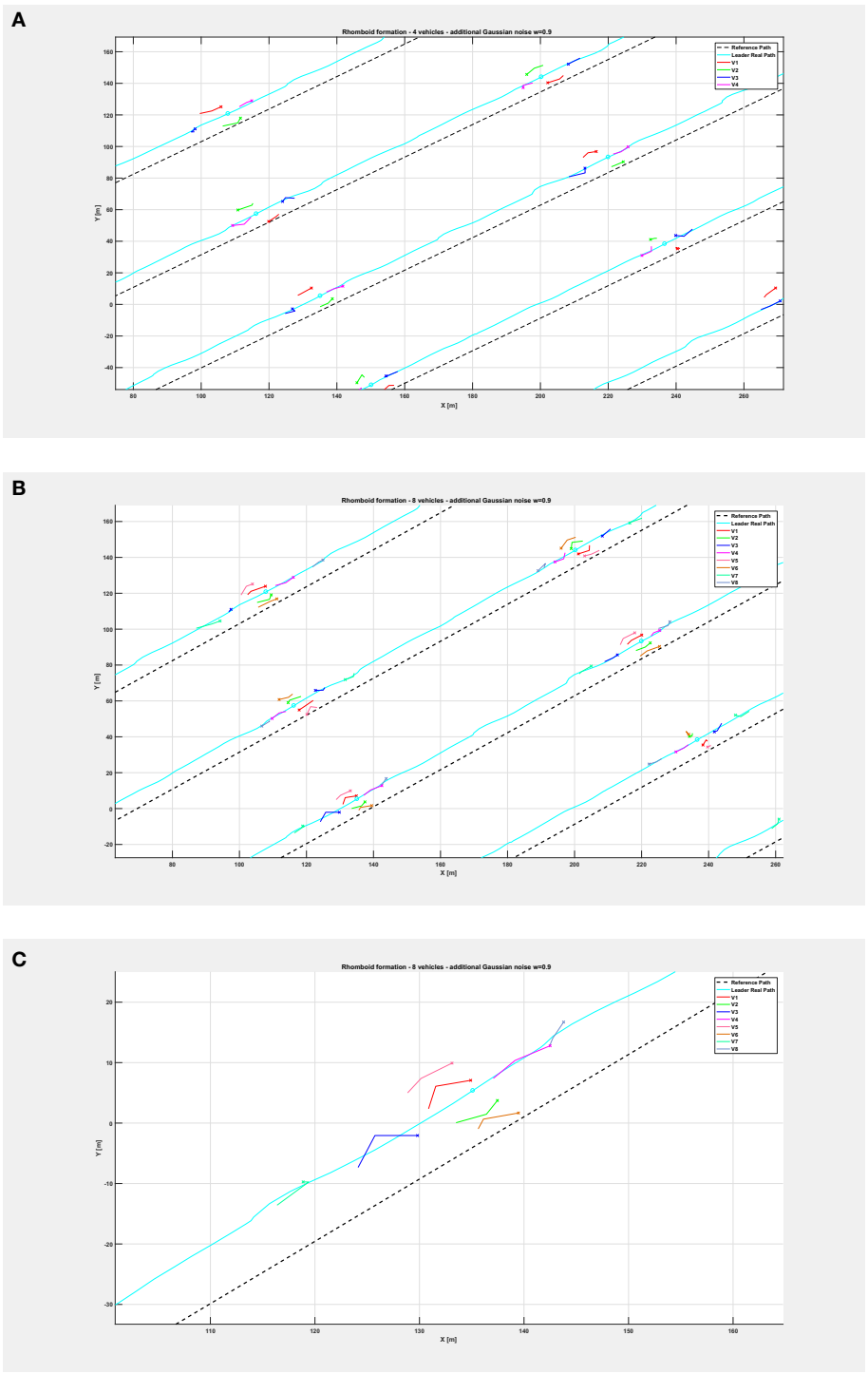


FIGURE 13 | Insight of the formation at selected instants of the path followed by the vehicle swarm with injected additional noise with $w = 0.9$ and **(A)** $n = 4$ companion robots, **(B)** $n = 8$ companion robots (note the rhomboid and double rhomboid formations kept by the swarm in the two different cases). **(C)** Zoom of the fourth swarm formation (in the path order) of the previous **(B)**, highlighting that vehicles in the double rhomboid formation do not collide.

the vehicle controller that drives the robot in the requested intermediate position. The final goal in the belief space is reached by iterating this process. At each step, the algorithm verifies if the current calculated intermediate end point of the computed segment of the plan is good enough to approach the final position and, if not, re–plans intermediate point. In our case, all companion autonomous vessels follow the trajectory of the leader vessel—the one managing the measure certification functions—by maintaining a predefined distance (some of them in terms of the cross-track error, some in terms of distance along the path). In this way, the whole robotics swarm system maintains a rhomboid formation while following the required trajectory. All the swarm vessels follow the leader, which is in charge of managing the certification of the samples by means of the distributed ledger processes provided by the blockchain platform.

Since all actions in the belief space actually weigh the objective to move the robot in the desired position with the objective of reducing the observation covariance in the mid waypoints and at the end point, they can be seen as information gathering actions (as they reduce the uncertainties on the vehicle position). The planning occurs in a state space which is inherently non-linear and has a higher number of dimensions than the physical state space; as a consequence the resulting dynamic is significantly underactuated (as the number of control input affecting the physical system is lower than the dimensions of the belief space). The problem can be simplified by the assumption of maximum likelihood of observations, as in the present paper,

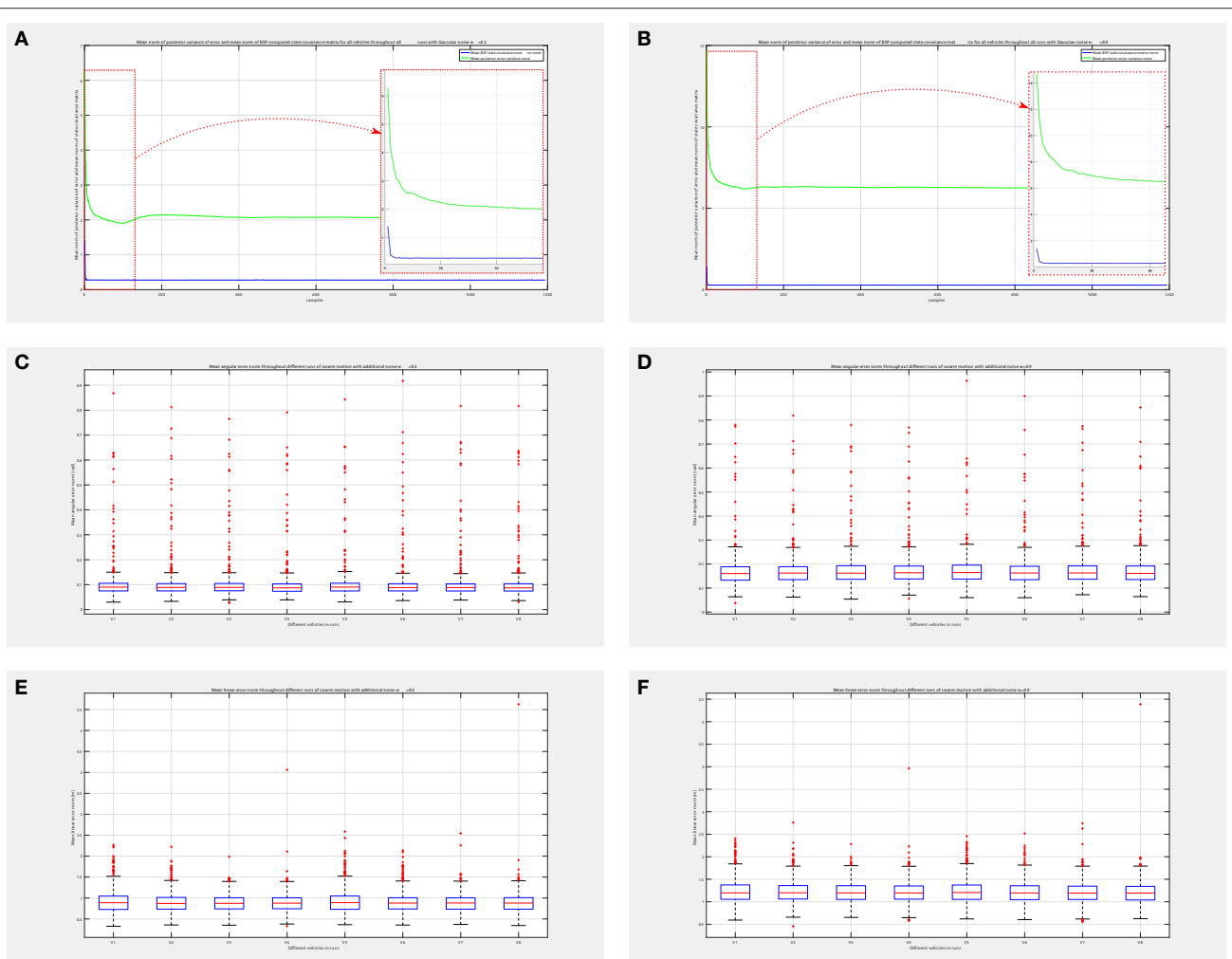


FIGURE 14 | Simulation performance assessment for swarm with $n = 8$ companion vehicles. Mean norm of posterior variance of error and mean norm of BSP-computed state covariance matrix of all vehicles throughout all runs with noise **(A)** $w = 0.5$, **(B)** $w = 0.9$, with related zoom on the initial part of the graphs. Comparing norm of covariance matrix with that of posterior variance it is clear that the BSP strategy is able to quickly reduce the covariance matrix of system state, thus demonstrating its good performance. Mean angular error norm of all vehicles throughout all runs with noise **(C)** $w = 0.5$, and **(D)** $w = 0.9$. Mean linear error norm of all vehicles throughout all runs with noise **(E)** $w = 0.5$, and **(F)** $w = 0.9$. Please note that there are some outliers in the boxplots due to the fact that our approach provides a way-point control; the greater number of outliers in the angular component is due to the very steep angular changes in reference when the vehicles are required to invert their motion direction.

in our previous work and in Platt et al. (2010). This maximum likelihood assumption asserts that the current system state is the most likely state according to the past observations and the performed actions; this means that the performed actions achieve their intended purpose, leading the system to the desired state. Our simulations (in various application contexts) confirm that this assumption is usually correct. In Platt et al. (2010), a formal proof is provided about its optimality under linear Gaussian process assumptions. Here, the observations $z_t \in \mathbb{Z}$ of the distance between the goal and the vehicle position are modeled as a non-linear stochastic function of the vector $x_t \in \mathbb{X}$, representing the (not directly observed) state of the system and of the environment

$$z_t = g(x_t) + \xi \tag{1}$$

where g is a deterministic function of the measurement and ξ is a zero mean Gaussian noise with covariance W_t, dependent on the state. The deterministic function f links the new state to the older one under the control action u_t

$$x_{t+1} = f(x_t, u_t) \tag{2}$$

where f and g are assumed to be differentiable functions of x_t and u_t. The controller is assumed to know the state through a probabilistic density function $P(x)$. The parameters of such a distribution are the "belief state" $b_t = \left[m_t^T \; s_t^T\right]^T$, where m_t is the mean of the belief state and $s = \left[s_1^T, \cdots, s_d^T\right]^T$ is a vector composed by the d columns of the covariance matrix Σ. If a linear Gaussian dynamics is assumed, the belief state can be updated by rules of the form

$$x_{t+1} = A_t(x_t - m_t) + f(m_t, u_t) \tag{3a}$$

$$z_t = C_t(x_t - m_t) + g\left(f(m_t, u_t)\right) + \xi \tag{3b}$$

where A_t and C_t are the Jacobian matrices $A_t = \frac{\delta f}{\delta x}(m_t, u_t)$, $C_t = \frac{\delta g}{\delta x}(m_t)$. The Gaussian distribution is given by:

$$\Sigma_t : P(x) = \mathcal{N}\left(x/m_t, \Sigma_t\right) \tag{4}$$

In these hypotheses, and assuming maximum likelihood of the observations, it can be proved that it is possible to derive by iteration a series of segments with an associated set of control actions by minimizing the cost function J

$$J\left(b_{\tau:T}, u_{\tau:T}\right) = \sum_{i=1}^{k} w_i \left(\hat{n}_i^T \Sigma_t \hat{n}_i\right)^2 + \sum_{t=\tau}^{T-1} \tilde{m}_t^T Q \hat{m}_t + \tilde{u}_t^T R \tilde{u}_t \tag{5}$$

where $b_{\tau:T}$ is the subset of the state space, $u_{\tau:T}$ are the corresponding actions for a given state space trajectory, Q and R are weight matrices, and the n_i are the versors of belief space along which the optimization is performed. Finally, Σ_T is the covariance matrix at the end of the segment, and m_t^T the value of the mean of the Gaussian of the measures. The function J is minimized by a standard SQP (Sequential Quadratic Programming) algorithm; after this, a linear quadratic regulator is applied to move along the segments.

Algorithm 1 : BSP Algorithm

Input : b_0, b_{goal}
Output: $u_{1:s}$

```
1  initBSP ();
2  for i = 1 to N do
3      (m̄_{1:s}, ū_{1:s}) = CreatePlan(m_0, m_goal);
4      for j = 1 to s − k do
5          k = r + j;
6          u_t = LQR(ū_t, m̄_t, m_t);
7          z_t = g(x_t) + ξ;
8          m_{t+1} = EKF(m_t, u_t, z_t);
9          if ‖m̄_t − m_t‖ < thr_1 then
10             while e_t > thr_2 do
11                 η_t = DriveVehicle (m_t);
12         else
13             r = r + j − 1;
```

The procedure is summarized in Algorithm 1. The BSP strategy needs to know the initial belief state and final goal b_0 and b_{goal}, and returns the sequence $u_{1:s}$ of the control actions. As a preliminary step all the variables of the algorithm are initialized to proper values via the function `InitBSP` (line 1 of Algorithm 1). The procedure is then executed for a predefined number of steps N. At each step, a plan is calculated via the `CreatePlan` function (line 3), obtaining the two sequences $(\bar{m}_{1:s}, \bar{u}_{1:s})$. This plan is executed for s steps; in case the planned steps do not converge to the final goal, variable k and counter r (line 13) are in charge of managing the eventual re–planning. In this phase, three values are calculated (lines 6–8): u_t is returned by the LQR control, while z_t is the noisy perceived position measurement (see Equation 1); finally the value m_{t+1} is propagated through an Extended Kalman Filter (EKF). If the resulting error between the mean of the current reference belief state $\bar{m}_t$ and the mean of the current belief state m_t is under a given threshold thr_1 (line 9), the algorithm sends the commands to the underlying vehicle low-level control system. The algorithm drives the robot, via the function `DriveVehicle`, toward the desired intermediate point in the trajectory (line 11), allowing the Cartesian error e_t to converge under a given threshold thr_2 (line 10). The function `DriveVehicle` returns as output the necessary vehicle trajectories η_t. Finally, if the error $\|\bar{m}_t - m_t\|$ is greater than thr_1, the counter r is updated in order to proceed with a necessary re–planning step (line 13).

The adoption of LQR standard control improves the efficiency of BSP planning: the evaluation of the optimal control action for the system leads to the stabilization of the trajectory in spite of the non-linear dynamics of the system. Indeed, LQR control is able to handle small divergences from the planned motion, and thus minimizes the number of re–planning steps the system must calculate, improving in this way the

computational efficiency. Previous experimental tests within a different application described in Zereik et al. (2015) showed that the statistical LQR calculation produces a decrease in computing time of about 70%. Hence, it is clear that the use of LQR in the algorithm leads to a smoother, and hence more efficient, motion of the robots.

The BSP approach is here integrated with the classical Reynolds flocking model. Reynolds has shown in Reynolds (1987) that flocking behaviors can be implemented by imposing to the individual agents a surprisingly simple set of rules:

1. *Separation*—Avoid crowding neighbors (short range repulsion).
2. *Alignment*—Steer toward average heading of neighbors.
3. *Cohesion*—Steer toward average position of neighbors (long range attraction).

In our system, agents target the school of (robot) fishes center (rules 2, 3) and keep distance among themselves (rule 1).

7. BSP-BASED SWARM MIXED REALITY SIMULATION

Using the BSP-based swarm approach described above, a swarm of vehicles has been simulated, with a different number n of companions, namely $n = 4$ and $n = 8$. The boat in charge of the sample collection and certification has been considered as the "leader" and the other vehicles have been requested to keep a rhomboid formation with the boat at the center, while following the master path. Note that the boat position at each instant is assumed to be known by the vehicle swarm. This is reasonable in a real application: indeed, the swarm and the boat move together along the reference path, so that the boat can easily communicate its current position to the vessel swarm, e.g., through a long-range WiFi or radio connection. Furthermore, each vehicle has its own GPS+IMU localization system on-board, in such a way to be able to determine its own relative position with respect to the main boat.

The tests of the swarm have been performed in a mixed reality simulation setting as the trajectory of the leader has been obtained by field tests of the vessel described above. We assume zero latency in the sample transmission from the follower vessels to the leader one. This assumption is realistic in comparison to the sampling rate and Ethereum typical transaction rate. The BSP swarm strategy is simple but very effective, since (as already stated) it allows each vehicle to follow the requested path while keeping the desired formation and, in the meantime, reducing uncertainty due to both inaccurate measurement and environmental disturbance. To this aim, in order to test robustness of the approach, beside the first simulation in nominal conditions, an additional Gaussian-distributed noise has been injected in the system, to stress the algorithm. In particular, the noise has been generated as a normal distribution with zero mean and covariance equal to wW_i, where w is a tunable scalar coefficient and W_i is the corresponding diagonal component (x, y or ψ) of the covariance matrix $W \in \mathbb{R}^{3\times3}$ (each noise component is assumed to be independent and uncoupled from the other). For the angular component ψ, there is a further scale factor (equal to 0.1) to adjust noise values with respect to radians. Each diagonal element of matrix W is equal to $\sqrt{\sum_i \left(\xi_i - \xi_i^*\right)^2}$ (where ξ_i is the single component of the Cartesian error and ξ_i^* is the related goal value for that component) if the Cartesian error is below a given threshold, or equal to a larger constant value rg otherwise.

For each formation type ($n = 4$ or $n = 8$ companion vehicles), two different values of w have been tested, namely $w = 0.5$ and $w = 0.9$; 10 simulation runs for each category were successfully executed, thus resulting in a total of 40 successful experiments.

Figure 5 shows the path followed by the vehicle swarm in nominal conditions (no additional noise injected in the system), while **Figure 6** depicts only selected time instants of the path following execution, highlighting the rhomboid formation maintained by the swarm. **Figure 7** provides a zoomed insight of a part of the simulation, showing the formation at some of the previous selected time instants.

The same organization is kept for the next figures: again, **Figure 8** depicts the overall path followed by each vehicle when an additional noise with $w = 0.5$ is injected in the system, while **Figure 9** shows only selected time instants of the path following execution, highlighting the rhomboid formation maintained by the swarm. **Figure 10** provides a zoomed insight of a part of the simulation, showing the formation at some of the previous selected time instants.

Finally, relatively to the case with additional noise with $w = 0.9$, **Figure 11** depicts the overall paths followed by the vehicle swarm. **Figure 12** shows only selected time instants of the path following execution, highlighting the rhomboid formation maintained by the swarm. **Figures 13A,B** provide a zoomed insight of a part of the simulation, showing the formation at some of the previous selected time instants. **Figure 13C** highlights a particular time instant of the simulation, in order to show that vehicles are not colliding.

An assessment of good performance can be obtained by analyzing **Figure 14**. The evolution of mean norm of error posterior variance and of mean norm of the state covariance matrix Σ_t, computed on-line by the BSP algorithm, can be compared from **Figures 14A,B** in both noisy cases. From this analysis, it is clear that the BSP strategy strongly reduces the covariance matrix on the system state and keeps it low during the whole simulation run, even if the Cartesian error is large. A zoom of the initial part of the graph is provided for both cases, for which the very fast reduction of the covariance matrix mean norm is clear. Boxplots of **Figures 14C–F** show the mean angular and linear error norm of all vehicles throughout all experiments. The related dataset can be found here: https://github.com/cyber-chicca/Swarm-BSP.

8. CONCLUSIONS AND FUTURE WORK

We have developed a system that allows certified and trustable environmental sampling and logging by joining a sample certification scheme based on blockchain technologies and

swarm behaviors based on a BSP implementation of Reynolds' Boids. Our experiments at field, integrated by extended mixed reality simulation, show the viability of the approach.

This article describes a new platform for trustable environmental monitoring based on citizen-led certification of the samples by means of distributed ledger technologies. This constitutes a further step in "Citizen Science." The idea of citizen science, see Hippel (1991); Gura (2013); Hand (2010) is usually implemented by sharing with the public experimental data collected by one or more public or private organizations. In our case the citizens, thanks to intelligent robotics technologies integrated within a distributed ledger framework, have under their control the collection of the data themselves. This is particularly important for sensitive issues related to environmental quality, but can be relevant in many other societal relevant issues, for example the geographical distribution of infected people during a pandemic. We have shown that this conceptual approach can be implemented on top of the Ethereum blockchain network in a robust way and scalable way. Our platform allows to merge the benefits of distributed certification of the samples, made possible by the blockchain technology, with the adaptivity and scalability of swarm architecture. It separates the processes related to the sample certification, managed by the leader vessel, from the processes related to the optimization of sample collection performed by means of a potentially heterogeneous swarm of smaller vessels dedicated to the physical execution of the sampling activities. We have shown how mixed reality simulation can be a valuable tool for the design of specific system architectures for specific applications. Simulation technologies cannot substitute field experiments. However, they allow a greater and more systematic set of test runs than usually possible in the field. The experiments that we have performed in the field, where we have equipped the leader vessel with the set of sensors that in future implementations will be spread among the smaller vehicles, have shown the trustability of the certification of the samples. The purpose of the swarm simulation was to show that it is possible to implement the swarming behaviors that we have devised and provide guidance for future developments of the platform and its deployment in the field.

In the future, we will develop and perform two-ways mixed reality simulation in order to refine the system design and we will then proceed to the implementation of real world swarms at field.

Our approach is scalable since we can manage more swarms with different leader vessels. We will also consider the implementation of the BSP swarming approach to a fleet of leader vessels and other approaches based on Gaussian Processes and information gain. We will also consider the possible benefits of Deep Reinforcement Learning methodologies for the platform described in this paper.

DATA AVAILABILITY STATEMENT

All datasets generated for this study are included in the article/supplementary material.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial and intellectual contribution to the work, and approved it for publication.

FUNDING

This work was supported by the Russian Science Foundation grant (project 19-19-00403).

ACKNOWLEDGMENTS

The authors want to thank the Airalab[22] team for the development and support of the unmanned surface vessel, as well as for providing the Robonomics platform support and related software.

[22]https://aira.life/en/

REFERENCES

Akpor, O. B., Ohiobor, G. O., and Olaolu, D. (2014). Heavy metal pollutants in wastewater effluents: sources, effects and remediation. *Adv. Biosci. Bioeng.* 2, 37–43. doi: 10.11648/j.abb.20140204.11

Alharsha, I., Memon, F., and Farmani, R. (2018). "An assessment of per capita water consumption in Sirte, Libya," in *Int. Conf. on Urban Drainage Modelling* (Palermo: Springer), 969–975. doi: 10.1007/978-3-319-99867-1_167

Alkhelaiwi, A., and Grigoras, D. (2015). "The origin and trustworthiness of data in smart city applications," in *2015 IEEE/ACM 8th Int. Conf. on Utility and Cloud Comp*. (*UCC)* (Limassol: IEEE), 376–382.

Arias, P. A., Villegas, J. C., Machado, J., Serna, A. M., Vidal, L. M., Vieira, C., et al. (2016). Reducing social vulnerability to environmental change: building trust through social collaboration on environmental monitoring. *Weather Clim. Soc.* 8, 57–66. doi: 10.1175/WCAS-D-15-0049.1

Arnold, R., Carey, K., Abruzzo, B., and Korpela, C. (2019). "What is a robot swarm: a definition for swarming robotics," in *2019 IEEE 10th Ann. Ubiquitous Comp., Electronics & Mobile Comm. Conf.* (*UEMCON)* (New York, NY: IEEE), 74–81. doi: 10.1109/UEMCON47517.2019.8993024

Assemany, P. P., Calijuri, M. L., do Couto, E. de A., da Silva, F. P., and de Souza, M. H. B. (2019). "Energy recovery in high rate algal pond used for domestic wastewater treatment," in *Algal Technologies for Wastewater Treatment and Resource Recovery*, eds R. Muñoz, H. Temmink, A. M. Verschoor, and P. van der Steen (London: IWA Publishing), 12–20.

Betts, J. T. (2010). "Practical methods for optimal control and estimation using nonlinear programming," in *Advances in Design and Control, 2nd Edn*, ed R. C. Smith, A. C. Antoulas, S. Banda, B. A. Batten, J. Betts, S. L. Campbell, E. M. Cliff, M. C. Delfour, M. D. Gunzburger, J. W. Helton, A. J. Krener, K. Morris, R. Murray, and E. Sachs (SIAM), 132–133. doi: 10.1137/1.9780898718577

Bijani, S., and Robertson, D. (2014). A review of attacks and security approaches in open multi-agent systems. *Artif. Int. Rev.* 42, 607–636. doi: 10.1007/s10462-012-9343-1

Bonsignorio, F., Zereik, E., and Caccia, M. (2019). "BSP-based swarm behaviour implementation for underwater multiagent environmental low-frequency sensing," in *12th IFAC Conf. on Control Appl. in Marine Systems, Robotics, and Vehicles (CAMS 2019)* (Daejeon: IFAC). doi: 10.1016/j.ifacol.2019.12.309

Bouraoui, F., and Grizzetti, B. (2014). Modelling mitigation options to reduce diffuse nitrogen water pollution from agriculture. *Sci. Tot. Env.* 468, 1267–1277. doi: 10.1016/j.scitotenv.2013.07.066

Burek, P., Satoh, Y., Fischer, G., Kahil, M. T., Scherzer, A., Tramberend, S., et al. (2016). *Water Futures and Solution - Fast Track Initiative (Final Report)*. IIASA Working Paper [Monograph]. Available online at: http://pure.iiasa.ac.at/13008 (Accessed October 15, 2019).

Cai, C., Liu, C., and Men, B. (2019). "Research progress in water quality improvement," in *The 5th Annual International Conference on Material Engineering and Application (ICMEA 2018), Vol. 484* (Wuhan: IOP Publishing). doi: 10.1088/1757-899X/484/1/012050

Castelló Ferrer, E. (2018). "The blockchain: a new framework for robotic swarm systems," in *Proceedings of the Future Technologies Conference (FTC) 2018. FTC 2018* (Cham: Springer), 1037–1058. doi: 10.1007/978-3-030-02683-7_77

Christidis, K., and Devetsikiotis, M. (2016). Blockchains and smart contracts for the internet of things. *IEEE Access* 4, 2292–2303. doi: 10.1109/ACCESS.2016.2566339

Dannen, C. (2017). *Introducing Ethereum and Solidity*. New York, NY: Springer. doi: 10.1007/978-1-4842-2535-6

Duarte, M., Gomes, J., Costa, V., Rodrigues, T., Silva, F., Lobo, V., et al. (2016). "Application of swarm robotics systems to marine environmental monitoring," in *OCEANS 2016-Shanghai* (Shanghai: IEEE), 1–8. doi: 10.1109/OCEANSAP.2016.7485429

Food and Agriculture Organization of the United Nations (2011). *The State of the World's Land and Water Resources for Food and Agriculture: Managing Systems at Risk*. Earthscan, New York, NY.

Gura, T. (2013). Citizen science: Amateur experts. *Nature* 496, 259–261. doi: 10.1038/nj7444-259a

Hajdari, R. (2015). *Establishing an effective national environmental information system* (Master's thesis). Rochester Institute of Technology, Kosovo.

Hand, E. (2010). Citizen science: people power. *Nature* 466, 685–687. doi: 10.1038/466685a

Hippel, F. V. (1991). *Citizen Scientist*. New York, NY: Springer.

Jacques, P. (2016). *Environmental Skepticism: Ecology, Power and Public Life. Global Environmental Governance*. New York, NY: Taylor & Francis. doi: 10.4324/9781315580050

Kapitonov, A., Lonshakov, S., Berman, I., Ferrer, E. C., Bonsignorio, F. P., Bulatov, V., et al. (2019). Robotic services for new paradigm smart cities based on decentralized technologies. *Ledger* 4, 56–66. doi: 10.5195/ledger.2019.177

Koubaa, A. (2018). *Robot Operating System (ROS): The Complete Reference*, Vol. 3. New York, NY: Springer. doi: 10.1007/978-3-319-54927-9

Leaf, S. (2018). Taking the p out of pollution: an English perspective on phosphorus stewardship and the water framework directive. *Water Environ. J.* 32, 4–8. doi: 10.1111/wej.12268

Lončar, I., Babić, A., Arbanas, B., Vasiljević, G., Petrović, T., Bogdan, S., et al. (2019). A heterogeneous robotic swarm for long-term monitoring of marine environments. *Appl. Sci.* 9:1388. doi: 10.3390/app9071388

Mois, G., Sanislav, T., and Folea, S. C. (2016). A cyber-physical system for environmental monitoring. *IEEE Trans. Instrum. Measur.* 65, 1463–1471. doi: 10.1109/TIM.2016.2526669

Platt, R. Jr., Tedrake, R., Kaelbling, L., and Lozano-Perez, T. (2010). Belief space planning assuming maximum likelihood observations," in *Proceedings of Robotics: Science and Systems* (Zaragoza). doi: 10.15607/RSS.2010.VI.037

Reynolds, C. W. (1987). *Flocks, Herds and Schools: A Distributed Behavioral Model*. New York, NY: ACM. doi: 10.1145/37401.37406

Sachse, A., Liao, Z., Hu, W., Dai, X., and Kolditz, O. (2018). *Chinese Water Systems: Volume 2: Managing Water Resources for Urban Catchments: Chaohu. Terrestrial Environmental Sciences*. Cham: Springer International Publishing. doi: 10.1007/978-3-319-97568-9

Schreiber, M. P., Komppa, V., Wahlström, M., and Laine-Ylijoki, J. (2006). Chemical and environmental sampling: quality through accreditation, certification and industrial standards. *Accr. Quality Assur.* 10, 510–514. doi: 10.1007/s00769-005-0045-0

Sebastian, A., Sivagurunathan, S., and Ganeshan, V. M. (2018). "IoT challenges in data and citizen-centric smart city governance," in *Smart Cities*, ed Z. Mahmood (Cham: Springer), 127–151. doi: 10.1007/978-3-319-76669-0_6

Shafi, U., Mumtaz, R., Anwar, H., Qamar, A. M., and Khurshid, H. (2018). "Surface water pollution detection using Internet of Things," in *2018 15th International Conference on Smart Cities: Improving Quality of Life Using ICT & IoT (HONET-ICT)* (Islamabad: IEEE), 92–96. doi: 10.1109/HONET.2018.8551341

Shi, W., Zhao, X., Han, Y., Che, Z., Chai, X., and Liu, G. (2016). Ocean acidification increases cadmium accumulation in marine bivalves: a potential threat to seafood safety. *Sci. Rep.* 6:20197. doi: 10.1038/srep20197

Shimshack, J. P. (2014). The economics of environmental monitoring and enforcement. *Annu. Rev. Resour. Econ.* 6, 339–360. doi: 10.1146/annurev-resource-091912-151821

Soprani, M., Ponzoni, A., Sberveglieri, V., Carmona, E. N., Bhandari, M., Betto, G., et al. (2017). "Real-time chemical gas sensing of pathogenic microorganisms pollution in wastewater," in *2017 ISOCS/IEEE International Symposium on Olfaction and Electronic Nose (ISOEN)* (Montreal, QC: IEEE), 1–3. doi: 10.1109/ISOEN.2017.7968877

Soysal, O., and Sahin, E. (2005). "Probabilistic aggregation strategies in swarm robotic systems," in *Swarm Intelligence Symposium, 2005, SIS 2005* (Pasadena, CA: IEEE), 325–332. doi: 10.1109/SIS.2005.1501639

Strobel, V., Castelló Ferrer, E., and Dorigo, M. (2018). "Managing byzantine robots via blockchain technology in a swarm robotics collective decision making scenario," in *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018)* (Stockholm), 541–549.

United Nations Publications (2018). *SDG 6 Synthesis Report 2018 on Water and Sanitation*, Vol. 6. United Nations Fund for Population Activities, New York, NY.

Wang, W., Zhang, R., Yang, C., Kang, H., Zhang, L., and Yan, Y. (2018a). "Intelligent water environment monitoring system," in *Proceedings of the Fifth Euro-China Conference on Intelligent Data Analysis and Applications. ECC 2018* (Cham: Springer), 708–714. doi: 10.1007/978-3-030-03766-6_80

Wang, Y., Engel, B. A., Huang, P., Peng, H., Zhang, X., Cheng, M., et al. (2018b). Accurately early warning to water quality pollutant risk by mobile model system with optimization technology. *J. Environ. Manage.* 208, 122–133. doi: 10.1016/j.jenvman.2017.12.006

Wang, Y., Zhang, W., Engel, B. A., Peng, H., Theller, L., Shi, Y., et al. (2015). A fast mobile early warning system for water quality emergency risk in ungauged river basins. *Environ. Model. Soft.* 73, 76–89. doi: 10.1016/j.envsoft.2015.08.003

World Health Organization (2017). *Guidelines for Drinking-Water Quality, 4 Edn*. Geneva: World Health Organization.

World Water Assessment Programme (2019). *The United Nations World Water Development Report 2019: Leaving No One Behind*. UNESCO Publishing, Paris.

Xu, X., Pautasso, C., Zhu, L., Gramoli, V., Ponomarev, A., Tran, A. B., et al. (2016). "The blockchain as a software connector," in *2016 13th Work. IEEE/IFIP Conf. on Soft. Arch. (WICSA)* (Venice: IEEE), 182–191. doi: 10.1109/WICSA.2016.21

Zereik, E., Bibuli, M., Bruzzone, G., Gagliardi, F., and Bonsignorio, F. (2014). "Motion planning in the belief space for compliant behaviour of a diver companion robot," in *2014 IEEE Int. Conf. on Robotics and Biomimetics (ROBIO 2014)* (Bali: IEEE), 1933–1938. doi: 10.1109/ROBIO.2014.7090619

Zereik, E., Gagliardi, F., Bibuli, M., Sorbara, A., Bruzzone, G., Caccia, M., et al. (2015). "Belief space planning for an underwater floating manipulator," in *Int. Conf. on Computer Aided Sys. Theory* (Las Palmas de Gran Canaria: Springer), 869–876. doi: 10.1007/978-3-319-27340-2_106

Conflict of Interest: FB was employed by the company Heron Robots. SL was employed by the company Airalab. VB was employed by the company M2M Economy, Inc. ("Merklebot").

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Robotics and AI

REVIEW
published: 14 July 2020
doi: 10.3389/frobt.2020.00091

Interoperability Among Unmanned Maritime Vehicles: Review and First In-field Experimentation

Riccardo Costanzi[1], *Davide Fenucci*[2], *Vincenzo Manzari*[1,3]*, *Michele Micheli*[4], *Luca Morlando*[4], *Daniele Terracciano*[1,3], *Andrea Caiti*[1], *Mirko Stifani*[3] and *Alessandra Tesei*[4]

[1] DII (Dipartimento di Ingegneria dell'Informazione), Università di Pisa, Pisa, Italy, [2] Marine Autonomous & Robotic Systems, National Oceanography Centre (NOC), Southampton, United Kingdom, [3] CSSN (Centro di Supporto e Sperimentazione Navale), Italian Navy, La Spezia, Italy, [4] NATO STO CMRE (Science & Technology Organization—Centre for Maritime Research and Experimentation), La Spezia, Italy

Complex maritime missions, both above and below the surface, have traditionally been carried out by manned surface ships and submarines equipped with advanced sensor systems. Unmanned Maritime Vehicles (UMVs) are increasingly demonstrating their potential for improving existing naval capabilities due to their rapid deployability, easy scalability, and high reconfigurability, offering a reduction in both operational time and cost. In addition, they mitigate the risk to personnel by leaving the man far-from-the-risk but in-the-loop of decision making. In the long-term, a clear interoperability framework between unmanned systems, human operators, and legacy platforms will be crucial for effective joint operations planning and execution. However, the present multi-vendor multi-protocol solutions in multi-domain UMVs activities are hard to interoperate without common mission control interfaces and communication protocol schemes. Furthermore, the underwater domain presents significant challenges that cannot be satisfied with the solutions developed for terrestrial networks. In this paper, the interoperability topic is discussed blending a review of the technological growth from 2000 onwards with recent authors' in-field experience; finally, important research directions for the future are given. Within the broad framework of interoperability in general, the paper focuses on the aspect of interoperability among UMVs not neglecting the role of the human operator in the loop. The picture emerging from the review demonstrates that interoperability is currently receiving a high level of attention with a great and diverse deal of effort. Besides, the manuscript describes the experience from a sea trial exercise, where interoperability has been demonstrated by integrating heterogeneous autonomous UMVs into the NATO Centre for Maritime Research and Experimentation (CMRE) network, using different robotic middlewares and acoustic modem technologies to implement a multistatic active sonar system. A perspective for the interoperability in marine robotics missions emerges in the paper, through a discussion of current capabilities, in-field experience and future advanced technologies unique to UMVs. Nonetheless, their application spread is slowed

OPEN ACCESS

Edited by:
Enrica Zereik,
National Research Council (CNR), Italy

Reviewed by:
Ricardo Sanz,
Polytechnic University of Madrid, Spain
Antonio Vasilijevic,
University of Zagreb, Croatia

***Correspondence:**
Vincenzo Manzari
vincenzo.manzari@marina.difesa.it

Specialty section:
This article was submitted to Robotic Control Systems, a section of the journal Frontiers in Robotics and AI

Received: *15 September 2019*
Accepted: *05 June 2020*
Published: *14 July 2020*

Citation:
Costanzi R, Fenucci D, Manzari V, Micheli M, Morlando L, Terracciano D, Caiti A, Stifani M and Tesei A (2020) Interoperability Among Unmanned Maritime Vehicles: Review and First In-field Experimentation. Front. Robot. AI 7:91. doi: 10.3389/frobt.2020.00091

down by the lack of human confidence. In fact, an interoperable system-of-systems of autonomous UMVs will require operators involved only at a supervisory level. As trust develops, endorsed by stable and mature interoperability, human monitoring will be diminished to exploit the tremendous potential of fully autonomous UMVs.

Keywords: autonomous underwater vehicle, marine robotics, NATO experimentation, robotic middleware, unmanned vehicles interoperability, Unmanned Maritime Vehicles

1. INTRODUCTION

Unmanned Maritime Vehicles (UMVs) technology is increasingly demonstrating its potential to enhance existing naval capabilities, relying heavily on aircraft, helicopters, surface ships, and submarines to perform complex tasks. The integration with easily deployable, scalable systems of multiple UMVs offers an improvement in operation time, reduction of cost and mitigation of risk to personnel by leaving the man far-from-the-risk but in-the-loop of decision making. The achievement of the full potential of unmanned and autonomous systems must take into account the necessity of multi-national, multi-domain operations with multi-vendor, multi-protocol systems. The design of a clear framework for the interoperability of systems, both among them and with human operators, is essential toward effective planning and success of joint operations. On the other hand, UMVs in complex operational experimentations are hard to operate as an organic system-of-systems due to the expansion of non-standard solutions for mission control interface of the UMVs. In addition, the underwater domain poses significant communication challenges, such as multipath arrival structure, channel spread, and low data exchange rates.

In this complex framework, the definition of *Interoperability* is a tricky task *per se*. In order to share a common understanding, the National Institute of Standards and Technology definition is taken as a benchmark (Huang, 2004); the *Interoperability* is the ability of software or hardware systems to operate together successfully with minimal effort by the end-users, and it can be categorized into levels, types, or degrees of interoperability. It is pointed out that full interoperability would be facilitated by common or standard interfaces that are missing nowadays. section 2 and references therein provide an overview of the state-of-the-art of interoperability among UMVs, not neglecting the role of the human operator in the loop within the specific maritime domain, setting up a fundamental background for the reader.

The interoperability issue was approached in the Anti-Submarine Warfare—Operational Deployment of Concepts 2017 (ASW-ODC17) sea trial exercise, conducted in October 2017 off the coast of La Spezia (Italy). The sea trials were organized in the context of the Centre for Maritime Research and Experimentation (CMRE) of the NATO project Maritime Unmanned Systems (MUS) for ASW, involving NATO Naval Units and the Italian SEALab consortium. The SEALab is a joint laboratory between the Naval Support and Experimentation Center of the Italian Navy and the Italian Interuniversity Center of Integrated Systems for the Marine Environment (Terracciano et al., 2019). The goal of the MUS project is the development and verification at sea of a heterogeneous autonomous ASW network based on UMVs implementing a multistatic active sonar system. From the Italian point of view, the goal was to demonstrate the interoperability of a national Autonomous Underwater Vehicle [AUV Folaga WAVE, Fenucci et al. (2016)] within the CMRE robotic network for ASW (LePage et al., 2015) during a NATO operational exercise with assets of different NATO Navies. In-depth descriptions of the experimentation, high-level systems architectures and related software, and the specific interoperability experimental results are given in section 3.

Section 4 discusses the future challenges of interoperability, defining the current critical problems in marine robotics. Section 5 draws conclusions about interoperability among UMVs, merging the research advancement made over the past 20 years with the expertise of the authors and relevant guidelines for the future.

2. INTEROPERABILITY BACKGROUND AND RELEVANT LITERATURE

Sensors, platforms, software, and vehicle technologies are rapidly evolving, as well as processing and algorithm development, often outpacing the operational community capability to apply these new concepts in the field. In order to provide a broad view of the state of the art of interoperability, summaries—to be deepened with the cited bibliographical references—are provided below for the following topics:

- Adaptive Autonomous Communications and Networking;
- Command and Control System (C2S) and UMVs system-of-systems;
- Verification, Validation and Accreditation (VV&A) along with Modeling and Simulation (M&S);
- Interoperability Standardization;
- Robotics Middlewares.

It is necessary to stress the fact that these topics are all interconnected: only a synergic development of all of them enables a high level of interoperability.

Nowadays, research on *acoustic networking* is very engaged in supporting cooperative multi-vehicle missions which are increasingly dependent on the vehicles ability to inter-communicate. This must be accomplished exploiting and fusing the well-characterized Radio-Frequency (RF) channel with the time and space varying acoustic one. Ensuring the correct reception of a low bandwidth underwater acoustic signal affected by heavy delays and multipath interference is very challenging and error-prone, and it may result in limited

interoperability among UMVs (Stojanovic, 2006). Recently, Caiti et al. (2012) provided a remarkable illustration of a persistent acoustic communication network with heterogeneous platform and sensors, both fixed and mobile; Been et al. (2010) presented collaborative distributed ASW operations performed by a scalable and autonomous networking system from a comprehensive scientific and end-user point of view.

Communications are necessary to address the present knowledge representation that is still embryonic and is intended for basic single platform and single domain applications. This limits the possibility of multiple coordinated mission between UMVs, i.e., they essentially collect data from sensors. In addition, the data collected during the mission are then typically processed offline. However, greater autonomy requires distributed service-oriented agents that need access to higher data representation levels. To our knowledge, the work reported in Miguelañez et al. (2011) was the first example of online underwater mission adaption thanks to a goal-based planning using semantic representation. A semantic-based framework was presented in this paper, which provides the central architecture for the representation of information in embedded autonomous agents. A pool of hierarchical ontologies to represent the information derived from the sensor data is used in the proposed architecture. The key benefit is that service-oriented agents can have access to various types of knowledge and can also contribute to its advancement in an interoperable way. For example, if the required information is not accessible to an agent due to poor communication in case of unfavorable underwater acoustic channel, the architecture provides the facility to request that the information be produced by other agents with the appropriate capabilities. The framework was also validated and assessed in a Mine Counter Measure (MCM) scenario, where ontological information representation, model-based diagnostics and adaptive mission techniques are integrated.

From the point of view of reconfigurable and adaptive communication networks, it is important to recall the SUNRISE (Sensing, monitoring and actuating on the UNderwater world through a federated Research InfraStructure Extending the Future Internet, http://fp7-sunrise.eu) European project (Braga et al., 2016). The SUNRISE consortium has, in particular, established an abstraction layer which enables the interaction between networking and communication components and the control software of different UMVs. Any networking or control program may use this interface mechanism, defined as Software-to-Software Communication (SSC), and an XML document (eXtended Markup Language) is used to define command structure and semantics. This method aims to combine control software with underwater communication and networking elements such that underwater networks can be more dynamic, versatile and efficient. The SSC protocol was fully tested and assessed in lab for all the robotics middlewares mentioned in section 2. In 2014 and 2015, sea experiments have also been performed in Porto, the Atlantic Ocean and the Mediterranean Sea. The SUNRISE redeployable testing facility was robust, simple to use and highly adaptable to different requirements, and a network of up to eight heterogeneous nodes were deployed during those sea trials. The SUNRISE open architecture allows additional hardware [e.g., sensor(s), battery pack(s), modem(s), external disk(s)], requested by the mission, to be quickly fitted on every node of the testbed. SUNRISE was one of the biggest demonstrations of the capabilities provided by the forthcoming Underwater Software-Defined Open-Architecture Modem (SDOAM) framework. The current state of these developments mainly involves academic and industrial R&D prototypes, while most of the commercial modems currently available are not "open" for reconfiguration and user programming (Dol et al., 2014), i.e., their physical-layer algorithms are hardcoded in the modem firmware. Since 2011, the CMRE has been promoting the introduction of SDOAMs and recently published a study outlining SDOAM development and deployment activities, as well as future directions (Potter et al., 2014). Part of the CMRE activities was made in the SUNRISE project. The CMRE SDOAM concept includes a policy engine that handles several protocols for all layers of the OSI stack. This, in effect, was a starting point for the evolution of cognitive architectures. In fact, the CMRE communication stack is evolving in a fully cognitive communications architecture (CCA) that uses intelligent, adaptive and secure underwater networking techniques (Petroccia et al., 2018).

A further element in interoperability studies is *the design of C2S* to support autonomous collaborative tasks. In the area of adaptive control of heterogeneous UMVs to find suitable solutions for their interoperability, the Massachusetts Institute of Technology (MIT) research team has developed a uniform approach (the Generic Ocean Array Technology Sonar—GOATS program Bovio et al., 2001) to command all the assets through a hierarchical structure capable of ensuring the data propagation in the entire network (Benjamin et al., 2010; Schneider and Schmidt, 2010). Alongside the advances in underwater acoustics communications, both in-field and analytical works on UMVs swarms cooperation have made significant progress with several initiatives supported by the European Union, e.g., the "Cooperative Cognitive Control for Autonomous Underwater Vehicles" (Co3AUVs) project (Birk et al., 2011).

EU projects such as GREX (Kalwa, 2010) present this kind of approach for hydrographic mapping where the vehicle surveys the seabed or water column utilizing sonars and other sensors. However, there are certain shortcomings in such unmanned surveys, including the rate of acquisition and the limited swath width of the high-resolution sensors while working near the bottom. In order to maximize the spatial range of the sensors, GREX set up a team of vehicles that moves in formation and thereby broadens the operational swath width of the system, demanding synchronized motion control, decentralized decision-making, and inter-vehicle coordination. However, owing to the lack of advanced autonomy of the vehicles employed in the project, the pre-planned navigation configuration was restricted to more or less flat regions, although geologically or biologically significant areas typically present a wide variety of relief. The prevailing operational methodology to address this problem is the participation of human operators in the loop utilizing tethered vehicles.

The European project MORPH proposes a significant move forward on this issue (Kalwa et al., 2016). The concept is that a group of heterogeneous cooperative vehicles or self-propelled sensors carry out a multimodal survey of underwater structures. Spatial disparity provides the multiple points of view needed for both high-resolution surveys and the detection and prevention of obstacles. The separation between nodes enables various sensing systems to work according to their specific optimum range, e.g., Sonar nodes are farther away from targets than camera nodes. The diversity of vehicles even allows for a better combination of navigation and localization data, e.g., nodes adequately separated from interference structures may provide an external navigation guide to nodes near to these structures. More specifically, this decentralized and physically separated configuration of the system allows for morphing, i.e., the fleet will rapidly respond in real-time to perceived variations in the real world that can not be accounted for by *a priori*. MORPH explores and tests a range of interdisciplinary problems relating to fleet navigation and control, secure and efficient morphing, feedback and information affecting morphing, fleet design and knowledge sharing, C2S for useful mission management.

A successful interoperable C2S initiative was initiated in 2005 (Dias et al., 2005) by the Porto University—Underwater Systems and Technology Laboratory (LSTS). The NEPTUS architecture goal is to enable integrated operations of heterogeneous UMVs teams, operating sea, ground and air vehicles and individuals. People also play a key role in autonomous vehicles, where a mixed-initiative process is necessary. The operating situations for these teams are primarily environmental protection operations, but they may also include environmental disasters, rescue missions, etc. The distributed architecture of Neptus is a service-oriented architecture that enables high degrees of interoperability (between applications), scalability (number of nodes), and reconfiguration (number and kind of nodes). NEPTUS has been used in many at-sea tests such as the Rapid Environmental Picture Atlantic exercise 2014 (de Sousa et al., 2015). This experiment, involving more than 10 military and civil organizations, emphasized multi-domain missions in order to foster interoperability and cooperation between UMVs and aerial vehicles. The NEPTUS toolchain supplied an unified C2S that allowed the integration of numerous vehicle systems, enabling wireless and underwater interoperable communications and destructive delay-tolerant networking (DTN) capabilities.

More recently, the Widely scalable Mobile Underwater Sonar Technology (WiMUST) H2020 project (Abreu et al., 2016; Indiveri et al., 2016) validates at-sea a system of cooperative UMVs for geotechnical measurement and geophysical mapping. The new core innovation of the WiMUST framework is the use of a team of collaborative autonomous underwater robots, functioning as intelligent sensing and communication reconfigurable mobile acoustic network. The project brings together a community of academic organizations, geophysical survey firms and SMEs with an established track record in autonomous adaptive technologies, networked cooperative control and navigation, and marine robot architecture and manufacturing. Bottom surveys are nowadays collected from side-scan or multi-beam sonars, which are towed from ships or embarked on autonomous vehicles. The WiMUST concept offered a technological breakthrough in the development of robotic distributed sonar system with autonomous mobile nodes, making operation at sea much easier given the fact that no physical connection exists between the surface ship and the acquisition equipment. The final demonstration in the Atlantic Ocean involved ten heterogeneous vehicles but all the technologies designed during the project were conceived and implemented with a specific long term vision: underwater missions performed by a large number of autonomous cooperating robots.

Moreover, the system presented in Robb et al. (2018) dealt with the difficulty of monitoring multi-objective, multi-vehicle operations, while at the same time resolving the ambiguity about the actual status and the protection of distant, high-value platforms. In order to increase the reliability and efficacy of UMV C2S, the authors suggested a hybrid of an interactive, natural language operator interface coupled with communications using multi-domain channels to transmit data through various devices and delivery modes, enhancing C2 of coordinated and ultimately autonomous missions. The Multimodal Intelligent inteRactIon for Autonomous systeMs (MIRIAM) natural language interface enables operators to straightforwardly update an autonomous network about the progress of the mission goals and the state of the AUVs assigned to it (Hastie et al., 2017). MIRIAM has the capacity of connecting to commercial C2 and applications, collects continuously updated task and vehicle data, acknowledges user requests, supply outputs, and produces its own messages of significant issues in natural language. They have successfully demonstrated their interoperable systems at sea using the OceanServer IVER-3 AUV as the vehicle to be operated and tracked, the EvoLogics Sonobot USV as the communication gateway and the Seebyte Seetrack-Neptune C2 program with the MIRIAM natural language interface in the on-shore C2S.

Last but not least, all the models, hardware and software composing a system-of-systems must pass a *VV&A process* (Hodicky, 2014). The article underline how the analysis of potential integrations of an Autonomous System (AS) into the operational field must be a priori tested to keep costs low, i.e., through M&S systems for experimentation in synthetic distributed environments. These environments are based on cooperating entities using data interchange mechanisms such as the High Level Architecture (HLA) (Möller et al., 2008). It is a suitable candidate due to its maturity and broad adoption (e.g., it is the only distributed simulation framework accepted as a NATO standard—STANAG 4603). Its most recent and major improvement is the Federation Object Model (FOM), which acts as a shared vocabulary for communication between M&S systems. The creation of an AS common vocabulary in M&S can even increase the potential of the synthetic experimental framework to ensure the easiest and most effective way to implement AS in the operational field. Although there are also tight industry standards for networked system, there are no exhaustive methods to lead the researchers through the VV&A cycle for the evolution of autonomous interoperable systems. In order to ensure coherent outcomes in all network

simulation configuration, new practices are necessary because of the nature of the system-of-systems relationships in a decentralized environment which simulates exchanges between autonomous assets (Hodicky, 2018). For VV&A of modular network of systems, (Tremori et al., 2018) introduces an agile M&S architecture which enables better knowledge of the whole system by testing together all its component (i.e., hardware and software) in virtual-reality environments that are operationally meaningful. The functional architecture suggested adheres to the most recent Institute of Electrical and Electronic Engineers (IEEE) recommended practice for VV&A (IEEE, 2007) and NATO guidelines in the sector (Ruiz et al., 2016).

Finally, it is worth to mention that the CMRE started a multi-year project in 2014 entitled Persistent Autonomous Reconfigurable Capability (PARC) aimed at assisting NATO in preparing the future in this domain and addressing common technological shortfalls, cost aspects and challenges related to the transition of this type of technology. The objectives of PARC include increasing maritime unmanned systems' persistence, *interoperability*, scalability while addressing standardization, information assurance, and cost aspects. Two examples of remarkable results can be found in Carrera et al. (2016) and Petroccia et al. (2018). In the first, an HLA connection between simulated assets and an autonomous system using the Robot Operating System (ROS) middleware is provided, enabling both M&S and robotics researchers to develop more complicated and accurate simulations of operationally relevant environments. The latter reference is a description of the AutoLARS (Launch And Recovery System) system which allows AUV docking, wireless battery charging and high data-rate download of collected data with the ultimate aim of improving the persistence of these systems far beyond their batteries limits.

Besides PARC, CMRE is constantly involved in promoting *Standardization Agreement (STANAG) also about interoperability*. Among NATO members, a STANAG establishes methods, requirements and constraints for operations and systems. The aim is to provide joint procedures and logistics so that the military of one Member State can interoperate easily with the others. STANAGs are also the foundation for interoperability between a different Information and Communication Technology (ICT) systems, which is crucial for NATO and allied missions. Two of them are very important and worth mentioning in this paper: 4586 and 4748.

First, the STANAG 4586 (Marques, 2012) is the current standard for Unmanned Aerial Vehicles (UAV) (Platts et al., 2007; Frazzetta and Pacino, 2013) and it is in place a study group on Multi-Domain Control Station (MDCS) which aims at going toward a "joint" standard, i.e., one that will cover air, land and maritime unmanned systems. The MDCS working group "a mixed industrial and government representatives group" will likely turn its ideas into a STANAG (i.e., an updated version of the STANAG 4586) in the future. Secondly, CMRE, together with academia and industry, created an underwater communications protocol known as JANUS (Potter et al., 2014), recently advertised as NATO STANAG 4748 (NSO, 2017). It is the first globally accepted and openly accessible communications standard for all the communities of the underwater domain (http://www.januswiki.com). It is built on the binary frequency change with tunable center frequency and bandwidth. The subsequent bit rate is 80 bps if the default channel is selected (9.4 − −13.6 kHz band). Automatic Identification System (AIS), meteorological and oceanografical data transfer to submarines, other than assistance in distressed submarine operations, are distinctive functions of JANUS (Alves et al., 2016; Petroccia et al., 2016, 2017). Thanks to the use of a standardized approach, it will be possible to raise the level of Maritime Situational Awareness (MSA) as well as to enhance security and water-space governance employing heterogeneous and hybrid systems, both manned and unmanned, including collaborative UMVs networks. In conclusion, JANUS is the first comprehensive solution that makes it possible to standardize communication protocols at the physical level between multi-vendor devices, acting as a fundamental *glue* between the existing proprietary protocols.

Another very active player in the interoperability field is the European Defense Agency (EDA) since its 2008 Unmanned Maritime Systems (UMS) programme (EDA, 2011; Dahlmann et al., 2015), which established a list of key-technologies needed for the appropriate functioning of the UMVs irrespective of naval application. The whole programme aims at coordinating efforts from individual member states to foster interoperability, safety and more broadly the use of UMS. A specific UMS-project has been launched (called STANDIN: Standards and Interfaces for more interoperable European UMS) to take into account information on standards/interfaces from UMS-projects. The STANDIN project aim is to provide a relevant recommendation, identifying issues that may hinder the eventual achievement of the UMS-programme objectives. The endorsement of the recommendation of the STANDIN project should enhance innovation (use of common interfaces/standards to enable industries to produce components to be easily integrated and tested on UMS), upgradability of UMS and plug & play. However, the recommendation is not expected to be translated into new regulated standards but instead it will depend on the will of governments and industry to enforce the recommendation, i.e., adopting it as a "de-facto standard" for European UMS.

The last but fundamental element of interoperability is the adoption of a specific *robotics middleware*. These intermediate software level is the fulcrum of interoperability between UMVs. A robotics middleware can essentially be thought as a software layer, composed of several modular packages, which has the task of collecting and harmonizing the information coming from the on-board sensors, making them available to the various processing nodes. These nodes must in turn process these data according to their specific function (tracking, communications, etc.) and pass the outputs to other nodes or directly to sensors and/or actuators of the robotic system. At the end of the various elaborations, the middleware will be the responsible for the passage of the system in another state, e.g., the execution of a specific action.

The current *de facto* standard middlewares for UMVs include MOOS [11] and ROS [12]. They are both publish-and-subscribe systems, which provide the communication of arbitrary data throughout a network. However, in order to complete mission objectives, a robotic system also requires a

deliberative component in addition to its reactive aspects (e.g., avoiding obstacles) (DeMarco et al., 2011). This is the goal of the MOOS process IvP-Helm, which act as an autonomous decision-making engine that executes in the backseat of the robotic platform (Benjamin et al., 2010). In fact, the MOOS-IvP is a combination of two components, IvP Helm and MOOS. While MOOS is the actual robotics middleware, abstracting a TCP/IP based inter-process communication (IPC) protocol, IvP is a multi-objective optimization framework for constructing complex autonomous behaviors from a collection of more simple routines. The whole environment is composed of several other processes, which communicate using the MOOSDB database as a broker. The rate of interactions across information channels, called topics, is controlled tightly by the MOOSDB, occurring synchronously at a predefined rate. The exchange of information occurs between TCP ports, which may or may not exist on the same physical machine. Therefore, MOOSDB is relatively rigid, and each topic is bound to a startup-defined data type, with induced additive latencies because of the travel path of information (node-database-node).

On the other hand, ROS does not impose any architectural constraints, i.e., processes (called nodes) communicate directly with each other, with no central broker. A central node (called master) exists, but only to manage node startup and shutdown. This is the main differences between MOOS and ROS: while all data within a MOOS system are transmitted through the MOOSDB, in ROS data are transferred using peer-to-peer communications. The basic unit of interaction in ROS is a message, which is typically exchanged on a topic: from a node's perspective, messages are published synchronously and read from a subscription asynchronously. Furthermore, a single topic can contain multiple instances of a message. In ROS, there are two extensions to the typical publish/subscribe mechanism: *services* and *actions*. A service simply takes an input and returns an output, as a typical service does in computer science literature. An action keeps an internal state on a longer time scale because it is called with a goal, emits feedback during the action, and finally returns a single result at the end of the action. An important advantage of ROS over MOOS is its capability of handling very large datasets, e.g., live high-resolution video and multi-dimensional point clouds from LIDAR-like devices.

The development of MOOS-IvP in the research community is continuous and ongoing, while its usage in the industry is very limited. This may be a consequence of the lack of insurance on backward compatibility that is based on community agreement, not on enforced standard (even if newer versions are generally backward compatible). While MOOS has historically been popular within the underwater robotics community, ROS is now by far more pervasive in a multi-domain context (ground, sea, air). The main reasons for such success are the reconfigurability and ease of use of ROS. There are bindings for both C++ and Python, and it is regularly updated (at least yearly). There are no rigid standards guiding development, but it is so widely used that there is extreme prejudice against breaking backward compatibility.

Another popular middleware in the marine robotics community is DUNE (DUNE: Uniform Navigation Environment) mainly due to its usage in autonomous vehicles designed by the LSTS Group at the University of Porto (Pinto et al., 2013). DUNE offers a C++ programming framework for robust and flexible real-time reactive operations, and also uses the publish/subscribe method. DUNE modules, named tasks, publish and subscribe messages without requiring any specific knowledge of the other tasks. For most instances, basic interface modifications are sufficient to implement new features. Communication between tasks is carried out solely through a message-oriented Inter-Module Communication (IMC) protocol for UMVs and sensor networks (Martins et al., 2009).

Through an industrial point of view, the Common Object Request Broker Architecture (CORBA) is worth considering as a middleware standard (https://www.corba.org/). It was launched in 1991, supported by the Object Management Group (OMG) and commonly adopted by major organizations such as Thales, Raytheon, and BAE. The current edition dates back to the end of 2012. Several implementations are accessible, such as omniORB, PrismTech, and RT-CORBA, which share the primary strength of CORBA: framework and applications are separate, enabling vendors to interoperate based on the interface description language (IDL) specification (Henning, 2008). The successor to CORBA is the Data Distribution Service (DDS), an IPC standard specification initiated in 2004 by the OMG standardization committee (a description of the DDS standards can be found at https://www.omg.org/spec/category/data-distribution-service/). There are several industrial implementations relevant to companies involved in autonomous vehicles or ground control networks. The DDS is a language-agnostic IPC standard, and it does not recommend a specific implementation of any sort neither a device communication framework. DDS is a standard that is preserved and strengthened by OMG, an agency that carries a great deal of weight in the industry. However, as a successor to CORBA, the development of industrial migration to the newer standard is not obvious. Additionally, although the specification is fully available, most of the implementations are private, with no outstanding candidates in the open-licensed domain.

Finally, the Joint Architecture for Unmanned Systems (JAUS) is worth noting as a concrete middleware standardization initiative. JAUS is a specification directed at unmanned systems, introduced by the US DoD to provide a basis for interoperability between unmanned systems (Whitsitt and Sprinkle, 2011). In order to guarantee that the device design is valid to the whole domain of existing and future unmanned systems, the JAUS Reference Architecture (RA) was focused on five principles: vehicle platform independence, task autonomy, computer hardware independence, technology independence and operator independence. This design has passed from the JAUS Working group, which consisted of individuals from government, industry and academics, to the Society of Automotive Engineers (SAE). The Technical Committee of the SAE Unmanned Systems now retains and promotes the collection of standards. Most specifications have been transferred from the JAUS Reference Architecture to a services-based system including

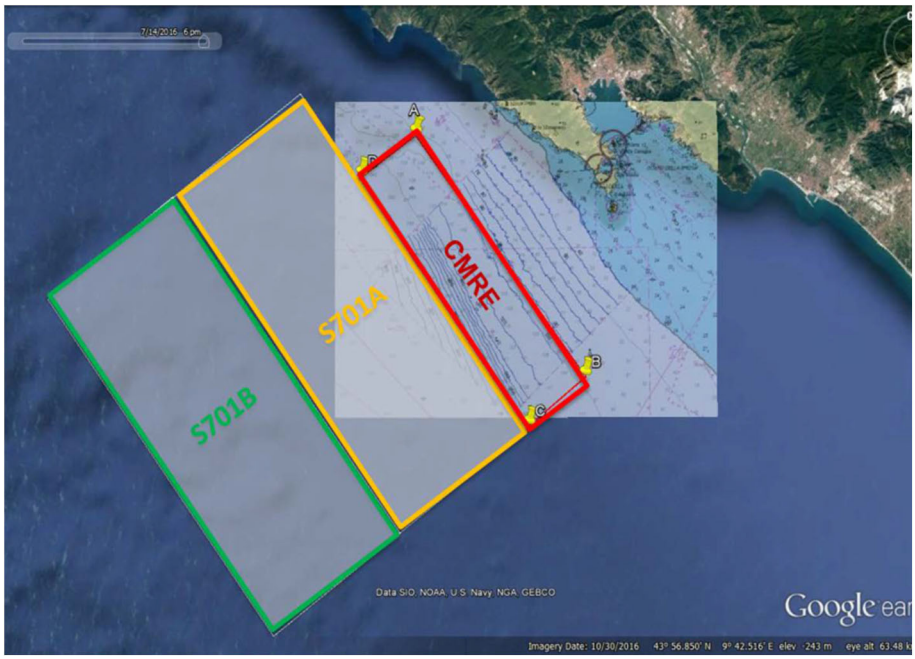


FIGURE 1 | Map of the area off Cinque Terre, La Spezia, Italy, requested for ASW-ODC17 sea trials (snapshot from Google Earth) in the period 12–17 OCT 2017.

for example AS5669 (JAUS Transport Standard) and AS5710 (JAUS Core Service Set). In particular, the AS5669 (https://www.sae.org/standards/content/as5669a/) specifies the transport layer between processes (header compression, source/destination address, TCP, UDP, or serial link) but does not identify lower-layer (data link-physical) operations and is therefore independent of the medium from which the messages are sent. This characteristic improves interoperability between distributed networks and inter-node (or vehicle) communications. Before the "official" SAE JAUS version, the JAUS RA was developed to provide first-time developers with an open implementation that would be comparable to the norm. However, when the first pay-for edition of SAE JAUS was released, there was considerably less open-source support. Several of the initial free implementations compliant with JAUS RA have been scrapped, and others have vanished behind new proprietary licenses and other pay-as-you-go systems. To sum up, the JAUS standard claims to be well-written, but it has not been broadly implemented in the underwater scientific field because the standard and several of its applications are proprietary, raising obstacles to entry and reducing visibility for research groups.

3. AT-SEA EXPERIENCE: THE ASW OPERATIONAL DEPLOYMENT OF CONCEPTS '17

This section briefly summarizes the work presented in Costanzi et al. (2018): besides describing how the *interoperability issue* was approached during an experimental campaign held in October 2017, it provides more information on the overall network and the deployed components to emphasize the complexity of the scenario and to underline the high level of interoperability reached.

The ASW-ODC17 experimentation aims, among other objectives, to demonstrate interoperability of different legacy and modern systems, like the Folaga WAVE glider-AUV, within the CMRE network. This kind of operational experimentation is vital to receive UMVs requirements from the end-users and to demonstrate to them the potential capabilities and challenges of integrating unmanned systems with legacy maritime assets. Although CMRE has always worked with other naval units on the margins of other operational exercises on a non-interfering basis, this is the first time that it has been allotted a dedicated period to conduct trials with the NATO standing forces. The ASW-ODC17 experimentation area was defined by the three boxes shown in **Figure 1**. Almost 20 nodes were involved during the experimentation, including static and mobile assets, both manned and unmanned. The remarkable extension of this network poses an actual interoperability challenge.

The backbone of the experimentation was the Network Enabled Modem Operator (NEMO): the CMRE software framework designed with MOOS-IvP to support experiments in mixed media (air and underwater) (Vermeij et al., 2015) using underwater communications in data exchanges between mobile and potentially autonomous nodes (unmanned to unmanned) and C2. NEMO represents a move in the direction of a more versatile marine communication infrastructure which will leave behind pAcommsHandler's heritage, that was the first IvP process operating the vehicle's acoustic modem (Freitag et al., 2005). NEMO provides a workspace where efforts such as JANUS, SDOAM, clock synchronization or routing can be deployed,

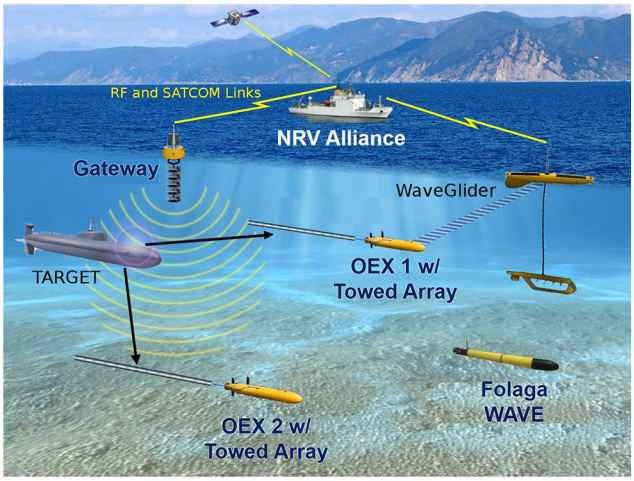


FIGURE 2 | A descriptive outline of the multistatic network for ASW of the CMRE with the integration of the Folaga-WAVE. In this ASW architecture, multiple active sonars are located on ships and buoys, and cooperate with multiple receivers, i.e., towed arrays installed on OEX AUVs and other manned assets. The communications between all the platforms, underwater and above the water surface, are ensured by the WaveGliders, acting as mobile gateways, together with the moored gateway buoys. The CMRE OEX AUVs, called Groucho and Harpo, are the mobile sonar receivers. The NRV Alliance is the network C2 centre and is also part of the network, towing both a source and an array. For the specific experimentation, the Folaga WAVE was equipped with a CTD probe, the data of which were made available for periodic updates of the environmental map in the area and for insertion into the acoustic engine, in which the onboard processed sound speed can be a valuable information.

tested and implemented. The NEMO is the current marine communication stack for CMRE research, which still prefers MOOS-IvP mainly because of the IvP part. It allows the vehicle to work autonomously toward a goal, while it operates within mission requirements, operating an efficient de-conflicting of the tasks. Anyway, due to their many similarities, there is almost a one-to-one correspondence between ROS and MOOS system calls, and the path forward for the Centre is the integration of the two middlewares. The next step will be the development of a cognitive communications architecture (CCA) that allows other channel access outside TDMA (Time Division Multiple Access), utilizing intelligent, adaptive and secure submarine networking strategies (Petroccia et al., 2018).

The physical cores of the CMRE's multistatic hybrid network for ASW were the active sonars (i.e., acoustic transmitters installed on a buoy and towed by NATO Research Vessel Alliance) and a typical scenario can be seen in **Figure 2**. In particular, the new triplet SLIm Cardioid Towed Array (SLICTA) array (Canepa et al., 2017) was towed from NRV Alliance along with the ATLAS Source and receiver array. When the transmitted pings are scattered by objects, receiving hydrophone arrays can collect those echoes from different positions, including the arrays towed by autonomous platforms (in particular two Ocean Explorer—OEX—AUVs). The two OEX AUVs owned by CMRE have been fitted with acoustic modems and they can tow an array developed and fitted for purpose, i.e., the SLICTA. CMRE has successfully tested—during ASW-ODC17—a paradigm for the allocation of robotic assets to ASW tasks, a problem called the Multi Robot Task Allocation (MRTA) (Ferri et al., 2017). In the underwater field, centralized network monitoring, which can enable tasks distribution optimization, is not feasible for the outlined problems and peculiarities of the channel, and this is why interoperability is so crucial. The suggested scheme of assignment operates in a fully federated manner and two simultaneous auctioning nodes handle the actual tasks. The CMRE MRTA utilizes adjacent node agreement and only needs local exchange of underwater data.

Also WaveGliders, UMVs that use wave motion to navigate (Willcox et al., 2009), were employed in the experimentation along with a set of deployed moored buoys to create a communications network that allows feedback, localization and exchange of control and information between manned and unmanned platforms (Munafò and Ferri, 2017). The Alliance has the role of C2S to allow the users to communicate in real-time with the network through multi-hop communications, via undersea or RF connections. This allows the Alliance to remain far from the patrolled area with the possibility to undertake additional operations. The main benefit of using various assets, active and passive, is the expansion of the network range employing the specific multistatic geometry to augment the probability of sonar identification with sonar signal processing.

During the experiment, the Italian Navy agreed to supply the Leonardo Coastal Research Vessel (CRV) equipped with a towed echo repeater to emulate an acoustic scatterer by retransmitting the sonar signal recorded from the source according to user-specified parameters (e.g., delay, attenuation, etc.) (Grimmett, 2009).

Finally, the WAVE vehicle—capable of navigating using wave motion and recharging with solar energy (Caiti et al., 2018)—was added to the network to stress interoperability and add significant data to the ASW network. The WAVE Mission Control System (WMCS) combines the modules specifically developed for the project with those already existing in the AUV and ensures a high level of abstraction for the user set-up of an autonomous mission (the conceptual scheme of the WMCS is illustrated in **Figure 3**). The term "high level" refers to the user not having direct control of the hardware (sensors and actuators) that are installed on the vehicle but interacting with them via the implemented WMCS request-response ROS-based mechanism.

In order to be effective, efficient and reliable, the WMCS has been designed and implemented foregrounding requirements such as modularity, scalability, reconfigurability, user-friendliness, and robustness. These requirements were set in the WAVE project to meet future interoperability need, as shown in section 4. The WMCS consists mainly of two subsystems:

- The Folaga Mission Control System (FMCS) onboard the vehicle, responsible for the management of mission payloads and interaction with the low-level control system of the Folaga vehicle;
- The WAVE C2S on the base station (positioned ashore or on a support vessel), which provides essentially a mission-managing level and a graphical user interface with all the tools to carry out mission supervision and control.

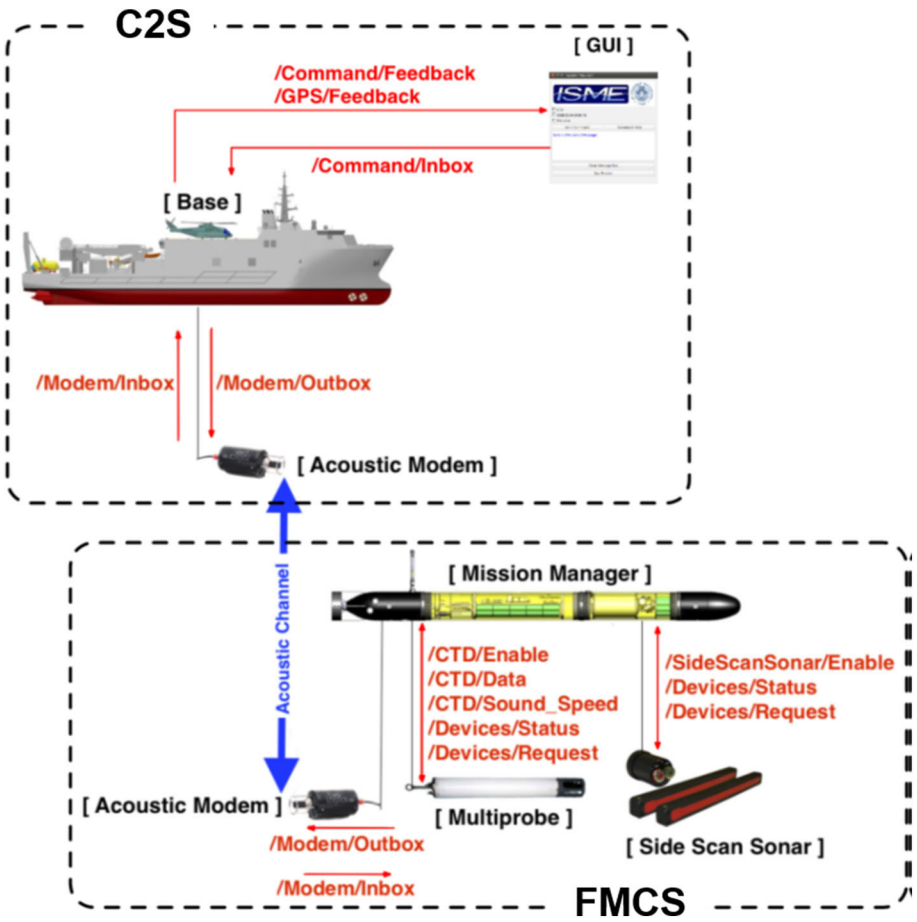


FIGURE 3 | Conceptual scheme of the WAVE Mission Control System (WMCS). The modularity of the WMCS allows both to displace nodes on different platforms in different domains (above, on, and under the surface) and to easily add new mission payloads simply by maintaining the interface and message architecture defined at system level. In fact, the [Acoustic Modem] and part of the [Base] ROS modules have been installed on the gateway buoy during the experimentation along with the ROS-MOOS bridge. Figure from Costanzi et al. (2018).

FIGURE 4 | Screenshots of real data during the experiments. On the left, the sound speed received acoustically in real time during the Folaga WAVE profiling. On the right, the CMRE C2S with the AUVs' positions displayed. Figure from Costanzi et al. (2018).

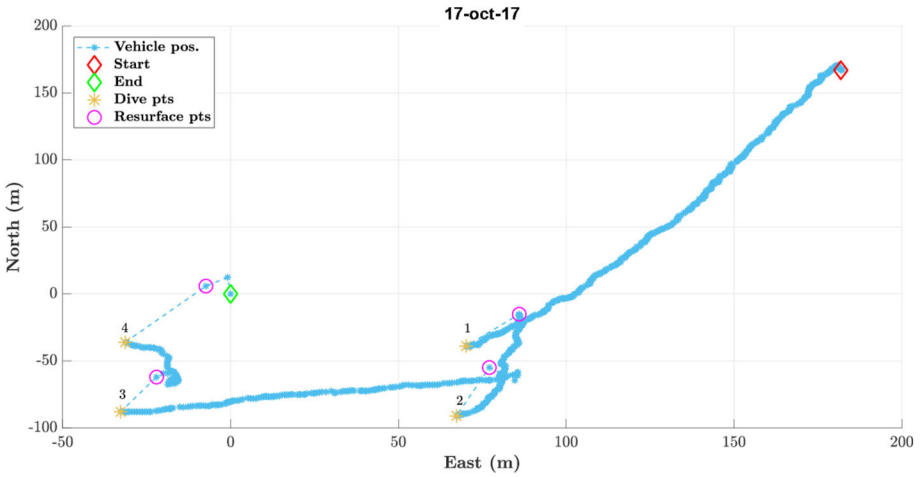


FIGURE 5 | The performed interoperability test mission in NED coordinates. The vehicle was moving in gliding mode for about 250 m in the first path between the starting point (red diamond) and the first dive point (orange asterisk). Then, it started doing four profiling tasks to characterize the water column down to 12 m depth on a rectangular area of 5,000 square meters. Recall that the vehicle simply dives vertically using the ballast and the internal moving mass for profiling the water column. Only the vertical jet-pumps are used for finely trimming the pitch angle to 0 degrees, i.e., it is free to drift during these kinds of missions. As it can be seen, it was present a strong sea current toward North-East so that the vehicle resurfaced about 30 – −40 m away from each diving points. Finally, the first path from the red-diamond to the dive point no. 1 was done in surface navigation in this specific plot. Figure from Costanzi et al. (2018).

Then, the WMCS implemented a distributed architecture: software modules were split between the WAVE C2S, the moored gateway buoy and the vehicle itself. While these systems have a physical separation, they merge acoustic communications (modems were installed on both the gateway buoy and the vehicle) with RF communications (between C2S and the gateway buoy, but in case also between C2S and the vehicle). This allows the user to send commands to the vehicle and receive required notifications and data. Indeed, the interoperability of the WAVE vehicle into the aforementioned CMRE heterogeneous network was demonstrated by exchanging both commands and data. In particular, the WAVE vehicle was equipped with a Conductivity, Temperature, Depth (CTD) sensor to obtain representative operational information that was communicated to CMRE Environmental Knowledge and Operational Effectiveness (EKOE) team (Grasso et al., 2016). Furthermore, CTD data has been distributed for periodic updates of the environmental map in the area and for injection into the MultiStatic Tactical Planning Aid (MSTPA) decision support tool (Strode et al., 2012), to which the sound speed measured by the vehicle can be relevant. Interaction with the EKOE team in particular enabled the view of the positions of all the underwater assets (CMRE OEX AUVs, WAVE, and CMRE Wavegliders) at the EKOE C2S. In addition, the position of the WAVE vehicle was made readily accessible on board the Flag-Ship of the NATO partners and at the command and control stations of NATO Allied Maritime Command (MARCOM). **Figure 4** shows the information as they were seen on board NRV Alliance.

As it is now clear, one of the main challenge lays in the different robotics middleware of the Folaga WAVE vehicle and the CMRE network which uses ROS and MOOS respectively. Therefore, a ROS-MOOS bridge software was installed on a moored buoy acting as a gateway between underwater and surface assets. Due to the characteristics of the different AUVs participating in the experimentation in addition to the WAVE vehicle, the gateway was fitted with acoustic modems working on different frequencies. This way, all the interoperability tests rely on a double channel communication: acoustic between the gateway buoy and the vehicle, RF between the Alliance C2S and the gateway buoy.

The complete mission done for all the considered experiments is shown in **Figure 5**. It is important to recall that all the tasks implemented on the AUV for the specific mission could be added, started, halted or terminated via acoustic modem or Wi-Fi. The integration of WAVE Folaga within the CMRE network, aimed at enhancing the interoperability in a multi-vehicle operation, was the best demonstration of the effectiveness of the system developed in the project in an operational context.

4. FUTURE TRENDS OF INTEROPERABILITY

System modularity and interoperability (between heterogeneous systems) are two keywords of the current world of marine robotics as an unprecedented growth of sensors, communication architectures and protocols, manned and unmanned platforms, and software at various levels is ongoing. One of the main reasons for their relevance in modern engineering developments is that modularity and interoperability ensure the reduction of the costs of a system—albeit large—along its entire life cycle, both from a maintenance point of view and from the possibility of

adaptation and evolution based on the growing applications and new technologies available in the future.

Today, several factors limit the interoperability of systems, not just UMVs. Among them, the main ones are certainly the proliferation of proprietary interfaces, often followed by a non-standard physical communication architecture characterized by non-shared waveforms, frequencies and settings. Besides, much of the engineering effort is now focused on standardizing the type of data, metadata and related encoding exchanged between systems.

The development of specific main techniques and technologies can tackle these changing requirements while mitigating the effect on the platform itself:

- *Multiformat aggregated data processing:* the capacity to concurrently communicate with, manage and elaborate different data formats. If it were possible to know beforehand the data type and information formatting exchanged (including metadata and labels), as well as being able to implement processing algorithms capable of handling different data formats agnostically, platforms can be then more synergistically updated in relation with emerging operational needs.
- *Federated Distributed computing:* the capacity to quickly change modules ("plug and play" manner), while barely physical requirements have to be addressed. Once a new payload has been physically installed in compliance with the requirements of modularity and interoperability, it is necessary to test, analyze and certify the behavior of the whole autonomous system with a high impact in terms of costs and development times. Federating the computing capabilities of the system up to the payload level or even to the remote control station could be an important lever to facilitate the rapid and effective integration of new technologies.
- *Open Standards, Architectures and Equipment will improve Interoperability:* the vision currently shared in the industrial world of UMVs is that only by developing new platforms, systems, payloads, software according to common standards make it possible to allow full interoperability of the various unmanned autonomous systems. At the same time, this will enable the interchangeability of the various modules of a platform respecting today's budget constraints. Once standards have been established, the massive use of Commercial-Off-The-Shelf (COTS) components will allow the sharing of various functional subsystems between different vehicles, such as payloads, navigation systems, power supply systems, communication systems, sensors, and launch and recovery systems. Hand in hand with the hardware, the software must also comply with a standard architecture that can facilitate changes to the configuration of a system when replacements or additions of entire vehicles or specific payloads are needed. This sharing of standards at each level between heterogeneous vehicles will have positive effects from the platform's acquisition until its disposal without loss of interoperability between systems. The larger and more expensive the UMV, the more customized and proprietary interfaces will be prohibitively expensive to develop and maintain, and therefore the greater the advantage associated with the use of open standards, architectures and equipment.
- *Collaborative, Opportunistic, Advanced Communications:* the communication required in modern applications is no longer merely a point-to-point communication. Today, we have flexible and adaptive networks working in a multi-channel environment that lays challenging constraints on the performances obtainable. In particular, one of the major limitations of the underwater acoustic channel is the low bandwidth, which requires a high level of discrimination in the exchange of information (necessity, extension, etc.), favoring cooperative strategies between UMVs to take advantage of their autonomous capabilities even in the case of data loss. Moreover, these peculiarities of the underwater acoustic channel impede the use of classic collaboration algorithms used in terrestrial and air domains based on the consensus theory (Ren et al., 2005). In fact, these algorithms require an information exchange overload for network management directly proportional to the network size. An alternative solution under maturity is the use of non-acoustic communication (mainly Light Emitting Diode systems and laser) at range below tens of meters, especially for transmitting high-speed data between close UMVs and between UMVs and node acting as gateway between the underwater and non-underwater environment (for example surface vehicles or buoys with satellite connections, Doniec et al., 2010); a remarkable example of such concept is the dual acoustic/optical modem of WHOI able to adaptively operate according to the relative distance between UMVs (Farr et al., 2010). WHOI was also a pioneer in the flexible use of the limited acoustic channel through the introduction of the Compact Control Language (CCL). CCL is a series of messages that contain UMV commands and data messages for standard sensors (Stokey et al., 2005). CCL commands include basic procedures such as "Abort Now" and "Abort to Mission Start" but also sophisticated commands such as a side-scan sonar redirection over the operational area. The open design of the specification enables vehicles produced by various academic organizations or commercial firms to work together using standard data formats. New signals can be introduced by users if required for new operations, both military and civilian. In addition to pure communication between UMVs, it is necessary to determine the position of nodes within a network, and this can be done with the same autonomous cooperative approach between UMVs: simulative and experimental results have been presented for example in Allotta et al. (2014), Ridolfi et al. (2018), and Masmitja et al. (2018). These recent works focused on underwater Multi-Target Tracking (MTT), evaluating the potential to use surface vehicles as mobile markers to locate and map a set of underwater vehicles. This collaborative solution minimizes the main downside to multi-target cooperation, which is the uncertainty in underwater positioning due to the environmental uncertainty. In the mentioned studies, various network levels of sensors, nodes, and vehicles operate together, opening new possibilities for detecting and understanding the complex dynamics of ocean phenomena and creating

new applications. In the civil sector, for example, there is an immediate and worldwide need for a technology that can allow environmental response teams to quickly identify the nature and magnitude of unintended leaks of toxic products to have an appropriate response. A cooperative robotic system, consisting of two heterogeneous UMVs, for environmental control is introduced in Vasilijević et al. (2017). The described hybrid surface-underwater architecture enables the operator to interpret the product concentration data in real-time, using the system's modeling and decision-making abilities, and to adjust the task on the move. The system is an application of the Human-on-the-loop (HOTL) concept (Cummings et al., 2012), which allows a minimal team of operators to manage a network of robotic agents working in complex, time-consuming environments. The tests demonstrated enhanced process efficiency for a network of autonomous vehicles in search, track, and neutralization missions. The HOTL principle is compared vs. the human-in-the-loop principle in Valavanis and Vachtsevanos (2015), highlighting the unique technological challenges and degrees of autonomy needed by autonomous vehicles to carry out a task without substantial human involvement. HOTL supports decision-making and helps the operator to conduct the most suitable task in an evolving scenario, i.e., to enable on-the-fly mission adjustment.

- *Interoperable, realistic integrated Modeling and Simulation environment:* approaches consisting purely of in-field experimentation in the marine robotics domain are prohibitively expensive; thus, the interoperability of UMVs and heterogeneous integrated platforms and systems will probably be critically assessed using modeling and simulation. The future trend of M&S may be focused on the design of a scalable architecture for interoperable simulation based on a system of systems approach, providing a V&V (Verification and Validation) capability to explore systems reliability in complex conditions and to analyze autonomous behaviors in cost effective and safe virtual environments (Hodicky, 2016). The paper sets out the importance of exploiting Augmented Reality and haptic feedback to deliver immersive simulation in a natural way to the human operator who is working with autonomous systems. Such simulations would make it possible to address the human factor, i.e., to create a condition similar to that faced in real operations. The modeling of human factors, such as stress, will therefore represent a future challenge for M&S systems. To sustain the UMVs development, M&S systems must be designed according to the IEEE Distributed Simulation Engineering and Execution Process (SISO, 2010), and the more recent Scenario Development Guideline of the Simulation Interoperability Standard Organization (SISO, 2016). Future M&S will consists of a network of simulators working together with C2 stations in the loop and with special emphasis on implementing autonomous behavior of UMVs and their messages transfer using standard procedures, e.g., the C2-Simulation Interoperability (C2SimI) language (Tolk and Boulet, 2007). In Biagini et al. (2018), the authors describe an M&S federation of simulators communicating with operational C2S. The simulator federation is based on the HLA Run-Time Infrastructure (RTI) working with several technologies such as *ad-hoc* Artificial Intelligence (AI) modules for robotic behavior and C2SimI for interaction between simulators and C2S. The paper illustrates how M&S will help evaluate potential scenarios concerning an Autonomous System of Systems, promoting the development cycle not just for new platforms, but also for performance assessment methodologies and operative procedures.

5. CONCLUSIONS

In this paper, a comprehensive view of interoperability among UMVs is provided. Interoperability is a fundamental feature for the success of UMVs missions and its development is a long-term goal for both civil and military communities. The current plethora of UMVs is characterized by poor interoperability among them and with external systems, including legacy ones, essentially because of the urgent needs in operational theaters—for the military world—and the parallel growth of the UMV market—for the industrial and civil world. However, interoperability remains the key to increase the capacity of an operational system of systems to share information quickly, improving the MSA and therefore the efficiency in using the available resources.

The critical analysis in section 2 of the various studies and projects focusing on the concept of interoperability among UMVs, not neglecting the role of human operators in the loop, demonstrates that interoperability is currently receiving a high level of attention with a large amount and diversity of efforts. This can be explained by the fact that the current missions of the UMVs are finding increasingly blurred lines of operational space while the requirement to standardize and re-utilize sensors, algorithms, information, systems and vehicles, is urgent but still very difficult. From the analysis of state of the art presented, it is noted that currently there are no comprehensive standards on interoperability, but the groundwork effort is underway (e.g., STANAG 4586). The real challenge, following the promulgation of such a standard, will be to convince UMVs and sensors suppliers to fully adopt it due to proprietary interests.

Moving to the field, interoperability tests between the innovative Folaga WAVE AUV, equipped with oceanographic sensors, and the CMRE network C2S onboard NRV Alliance, using a gateway buoy as a ROS-MOOS bridge, are presented in section 3. Their integration has been evaluated and validated through at-sea operational experiments off the La Spezia coast (Italy). This successful interoperability experimentation between different autonomous systems, with their own acoustic modems and middleware, is an important step to improve MSA with respect to underwater assets during a joint NATO exercise. The Folaga WAVE software modularity makes it possible to incorporate a new AUV in the existing CMRE network, which in turn demonstrated its flexibility to integrate newly available assets

thanks to its decentralized architecture. According to the authors' knowledge, this was the first Italian interoperable approach to be thoroughly tested and demonstrated in an operational exercise involving NATO manned and unmanned assets.

A common view arises in this paper from the theoretical and experimental evaluation of missions peculiar to UMVs: they must be fully interoperable in order to enhance their efficiency, reliability, and survivability while lowering the human burden keeping reduced cost. For future unmanned systems, modularity, interoperability and the use of advanced technology must meet more sophisticated operational requirements (section 4), while solutions developed for terrestrial networks could not effectively be extended to marine scenarios. Future challenging scenarios will require UMVs to interoperate with other manned and unmanned components of the *whole system-of-systems* to enhance the capability to gather information, make decisions, and execute actions, thus reducing reaction time. The authors' proposed challenge on interoperability is to find a transparent way to transfer control of a given UMV's payload from one "control station" to another controlling entity (human or robotic) while keeping the rest of the UMV controlled by the original one.

Finally, current operational culture poses a brake on the interoperability among UMVs. Even if out of the field, human operators are still very involved in the non-autonomous unmanned systems missions, i.e., a point-to-point communication and command line is needed and typically established. An Interoperable system-of-systems of autonomous UMVs will request the human being involved at a supervisory level only and on limited time windows. The acceptance of this concept of operations will be achieved with a progressive approach, reducing the human supervision step-by-step after establishing trust in the system's performance (in a very broad sense). The extent of human monitoring will be reduced as reliability increases, supported by stable and mature interoperability, enabling Autonomous UMVs to attain their maximum capacity.

AUTHOR CONTRIBUTIONS

VM wrote the manuscript with support from RC and DT. RC, VM, MS, AC, and AT conceived and developed the concept. DF, MM, and LM provided support in algorithms and C++ code for integration in ROS and MOOS middlewares. AT was the Scientist in Charge of the whole ASW-ODC17 experimentation while RC, DF, VM, MM, and LM designed and ran the specific experiments here presented. MS, AT, and AC gave feedback on the paper and helped to obtain the funding for this research.

FUNDING

This research was partially funded by the National Research Projects of Military interest (PNRM), contract 20332 WAVE of December 2014, and partially funded by the NATO Allied Command Transformation (ACT), Future Solutions (FuSol) Branch, under the Autonomy for ASW Project.

ACKNOWLEDGMENTS

This work acknowledges the use of data that was made possible by the ASW-ODC17 sea trial, including as participants CMRE and CSSN. The authors are indebted to all the CMRE staff that supported them during ASW-ODC17. The authors would like to thank all the SEALab members, the joint laboratory between the CSSN and ISME, who helped the research team during the sea trials.

REFERENCES

Abreu, P., Antonelli, G., Arrichiello, F., Caffaz, A., Caiti, A., Casalino, G., et al. (2016). Widely scalable mobile underwater sonar technology: an overview of the H2020 WiMUST project. *Mar. Technol. Soc. J.* 50, 42–53. doi: 10.4031/MTSJ.50.4.3

Allotta, B., Costanzi, R., Meli, E., Pugi, L., Ridolfi, A., and Vettori, G. (2014). Cooperative localization of a team of AUVs by a tetrahedral configuration. *Robot. Auton. Syst.* 62, 1228–1237. doi: 10.1016/j.robot.2014.03.004

Alves, J., Furfaro, T., LePage, K., Munafò, A., Pelekanakis, K., Petroccia, R., and Zappa, G. (2016). "Moving JANUS forward: a look into the future of underwater communications interoperability," in *OCEANS 2016 MTS/IEEE Monterey* (Monterey, CA), 1–6. doi: 10.1109/OCEANS.2016.7761094

Been, R., Hughes, D. T., Potter, J. R., and Strode, C. (2010). "Cooperative anti-submarine warfare at NURC moving toward a net-centric capability," in *OCEANS'10 IEEE Sydney* (Sydney, NSW), 1–10. doi: 10.1109/OCEANSSYD.2010.5603637

Benjamin, M. R., Schmidt, H., Newman, P. M., and Leonard, J. J. (2010). Nested autonomy for unmanned marine vehicles with MOOS-IvP. *J. Field Robot.* 27, 834–875. doi: 10.1002/rob.20370

Biagini, M., Corona, F., and Casar, J. (2018). "Operational scenario modelling supporting unmanned autonomous systems concept development," in *Modelling and Simulation for Autonomous Systems*, ed J. Mazal (Cham: Springer International Publishing), 253–267. doi: 10.1007/978-3-319-76072-8_18

Birk, A., Antonelli, G., Caiti, A., Casalino, G., Indiveri, G., Pascoal, A., et al. (2011). "The CO3AUVs (Cooperative Cognitive Control for Autonomous Underwater Vehicles) project: overview and current progresses," in *OCEANS 2011 IEEE-Spain* (Spain), 1–10. doi: 10.1109/Oceans-Spain.2011.6003552

Bovio, E., Tyce, R. C., and Schmidt H. (2001). "Autonomous underwater vehicle and ocean modelling networks," in *NATO Undersea Research Centre GOATS 2000 Conference Proceedings*, La Spezia.

Braga, J., Martins, R., Petrioli, C., Petroccia, R., and Picari, L. (2016). "Cooperation and networking in an underwater network composed by heterogeneous assets," in *Proceedings of MTS/IEEE OCEANS 2016* (Monterey, CA), 1–9. doi: 10.1109/OCEANS.2016.7761219

Caiti, A., Calabrò, V., Dini, G., Duca, A., and Munafò, A. (2012). Secure cooperation of autonomous mobile sensors using an underwater acoustic network. *Sensors* 12, 1967–1989. doi: 10.3390/s120201967

Caiti, A., Costanzi, R., Fenucci, D., Manzari, V., Caffaz, A., and Stifani, M. (2018). "WAVE module for hybrid oceanographic autonomous underwater vehicle-prototype experimental validation and characterization," in *International Ship Control Systems Symposium (iSCSS)* (Glasgow). doi: 10.24868/issn.2631-8741.2018.001

Canepa, G., Tesei, A., Troiano, L., Biagini, S., Aglietti, F., Dymond, R., et al. (2017). "Comparison of computation time and accuracy of the real time implementation of two beamforming algorithms," in *4th International Conference and Exhibition on Underwater Acoustics*, (Skiathos).

Carrera, A., Tremori, A., Caamaño, P., Been, R., Crespo Pereira, D., and Bruzzone, A. G. (2016). "HLA interoperability for ROS-based autonomous systems," in *Modelling and Simulation for Autonomous*

Systems, ed J. Hodicky (Cham: Springer International Publishing), 128–138. doi: 10.1007/978-3-319-47605-6_10

Costanzi, R., Fenucci, D., Manzari, V., Micheli, M., Morlando, L., Natale, D., et al. (2018). "At-sea NATO operational experimentation with interoperable underwater assets using different robotic middlewares," in *Technology and Science for the Ships of the Future: Proceedings of NAV 2018: 19th International Conference on Ship & Maritime Research* (Trieste: IOS Press), 194.

Cummings, M. L., How, J. P., Whitten, A., and Toupet, O. (2012). The impact of human-automation collaboration in decentralized multiple unmanned vehicle control. *Proc. IEEE* 100, 660–671. doi: 10.1109/JPROC.2011.2174104

Dahlmann, A., Dickow, M., and Tisserant, L. (2015). *The EDA and the Field of Research and Technology*. London, UK: The European Defence Agency: Arming Europe.

de Sousa, J. B., Pereira, J., Alves, J., Galocha, M., Pereira, B., Lourenço, C., et al. (2015). "Experiments in multi-vehicle operations: the rapid environmental picture Atlantic exercise 2014," in *OCEANS 2015-Genova* (Genova), 1–7. doi: 10.1109/OCEANS-Genova.2015.7271761

DeMarco, K., West, M. E., and Collins, T. R. (2011). "An implementation of ROS on the yellowfin autonomous underwater vehicle (AUV)," in *OCEANS 2011 MTS/IEEE Kona* (Kona, HI: IEEE), 1–7. doi: 10.23919/OCEANS.2011.6107001

Dias, P. S., Fraga, S. L., Gomes, R. M. F., Goncalves, G. M., Pereira, F. L., Pinto, J., et al. (2005). "Neptus - a framework to support multiple vehicle operation," in *Europe Oceans 2005*, (Brest), 963–968. doi: 10.1109/OCEANSE.2005.1513187

Dol, H., Casari, P., and Van Der Zwan, T. (2014). "Software-defined open-architecture modems: Historical review and the NILUS approach," in *2014 Underwater Communications and Networking, UComms 2014* (Sestri Levante: Institute of Electrical and Electronics Engineers Inc.) doi: 10.1109/UComms.2014.7017154

Doniec, M., Detweiler, C., Vasilescu, I., and Rus, D. (2010). "Using optical communication for remote underwater robot operation," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE), 4017–4022. doi: 10.1109/IROS.2010.5650224

EDA (2011). *UMS Methodology and Coordination Roadmap*. European Defence Agency (EDA). Available online at: https://www.eda.europa.eu/docs/documents/ums.pdf?sfvrsn=0 (accessed May 18, 2020).

Farr, N., Bowen, A., Ware, J., Pontbriand, C., and Tivey, M. (2010). "An integrated, underwater optical/acoustic communications system," in *OCEANS'10 IEEE Sydney* (Sydney, NSW), 1–6. doi: 10.1109/OCEANSSYD.2010.5603510

Fenucci, D., Caffaz, A., Costanzi, R., Fontanesi, E., Manzari, V., Sani, L., et al. (2016). "WAVE: a wave energy recovery module for long endurance gliders and AUVs," in *OCEANS 2016 MTS/IEEE Monterey* (Monterey, CA). doi: 10.1109/OCEANS.2016.7761136

Ferri, G., Munafó, A., Tesei, A., Braca, P., Meyer, F., Pelekanakis, K., et al. (2017). Cooperative robotic networks for underwater surveillance: an overview. *IET Rada Sonar Navigat*. 11, 1740–1761. doi: 10.1049/iet-rsn.2017.0074

Frazzetta, S., and Pacino, M. (2013). A STANAG 4586 oriented approach to UAS navigation. *J. Intell. Robot. Syst*. 69, 21–31. doi: 10.1007/s10846-012-9694-5

Freitag, L., Grund, M., Singh, S., Partan, J., Koski, P., and Ball, K. (2005). "The WHOI micro-modem: an acoustic communications and navigation system for multiple platforms," in *Proceedings of OCEANS 2005 MTS/IEEE*, (Washington, DC), 1086–1092. doi: 10.1109/OCEANS.2005.1639901

Grasso, R., Braca, P., Fortunati, S., Gini, F., and Greco, M. S. (2016). Dynamic underwater glider network for environmental field estimation. *IEEE Trans. Aerospace Electron. Syst*. 52, 379–395. doi: 10.1109/TAES.2015.140935

Grimmett, D. J. (2009). "Automatic identification of specular detections in multistatic sonar systems," in *OCEANS 2009, MTS/IEEE Biloxi-Marine Technology for Our Future: Global and Local Challenges* (Biloxi, MS: IEEE), 1–10.

Hastie, H., Garcia, F. J. C., Robb, D. A., Patron, P., and Laskov, A. (2017). "Miriam: a multimodal chat-based interface for autonomous systems," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, (Glasgow) 495–496. doi: 10.1145/3136755.3143022

Henning, M. (2008). The rise and fall of CORBA. *Commun. ACM* 51, 52–57. doi: 10.1145/1378704.1378718

Hodicky, J. (2014). "HLA as an experimental backbone for autonomous system integration into operational field," in *International Workshop on Modelling and Simulation for Autonomous Systems* (Rome: Springer), 121–126. doi: 10.1007/978-3-319-13823-7_11

Hodicky, J. (2016). "Autonomous systems operationalization gaps overcome by modelling and simulation," in *International Workshop on Modelling and Simulation for Autonomous Systems* (Rome: Springer), 40–47. doi: 10.1007/978-3-319-47605-6_4

Hodicky, J. (2018). "Standards to support military autonomous system life cycle," in *Mechatronics 2017*, eds T. Březina, T., and R. Jabłoński (Cham: Springer International Publishing), 671–678. doi: 10.1007/978-3-319-65960-2_83

Huang, H. M. (2004). *Autonomy Levels for Unmanned Systems (ALFUS) Framework Volume I: Terminology Version 2.0*. National Institute of Standards and Technology (NIST).

IEEE (2007). *IEEE Recommended Practice for Verification, Validation, and Accreditation of a Federationan Overlay to the High Level Architecture Federation Development and Execution Process*. IEEE.

Indiveri, G., Antonelli, G., Arrichiello, F., Caffaz, A., Caiti, A., Casalino, G., et al. (2016). Overview and first year progress of the widely scalable mobile underwater sonar technology H2020 project. *IFAC-PapersOnLine* 49, 430–433. doi: 10.1016/j.ifacol.2016.10.442

Kalwa, J. (2010). Final results of the European project GREX: Coordination and control of cooperating marine robots. *IFAC Proc. Vol*. 43, 181–186. doi: 10.3182/20100906-3-IT-2019.00033

Kalwa, J., Tietjen, D., Carreiro-Silva, M., Fontes, J., Brignone, L., Gracias, N., et al. (2016). The European project morph: Distributed UUV systems for multimodal, 3D underwater surveys. *Mar. Technol. Soc. J*. 50, 26–41. doi: 10.4031/MTSJ.50.4.10

LePage, K. D., Goldhahn, R., Alves, J., Strode, C., Braca, P., Ferri, G., et al. (2015). "Autonomous networked anti-submarine warfare research and development at CMRE," in *OCEANS 2015-Genova* (Genova), 1–6. doi: 10.1109/OCEANS-Genova.2015.7271777

Marques, M. M. (2012). *STANAG 4586-Standard Interfaces of UAV Control System (UCS) for NATO UAV Interoperability*. Afeite: NATO Standardization Agency.

Martins, R., Dias, P. S., Marques, E. R., Pinto, J., Sousa, J. B., and Pereira, F. L. (2009). "IMC: a communication protocol for networked vehicles and sensors," in *Oceans 2009-Europe* (Bremen: IEEE), 1–6. doi: 10.1109/OCEANSE.2009.5278245

Masmitja, I., Gomariz, S., Del Rio, J., Bouvet, P., and Aguzzi, J. (2018). "Underwater multi-target tracking with particle filters," in *2018 OCEANS - MTS/IEEE Kobe Techno-Oceans, OCEANS-Kobe 2018* (Kobe: Institute of Electrical and Electronics Engineers Inc.). doi: 10.1109/OCEANSKOBE.2018.8558865

Miguelañez, E., Patrón, P., Brown, K. E., Petillot, Y. R., and Lane, D. M. (2011). Semantic knowledge-based framework to improve the situation awareness of autonomous underwater vehicles. *IEEE Trans. Knowledge Data Eng*. 23, 759–773. doi: 10.1109/TKDE.2010.46

Möller, B., Morse, K. L., Lightner, M., Little, R., and Lutz, R. (2008). "HLA evolved-a summary of major technical improvements," in *Proceedings of 2008 Spring Simulation Interoperability Workshop, 08F-SIW-064* (Edinburgh).

Munafó, A., and Ferri, G. (2017). An acoustic network navigation system. *J. Field Robot*. 34, 1332–1351. doi: 10.1002/rob.21714

NSO (2017). *Digital Underwater Signalling Standard for Network Node Discovery & Interoperability*. NATO Standardization Office.

Petroccia, R., Alves, J., and Zappa, G. (2016). "Fostering the use of JANUS in operationally-relevant underwater applications," in *2016 IEEE Third Underwater Communications and Networking Conference (UComms)*, (Lerici), 1–5. doi: 10.1109/UComms.2016.7583424

Petroccia, R., Alves, J., and Zappa, G. (2017). JANUS-based services for operationally relevant underwater applications. *IEEE J. Ocean. Eng*. 42, 994–1006. doi: 10.1109/JOE.2017.2722018

Petroccia, R., Sliwka, J., Grati, A., Grandi, V., Guerrini, P., Munafó, A., et al. (2018). "Deployment of a persistent underwater acoustic sensor network: the CommsNet17 experience," in *Proceedings of MTS/IEEE OCEANS 2018* (Kobe). doi: 10.1109/OCEANSKOBE.2018.8559262

Petroccia, R., Zappa, G., Furfaro, T., Alves, J., and D'Amaro, L. (2018). "Development of a software-defined and cognitive communications architecture at CMRE," in *OCEANS 2018 MTS/IEEE Charleston*, 1–10. doi: 10.1109/OCEANS.2018.8604849

Pinto, J., Dias, P. S., Martins, R., Fortuna, J., Marques, E., and Sousa, J. (2013). "The LSTS toolchain for networked vehicle systems," in *2013 MTS/IEEE OCEANS-Bergen* (Bergen: IEEE), 1–9. doi: 10.1109/OCEANS-Bergen.2013.6608148

Platts, J., Cummings, M., and Kerr, R. (2007). "Applicability of STANAG 4586 to future unmanned aerial vehicles," in *AIAA Infotech@ Aerospace 2007 Conference and Exhibit*, (Rohnert Park, CA), 2753. doi: 10.2514/6.2007-2753

Potter, J., Alves, J., Furfaro, T., Vermeij, A., Jourden, N., Merani, D., et al. (2014). "Software defined open architecture modem development at CMRE," in *2014 Underwater Communications and Networking (UComms)* (Sestri Levante: IEEE), 1–4. doi: 10.1109/UComms.2014.7017157

Potter, J., Alves, J., Green, D., Zappa, G., Nissen, I., and McCoy, K. (2014). "The JANUS underwater communications standard," in *2014 Underwater Communications and Networking (UComms)* (Sestri Levante: IEEE), 1–4. doi: 10.1109/UComms.2014.7017134

Ren, W., Beard, R. W., and Atkins, E. M. (2005). "A survey of consensus problems in multi-agent coordination," in *Proceedings of the 2005, American Control Conference, 2005*, (Portland, OR), 1859–1864. doi: 10.1109/ACC.2005.1470239

Ridolfi, A., Spaccini, D., Fanelli, F., Franchi, M., Monni, N., Picari, L., et al. (2018). An autonomous underwater vehicle and sunset to bridge underwater networks composed of multi-vendor modems. *Annu. Rev. Control* 46, 295–303. doi: 10.1016/j.arcontrol.2018.10.005

Robb, D. A., Willners, J. S., Valeyrie, N., Garcia, F. J. C., Laskov, A., Liu, X., et al. (2018). "A natural language interface with relayed acoustic communications for improved command and control of AUVs," in *2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV)*, (Porto), 1–6. doi: 10.1109/AUV.2018.8729778

Ruiz, J., Behner, H., Herzog, R., Hodicky, J., Lofstrand, B., and Vrieler, S. (2016). "Towards a new NATO certification capability for HLA interoperability," in *Simulation Innovation Workshop (SIW) 2016*, (Orlando, FL).

Schneider, T., and Schmidt, H. (2010). Unified command and control for heterogeneous marine sensing networks. *J. Field Robot.* 27, 876–889. doi: 10.1002/rob.20346

SISO (2010). *Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP)*. IEEE Standard 1730-2010, Simulation Interoperability Standards Organization.

SISO (2016). *Guideline on Scenario Development for Simulation Environments*. Technical report, Simulation Interoperability Standards Organization.

Stojanovic, M. (2006). "On the relationship between capacity and distance in an underwater acoustic communication channel," in *Proceedings of the 1st ACM International Workshop on Underwater Networks, WUWNet '06* (Los Angeles, CA: ACM), 41–47. doi: 10.1145/1161039.1161049

Stokey, R. P., Freitag, L. E., and Grund, M. D. (2005). "A compact control language for auv acoustic communication," in *Europe Oceans 2005*, (Brest), 1133–1137. doi: 10.1109/OCEANSE.2005.1513217

Strode, C., Mourre, B., and Rixen, M. (2012). Decision support using the multistatic tactical planning aid (MSTPA). *Ocean Dyn.* 62, 161–175. doi: 10.1007/s10236-011-0483-7

Terracciano, D. S., Manzari, V., Stifani, M., Allotta, B., Caiti, A., and Casalino, G. (2019). "Sealab current research trends: maritime unmanned systems for dual-use applications," in *IMEKO International Workshop on Metrology for the Sea*, (Genoa).

Tolk, A., and Boulet, J. (2007). "Lessons learned on NATO experiments on C2/M&S interoperability," in *IEEE Spring Simulation Interoperability Workshop* (Norfolk, VA: IEEE CS Press).

Tremori, A., Sobrino, P. C., Viñas, A. C., Maglione, G. L., Solarna, D., Been, R., et al. (2018). "A verification, validation and accreditation process for autonomous interoperable systems," in *Modelling and Simulation for Autonomous Systems*, ed J. Mazal (Cham: Springer International Publishing), 314–323. doi: 10.1007/978-3-319-76072-8_22

Valavanis, K. P., and Vachtsevanos, G. J. (2015). "UAV autonomy: introduction," in *Handbook of Unmanned Aerial Vehicles*, eds K. Valavanis and G. Vachtsevanos (Dordrecht: Springer), 1723–1724. doi: 10.1007/978-90-481-9707-1_144

Vasilijević, A., Nađ, Đ., Mandić, F., Mišković, N., and Vukić, Z. (2017). Coordinated navigation of surface and underwater marine robotic vehicles for ocean sampling and environmental monitoring. *IEEE/ASME Trans. Mechatron.* 22, 1174–1184. doi: 10.1109/TMECH.2017.2684423

Vermeij, A., Furfaro, T. C., and Alves, J. (2015). "NEMO: an architecture for software communications research in the maritime domain," in *OCEANS 2015-Genova* (Genoa: IEEE), 1–4. doi: 10.1109/OCEANS-Genova.2015.7271694

Whitsitt, S., and Sprinkle, J. (2011). "Message modeling for the joint architecture for unmanned systems (JAUS)," in *2011 18th IEEE International Conference and Workshops on Engineering of Computer-Based Systems*, (Las Vegas, NV), 251–259. doi: 10.1109/ECBS.2011.17

Willcox, S., Manley, J., and Wiggins, S. (2009). The Wave Glider, an energy harvesting autonomous surface vessel. *Sea Technol.* 49, 29–31. Available online at: http://cetus.ucsd.edu/Publications/ReportsOLD/ManleySeaTechnology2009-WG+HARP.pdf

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Robotics and AI

ORIGINAL RESEARCH
published: 09 December 2020
doi: 10.3389/frobt.2020.519985

Control Protocols for Range-Based Navigation of a Networked Group of Underwater Vehicles

Daniela De Palma[1]*, *Giovanni Indiveri*[2] *and Gianfranco Parlangeli*[1]

[1] Department of Innovation Engineering (DII), University of Salento (Interuniversity Center of Integrated Systems for the Marine Environment node), Lecce, Italy, [2] Department of Informatics, Bioengineering, Robotics, and Systems Engineering (DIBRIS), University of Genova (Interuniversity Center of Integrated Systems for the Marine Environment node), Genova, Italy

This paper tackles the problem of formation reconstruction for a team of vehicles based on the knowledge of the range between agents of a subset of the participants. One main peculiarity of the proposed approach is that the relative velocity between agents, which is a fundamental data to solve the problem, is not assumed to be known in advance neither directly communicated. For the purpose of estimating this quantity, a collaborative control protocol is designed in order to mount the velocity data in the motion of each vehicle as a parameter through a dedicated control protocol, so that it can be inferred from the motion of the neighbor agents. Moreover, some suitable geometrical constraints related to the agents' relative positions are built and explicitly taken into account in the estimation framework providing a more accurate estimate. The issue of the presence of delays in the transmitted signals is also studied and two possible solutions are provided explaining how it is possible to get a reasonable range data exchange to get the solution both in a centralized fashion and in a decentralized one. Numerical examples are presented corroborating the validity of the proposed approach.

Keywords: autonomous underwater vehicles, multi-agent system, relative localization, active estimation, range-based navigation

OPEN ACCESS

Edited by:
Oussama Khatib,
Stanford University, United States

Reviewed by:
Elias Kosmatopoulos,
Democritus University of Thrace, Greece
Gianluca Antonelli,
University of Cassino, Italy

***Correspondence:**
Daniela De Palma
daniela.depalma@unisalento.it

Specialty section:
This article was submitted to Robotic Control Systems, a section of the journal Frontiers in Robotics and AI

Received: *13 December 2019*
Accepted: *09 September 2020*
Published: *09 December 2020*

Citation:
De Palma D, Indiveri G and Parlangeli G (2020) Control Protocols for Range-Based Navigation of a Networked Group of Underwater Vehicles. Front. Robot. AI 7:519985. doi: 10.3389/frobt.2020.519985

1. INTRODUCTION

Localization is one of the most important basic abilities for an autonomous vehicle to perform autonomously a wide number of tasks (Ferri et al., 2017; Simetti et al., 2017; Antonelli et al., 2018), so that an accurate and reliable localization algorithm is a key practical tool for the success of mission in many applications of underwater robotics.

In essence, the localization problem is often addressed exploiting geometrical relations between the pose of the vehicle and the sensors, so that the issue of solving the localization problem may be strongly related to the environment of the given application. Sensor technology strongly depends on the environment, e.g., the Global Navigation Satellite System (GNSS), Attitude Heading Reference Systems (AHRS), radar-based tracking systems, accelerometers, gyros, and compass devices. This makes the issue of the underwater localization problem more challenging, and it has been considerably studied in the past years. Underwater acoustic-based trilateration solutions as long base line (LBL) systems have been studied as well, but they require complex deployment operations (Scherbatyuk, 1995).

Localization is a long-time debated research area in robotics and beyond, and different aspects have been studied over time. In this paper, we consider the relative localization problem for a team of agents, that is, the formation reconstruction problem in a multi-vehicle framework. This is a

peculiar problem in the research area of localization, which has been recently considered by several authors for its importance in various applications. In the paper by Soares et al. (2013), the authors propose a formation keeping under severe communication and localization constraints, which is a typical condition of the underwater environment. In Sarras et al. (2017), the authors adopt an observer-based approach to treat the problem of multi-vehicle collaborative localization using time-varying range and relative velocity measurements, while in Halsted and Schwager (2017) a method of estimating the shape of an indoor environment using the echos of acoustic pulses among the robots is studied. Indeed, when an underwater mission is performed by a team of robots, it is often fundamental to know the relative positions and orientations in order to be able to correctly merge the data of the environment (e.g., merging pieces of map) collected by each individual robot (see, e.g., Roumeliotis and Bekey, 2000).

From a theoretic standpoint, the range-based estimation problems have been recently considered also in a single vehicle framework (Bayat et al., 2016). The challenges of dealing with single range measurements come from the fact that they are a non-linear algebraic map of the vehicles' positions hence the observability analysis requires the tools of local and weakly local observability (Hermann and Krener, 1977), but this approach suffers from several difficulties (Gadre and Stilwell, 2004; Ross and Jouffroy, 2005; Jouffroy and Reger, 2006). However, an alternative approach has been recently investigated recurring to a reformulation of the problem, which requires the observability of a linear time varying system (see, e.g., Batista et al., 2011; De Palma et al., 2017) so that a number of useless drawbacks of the local approach are avoided.

In this paper, we afford the relative positions reconstruction problem for a team of collaborative robots using local data. Collaborative navigation based on single-range data has been studied in the underwater environment (Fallon et al., 2010; Soares et al., 2013; Webster et al., 2013) as well as in more general settings (Cao et al., 2011). Indeed, since the milestone paper by Sanderson (1997), the area of cooperative navigation and localization has been significantly explored. One first significant attempt to the decentralized collective localization problem is explained in Roumeliotis and Bekey (2002); to achieve this goal, data processed during each collective localization session are propagated among all the robots in the group. This approach is further investigated in Mourikis and Roumeliotis (2006), where the Relative Position Measurement Graph (RPMG), i.e., the weighted directed graph representing the network of robot-to-robot exteroceptive measurements, is introduced and used as a key tool for the analysis of cooperative localization. The distributed acoustic navigation problem for Autonomous Underwater Vehicles (AUVs) is explored in Bahr et al. (2009), where the authors use acoustic ranging and data exchange based on dead-reckoning and range-only measurements provided by acoustic modems that are mounted on each vehicle to achieve cooperative positioning. In the paper by Allotta et al. (2014), the use of AUVs with low-cost instrumentation is explored (namely, each of them is equipped with a low-cost IMU, a compass and depth sensor, but only one of them, the master, has a high accuracy navigation sensor such as the DVL), and acoustic modems for communication are used as sensors of relative distance to achieve an innovative cooperative localization algorithm. In Soares et al. (2017), the authors optimize the non-convex maximum-likelihood estimator in the presence of range measurements contaminated with Gaussian noise, and obtain a convergent, accurate, and distributed positioning algorithm that outperforms the extended Kalman filter. However, this topic has been largely explored by several authors, and the interested reader may refer to Arai et al. (2002) (section V).

The research activity reported in this paper stems from the above considerations and is strongly inspired by the experience gained within a European project (Antonelli et al., 2018). The goal is to extend the preliminary results achieved by the same authors in De Palma et al. (2015) and De Palma et al. (2019) as follows. One of the main novelties with respect to De Palma et al. (2015) is relative to the following fact. Based on the consideration that communications networks in the underwater environment do not perform well, we want to avoid the direct communication between vehicles of the relative velocity by setting a suitable agreed control protocol in which it is possible to encapsulate the data which one vehicle wishes to communicate as a parameter that can be easily estimated using the relative motion by any of its neighbors. As opposed to the approach in Mourikis and Roumeliotis (2006), in this paper the solution proposed relies on intra-vehicle ranges only rather than relative positions of vehicles. As a further peculiar feature of the approach proposed by the authors of this paper, we further use topology-based relations among the unknown variables as an additional constraint and this results in reduction of the overall estimate uncertainty. Further, in this paper we explicitly account for delays in range measurements acquisition. Indeed, the technology underneath underwater networks is typically acoustic and communication delays may be significant and their impact may not be neglected. The solution provided in this paper exploits the intuitive idea of a neat time-division protocol to prevent any delay-related issue in the localization solution provided that an upper bound is available.

The paper is structured as follows: after a brief summary of notation and terminology in section 2, we provide the general problem statement in section 3, where section 4 is dedicated to the localization-oriented control protocol. In section 5, the observer design is performed, and section 5.1 is fully dedicated to the projection approach, which allows to improve the estimate precision. In section 6, the issue of delays in measurements is faced, and in section 7, two possible communication protocols are provided both in the case of a single "leader" agent performing the elaboration (thus only one agent collecting all the estimates) or any agent of the network. In section 8, a wide simulation activity is reported and discussed, showing the effectiveness of the proposed approach. Section 9 closes the paper summarizing the results achieved in the paper.

2. NOTATION AND GRAPH THEORY TERMINOLOGY

In the following, we introduce the notation adopted in the paper and some tools from graph theory (Godsil and Royle, 2001),

which are useful for a mathematical treatment of the problem. We use the symbol $\otimes$ to denote the Kronecker product between two matrices, which is defined as follows. For a pair of matrices $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ and $\boldsymbol{B} \in \mathbb{R}^{p\times q}$:

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & \cdots & a_{1m}\boldsymbol{B} \\ \vdots & \ddots & \vdots \\ a_{n1}\boldsymbol{B} & \cdots & a_{nm}\boldsymbol{B} \end{bmatrix}. \tag{1}$$

We use $\mathrm{diag}(\boldsymbol{A}_1, \ldots, \boldsymbol{A}_n)$ to denote a block-diagonal matrix with matrix diagonal entries $\boldsymbol{A}_i$. A *graph* $\mathcal{G}$ is the collection of a set $\mathcal{V} = \{1, \ldots, n\}$ called *set of nodes* and another set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, which is called *the set of edges*. For a given $i \in \mathcal{V}$, the set $\mathcal{N}_i = \{j \in \mathcal{V} : (j, i) \in \mathcal{E}\}$ is called the set of its *neighbors*. A path $\mathcal{P}$ between node i and node j is a collection of nodes and edges of $\mathcal{G}$ connecting i and j; a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is connected if there exists a path connecting each $h, g \in \mathcal{V}$. A cycle is analogously defined with the additional condition that $i = j$. A cycle $\bar{\mathcal{C}}$ is linearly independent from a preassigned set of cycles if at least one edge in $\bar{\mathcal{C}}$ is not present in the union of the edge sets of the cycles.

3. PROBLEM FORMULATION

Let $\mathbf{x}_i \in \mathbb{R}^3$ for $i = 1, 2, \ldots, n$ denote the position of n agents and $\mathbf{v}_i \in \mathbb{R}^3$ their velocity. Each agent is able to know its own velocity with reference to the common frame $\mathcal{I}$ so that, unless specified, we assume that the velocity is expressed in this common frame $\mathcal{I}$. We assume that if two agents are able to measure the range between themselves, they are connected, so that it is possible to define a connection graph. Inspired by the work of Mourikis and Roumeliotis (2006), we refer to it as to an RPMG, which we assume to be a simple graph $\mathcal{G}$ with node set $\mathcal{V} = \{1, \ldots, n\}$ and the edges set $\mathcal{E}$. We further assume that if two agents are able to measure the range between themselves, they can establish a communication link to exchange data, so that it is possible to consider $\mathcal{G}$ also a communication graph.

The evolution of the agents can be computed using simple kinematic equations:

$$\dot{\mathbf{x}}_i(t) = \mathbf{v}_i(t) \quad : \quad i \in \mathcal{V} \tag{2}$$

$$\mathbf{z}_{ij}(t) := \mathbf{x}_i(t) - \mathbf{x}_j(t) \quad : \quad (i, j) \in \mathcal{E} \tag{3}$$

$$\mathbf{v}_{ij}(t) := \mathbf{v}_i(t) - \mathbf{v}_j(t) \quad : \quad (i, j) \in \mathcal{E} \tag{4}$$

so that

$$\dot{\mathbf{z}}_{ij}(t) = \mathbf{v}_{ij}(t) \quad : \quad (i, j) \in \mathcal{E} \tag{5}$$

$$y_{ij}(t) = \|\mathbf{z}_{ij}(t)\|^2 \quad : \quad (i, j) \in \mathcal{E}, \tag{6}$$

where $\mathbf{z}_{ij}$ in Equation (3) denotes the relative positions among those agents able to exchange information. All agents are assumed to be able to acquire measurements of their relative Euclidean distance y_{ij} in Equation (6), with the goal of estimating $\mathbf{z}_{ij}$ performing an elaboration of the relative range measurements y_{ij} and local data. A fundamental difference between this problem statement and the one afforded in the paper by De Palma et al. (2015) is that here we do not assume to exchange the velocity data through a dedicated underwater network, but we encapsulate this information as parameters of an agreed control protocol and infer the velocity value using a range-based Kalman filter as detailed in the following. We consider this choice of the problem statement a significant step forward for all those applications where only range information exchange is possible.

From now on we work under the assumption that the communication graph is time invariant. This choice is instrumental to keep the problem simple and the associated solution clear. The authors believe that it is a mild assumption considering that the resulting localization procedure requires a bounded amount of time. It is equivalent to assume that nodes that are neighbors at the initial time $\bar{t}$ keep this communication alive during the whole time span, while other nodes that may fall in the communication range after $\bar{t}$ are not included in the elaboration. Furthermore, from a practical point of view, it should be emphasized that acoustic underwater communications degrades drastically after certain threshold distances. If a group of underwater vehicles keeps its formation during a mission within such a threshold distance (most common case), the quality of the communications can be assumed to remain good and the communication links can be considered constant. Finally, even if a communication link (i,j) was lost, the proposed strategy could still be adopted by deleting the corresponding state variable $\mathbf{z}_{ij}$.

We now describe a strategy to improve the estimation when cycles are present in the communication graph. Indeed, the relative positions may be not independent, but they can be subject to geometric constraints if they belong to the same cycle. Based on the consideration that the sum of all the vectors representing the relative positions of agents belonging to a cycle must necessarily be zero, each set of independent cycles corresponds to a set of independent geometric constraints on the relative positions. Considering a connected graph with n nodes and m edges, any cycle basis can be mapped into a set of $m-n+1$ additional relations. A team of $n = 4$ agents with $m = 5$ links is depicted in **Figure 1**. It is possible to set $m-n+1 = 2$ additional relations, namely

$$\mathbf{z}_{12} + \mathbf{z}_{23} + \mathbf{z}_{31} = \mathbf{0}_{3\times 1} \tag{7}$$

$$-\mathbf{z}_{23} + \mathbf{z}_{24} + \mathbf{z}_{43} = \mathbf{0}_{3\times 1} \tag{8}$$

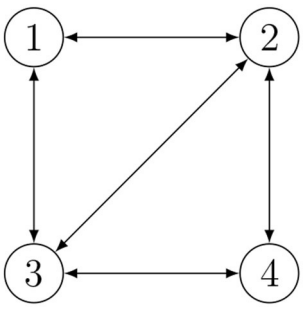


FIGURE 1 | Example of Relative Position Measurement Graph (RPMG) with 4 agents and 5 links.

that can be rewritten as $\boldsymbol{D}\mathbf{z}^* = \mathbf{0}_{3\times 1}$ with

$$\boldsymbol{D} = \begin{bmatrix} \boldsymbol{I}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & \boldsymbol{I}_{3\times 3} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \boldsymbol{I}_{3\times 3} & \boldsymbol{I}_{3\times 3} & \mathbf{0}_{3\times 3} & -\boldsymbol{I}_{3\times 3} \end{bmatrix} \in \mathbb{R}^{6\times 15} \quad (9)$$

$$\mathbf{z}^* = (\mathbf{z}_{12}^\top \ \mathbf{z}_{24}^\top \ \mathbf{z}_{43}^\top \ \mathbf{z}_{31}^\top \ \mathbf{z}_{23}^\top)^\top \in \mathbb{R}^{15} \quad (10)$$

The above example can be easily generalized, and it is possible to write a general setting as follows. For a team of n nodes and RPMG edge set $\mathcal{E}$, the additional relations can be expressed as:

$$\boldsymbol{D}\,\mathbf{z}^* = \mathbf{0}_{3(m-n+1)\times 1} \quad (11)$$

being

$$\mathbf{z}^* = (\cdots \ \mathbf{z}_{ij}^\top \ \cdots)^\top \in \mathbb{R}^{3m} \text{ with } (i,j) \in \mathcal{E}, \quad (12)$$

$$\boldsymbol{D} = \boldsymbol{A} \otimes \boldsymbol{I}_{3\times 3}, \quad (13)$$

$$\boldsymbol{A} = (a_{lk}) \quad (l = 1,\ldots,m-n+1;\ k = 1,\ldots,m) \quad (14)$$

where $\boldsymbol{D} \in \mathbb{R}^{3(m-n+1)\times 3m}$, $\boldsymbol{A} \in \mathbb{R}^{(m-n+1)\times m}$ and $\boldsymbol{A}$ is a signed structured $(0,1)$ matrix, namely $a_{lk} \in \{-1,0,+1\}$. Each geometrical relation associated with (11) can be encapsulated in the state estimation procedure in order to improve the estimation quality.

4. LOCALIZATION-ORIENTED CONTROL LAW

In this section, a motion control scheme for range-based relative localization is proposed. Using this strategy, it is possible to infer the relative velocity of agents from the motion measurements. Consider the following control law for each vehicle:

$$\mathbf{v}_i(t) = \sum_{j\in\mathcal{N}_i} K\,(\mathbf{x}_i(t) - \mathbf{x}_j(t)) \ : \ K > 0, \ \ i \in \mathcal{V} \quad (15)$$

where $K \in \mathbb{R}$ is a positive constant. According to such schema, the motion of each vehicle i depends only on the relative positions with its neighbors, namely $\mathbf{z}_{ij}$ with $j \in \mathcal{N}_i$.

It is worth noting that in our framework Equation (15) cannot be directly implemented (as the actual relative positions $\mathbf{z}_{ij}$ are not known to vehicle i), but we replace the estimated relative positions $\hat{\mathbf{z}}_{ij}$ instead:

$$\mathbf{v}_i(t) = K \sum_{j\in\mathcal{N}_i} \hat{\mathbf{z}}_{ij}(t), \quad i \in \mathcal{V}, \quad (16)$$

leading to the following relative velocities:

$$\mathbf{v}_{ij}(t) = K\left(\sum_{h\in\mathcal{N}_i} \hat{\mathbf{z}}_{ih}(t) - \sum_{\rho\in\mathcal{N}_j} \hat{\mathbf{z}}_{j\rho}(t)\right) \ : \ (i,j) \in \mathcal{E} \quad (17)$$

Details about the specific computation of the estimated relative positions $\hat{\mathbf{z}}_{ij}$ to be used in (16) are provided in the next section.

When agents adopt this control law, the system (5–6) becomes:

$$\dot{\mathbf{z}}_{ij}(t) = K\left(\sum_{h\in\mathcal{N}_i} \hat{\mathbf{z}}_{ih}(t) - \sum_{\rho\in\mathcal{N}_j} \hat{\mathbf{z}}_{j\rho}(t)\right), (i,j) \in \mathcal{E} \quad (18)$$

$$y_{ij}(t) = \|\mathbf{z}_{ij}(t)\|^2. \quad (19)$$

As a final remark, it is interesting to note that the control law in Equation (15) has the same structure of a consensus protocol as those described in Olfati-Saber and Murray (2004) and Ren and Beard (2007). Control strategies based on such kind of protocols have been widely studied for the coordination or formation control of a team of agents (Leonard et al., 2007; Ren and Cao, 2010). In this paper, we do not seek control objectives but we rather use Equation (16) as a localization-oriented control protocol that each vehicle must follow at each $t = kT_s$, being T_s a fixed time interval. Therefore, within each interval the agents keep their velocity constant. Indeed, in this paper such control law is adopted to make the motion informative of each agent's position and velocity, and hence make the range-based relative localization of a networked group of underwater vehicles solvable in finite time so that it can be executed as a routine inside a mission when localization is needed. This is useful when, during a cooperative mission, the relative localization accuracy of the agents decreases; in this case, the proposed localization-oriented control law can be activated so as to improve the accuracy of the relative localization.

The advantage of such solution with respect to the work presented in De Palma et al. (2015) is that by adopting the motion control scheme for range-based relative localization in (16), there is no need for the agents to share their velocity information in the communication channel. Indeed, an agent is able to derive the velocity of the other agents from the knowledge of the adopted control law and the estimated relative positions. This result in a significant reduction of the communication load over the network. The results achieved in this paper are particularly relevant in underwater applications where the bandwidth is often limited due to the acoustic communications.

5. OBSERVER DESIGN

The estimation of the relative positions $\mathbf{z}_{ij}(t)$ in Equations (18), (19) is tackled resorting to the methods presented in Indiveri et al. (2016). Let us integrate Equation (18)

$$\begin{aligned} \mathbf{z}_{ij}(t) &= \mathbf{z}_{ij}(t_0) + \int_{t_0}^{t} K\left(\sum_{h\in\mathcal{N}_i} \hat{\mathbf{z}}_{ih}(\tau) - \sum_{\rho\in\mathcal{N}_j} \hat{\mathbf{z}}_{j\rho}(\tau)\right) d\tau \\ &= \mathbf{z}_{ij}(t_0) + \mathbf{d}_{ij}(t), \end{aligned} \quad (20)$$

with $\mathbf{d}_{ij}(t)$ defined as

$$\mathbf{d}_{ij}(t) := \int_{t_0}^{t} K\left(\sum_{h\in\mathcal{N}_i} \hat{\mathbf{z}}_{ih}(\tau) - \sum_{\rho\in\mathcal{N}_j} \hat{\mathbf{z}}_{j\rho}(\tau)\right) d\tau \in \mathbb{R}^{3\times 1}. \quad (21)$$

Equation (20) allows to compute

$$\mathbf{z}_{ij}^{\top}(t_0)\,\mathbf{z}_{ij}(t_0) = y_{ij}(t_0) = \\ = y_{ij}(t) + \|\mathbf{d}_{ij}(t)\|^2 - 2\mathbf{d}_{ij}^{\top}(t)\mathbf{z}_{ij}(t) \tag{22}$$

yielding

$$\bar{y}_{ij}(t) := \frac{1}{2}[y_{ij}(t) - y_{ij}(t_0) + \|\mathbf{d}_{ij}(t)\|^2] \tag{23}$$

$$\bar{y}_{ij}(t) = \mathbf{d}_{ij}^{\top}(t)\mathbf{z}_{ij}(t). \tag{24}$$

The term $\bar{y}_{ij}(t)$ defined in Equation (23) as well as the term $\mathbf{d}_{ij}(t)$ defined in (21) are both known quantities, so that the new linear output equation in Equation (24) can be considered. Consequently, the original non-linear model (Equations 5 and 6) can be expressed as a Linear Time-Varying (LTV) model

$$\dot{\mathbf{z}}_{ij}(t) = K\left(\sum_{h\in\mathcal{N}_i}\hat{\mathbf{z}}_{ih}(t) - \sum_{\rho\in\mathcal{N}_j}\hat{\mathbf{z}}_{j\rho}(t)\right) \tag{25}$$

$$\bar{y}_{ij}(t) = \mathbf{d}_{ij}^{\top}(t)\mathbf{z}_{ij}(t). \tag{26}$$

Thus, the estimation of $\mathbf{z}_{ij}(t)$ in (25, 26) can be addressed exploiting the standard linear system theory. It should be noticed that the output matrix $\mathbf{d}_{ij}^{\top}(t)$ of the LTV model depends on the control input, hence the observability depends on the agents' relative velocity $\mathbf{v}_{ij}$. It can be proven that a sufficient condition for the observability of the original system (5–6) on $[t_0, t]$ is the invertibility of the observability Gramian of the LTV system (25, 26) defined as:

$$\boldsymbol{G}(t_0, t) = \int_{t_0}^{t}\mathbf{d}_{ij}(\tau)\,\mathbf{d}_{ij}^{\top}(\tau)\,d\tau. \tag{27}$$

The reader should refer to Indiveri et al. (2016) for a detailed analysis of the observability properties of such a system. Let us consider the discrete time formulation of the LTV system given by:

$$\mathbf{z}_{ij}(k+1) = \mathbf{z}_{ij}(k) + K\left(\sum_{h\in\mathcal{N}_i}\hat{\mathbf{z}}_{ih}(k) - \sum_{\rho\in\mathcal{N}_j}\hat{\mathbf{z}}_{j\rho}(k)\right)T_s + \boldsymbol{\omega}(k) \tag{28}$$

$$\bar{y}_{ij}(k) = \mathbf{d}_{ij}^{\top}(k-1)\mathbf{z}_{ij}(k) + \epsilon(k) \tag{29}$$

with

$$\bar{y}_{ij}(k) = \frac{1}{2}\left[y_{ij}(k) - y_{ij}(0) + \|\mathbf{d}_{ij}(k-1)\|^2\right], \tag{30}$$

$$\mathbf{d}_{ij}(k-1) = \sum_{l=0}^{k-1}K\left(\sum_{h\in\mathcal{N}_i}\hat{\mathbf{z}}_{ih}(l) - \sum_{\rho\in\mathcal{N}_j}\hat{\mathbf{z}}_{j\rho}(l)\right)T_s, \tag{31}$$

where $\boldsymbol{\omega}(k)$ and $\epsilon(k)$ are assumed to be zero mean, Gaussian, white, and uncorrelated process and measurements noises with covariances $\boldsymbol{Q}(k)$ and $R(k)$, respectively, and T_s represents the sampling time. A standard Kalman filter can be applied to the model in Equations (28) and (29), leading to the following equations:

$$\hat{\mathbf{z}}_{ij}(k+1|k) = \hat{\mathbf{z}}_{ij}(k|k) + K\left(\sum_{h\in\mathcal{N}_i}\hat{\mathbf{z}}_{ih}(k) - \sum_{\rho\in\mathcal{N}_j}\hat{\mathbf{z}}_{j\rho}(k)\right)T_s \tag{32}$$

$$\boldsymbol{P}_{ij}(k+1|k) = \boldsymbol{P}_{ij}(k|k) + \boldsymbol{Q}(k) \tag{33}$$

$$\boldsymbol{K} = (\boldsymbol{P}_{ij}^{-1}(k+1|k) \\ +\mathbf{d}_{ij}(k)R(k+1)^{-1}\mathbf{d}_{ij}^{\top}(k))^{-1}\mathbf{d}_{ij}(k)R(k+1)^{-1} \tag{34}$$

$$\hat{\mathbf{z}}_{ij}(k+1|k+1) = \hat{\mathbf{z}}_{ij}(k+1|k) + \boldsymbol{K}(\bar{y}(k+1) \\ -\mathbf{d}_{ij}^{\top}(k)\mathbf{z}_{ij}(k+1|k)) \tag{35}$$

$$\boldsymbol{P}_{ij}(k+1|k+1) = (\boldsymbol{P}_{ij}^{-1}(k+1|k) \\ +\mathbf{d}_{ij}(k)R(k+1)^{-1}\mathbf{d}_{ij}^{\top}(k))^{-1}. \tag{36}$$

In the considered scenario, thanks to the intra-vehicles acoustic communications, each agent is able to know the estimations $\hat{\mathbf{z}}_{ij}$ and their covariances $\boldsymbol{P}_{ij}$. Therefore, each agent can improve the estimation accuracy exploiting the additional geometric constraints (11). To this aim, we can benefit from the projection approach described in the following subsection.

5.1. Constraint Exploitation for the Estimate Improvement

Assuming to know the m Kalman filter estimates $\hat{\mathbf{z}}_{ij}$, it is possible to incorporate the constraint (11) in the estimation framework resorting to the approach described in Simon (2006). Let us define the Kalman filter estimate $\hat{\mathbf{z}}^*$ as

$$\hat{\mathbf{z}}^*(k) = (\cdots\ \hat{\mathbf{z}}_{ij}^{\top}(k)\ \cdots)^{\top} \in \mathbb{R}^{3m} : (i,j) \in \mathcal{E}, \tag{37}$$

and its posterior covariance as

$$\boldsymbol{P}^*(k) = \mathrm{diag}(\cdots\ \boldsymbol{P}_{ij}(k)\ \cdots) \in \mathbb{R}^{3m\times 3m} : (i,j) \in \mathcal{E}. \tag{38}$$

An estimate $\hat{\mathbf{z}}_p^*$ satisfying the constraint (11) can be derived projecting the Kalman filter estimate onto the constraint surface; this would lead to the following solution:

$$\hat{\mathbf{z}}_p^*(k) = \boldsymbol{U}\,\hat{\mathbf{z}}^*(k) \tag{39}$$

where $\boldsymbol{U}$ is the projection operator

$$\boldsymbol{U} := \boldsymbol{I}_{3m\times 3m} - \left[\boldsymbol{W}^{-1}\boldsymbol{D}^{\top}\left(\boldsymbol{D}\boldsymbol{W}^{-1}\boldsymbol{D}^{\top}\right)^{-1}\right]\boldsymbol{D} \tag{40}$$

such that $\boldsymbol{U}^2 = \boldsymbol{U}$, $\boldsymbol{D}\boldsymbol{U} = \mathbf{0}_{3(m-n+1)\times 3m}$, and $\boldsymbol{W} \in \mathbb{R}^{3m\times 3m}$ is any positive definite weighting matrix. As proven in Simon and Chia (2002), if the weight matrix $\boldsymbol{W}$ in Equation (40) is chosen as $\boldsymbol{W} = \boldsymbol{P}^{*-1}$, then the estimate $\hat{\mathbf{z}}_p^*$ in Equation (39) is minimum variance, namely

$$\boldsymbol{P}_p^* \leq \boldsymbol{P}^* \tag{41}$$

being $\boldsymbol{P}_p^*$ the error covariance of $\hat{\mathbf{z}}_p^*$; however, if the weight matrix $\boldsymbol{W}$ is chosen as $\boldsymbol{W} = \boldsymbol{I}$, then the constrained estimate $\hat{\mathbf{z}}_p^*$ in Equation (39) is always closer to the true state than the unconstrained estimate, namely

$$||\hat{\mathbf{z}}^* - \hat{\mathbf{z}}_p^*|| \leq ||\hat{\mathbf{z}}^* - \hat{\mathbf{z}}^*||. \tag{42}$$

Choosing the weight matrix $\boldsymbol{W} = \boldsymbol{P}^{*-1}(k)$, the estimate $\hat{\mathbf{z}}_p^*(k)$ becomes:

$$\hat{\mathbf{z}}_p^*(k) = \hat{\mathbf{z}}^*(k) - \left[\boldsymbol{P}^*(k)\boldsymbol{D}^\top \left(\boldsymbol{D}\boldsymbol{P}^*(k)\boldsymbol{D}^\top\right)^{-1}\right] \boldsymbol{D}\,\hat{\mathbf{z}}^*(k), \tag{43}$$

and resulting posterior covariance is given by

$$\boldsymbol{P}_p^*(k) = \boldsymbol{P}^*(k) - \boldsymbol{P}^*(k)\boldsymbol{D}^\top(\boldsymbol{D}\boldsymbol{P}^*(k)\boldsymbol{D}^\top)^{-1}\boldsymbol{D}\boldsymbol{P}^*(k). \tag{44}$$

From the $\hat{\mathbf{z}}_p^*$ resulting from (39), it is possible to extract the single $\hat{\mathbf{z}}_{ij}$ that appears in the control and estimation equations (16, 17, 18, 20, 21, 28, 25, 31, 32). Hence, the minimum variance estimate $\hat{\mathbf{z}}_p^*$ is actually used by each agent to set its velocity according to the control law (Equation 16). Therefore, we assume that this velocity is kept constant for the fixed time interval T_s, namely it does not change until a new estimation is available.

5.2. Remark

It is worth noting that the output $\bar{y}_{ij}(t)$ defined in Equation (23) depends on the first measurement $y_{ij}(t_0)$. This dependency may affect the robustness of the solution as a single erroneous measurement (e.g., an outlier or a fault signal) at $t = t_0$ would jeopardize the output. This issue can be overcome by periodically resetting the measurement $y(t_0)$ with $y(t)$. This would also prevent possible uncertainties in the knowledge of $\mathbf{v}_{ij}(t)$ from causing an unbounded bias in the displacement $\mathbf{d}_{ij}(t)$ in Equation (21) used to compute $\bar{y}_{ij}(t)$. In the discrete time case, this would correspond to periodically mapping $y_{ij}(0) \rightarrow y_{ij}(k^*)$ as if the measurement had started at step k^* while the state estimate $\hat{\mathbf{z}}_{ij}(k+1|k+1)$ follows its update dynamics. The results presented in the following section refer to the discrete time case with periodic mapping of the initial measurement $y_{ij}(0)$ with $y_{ij}(k-1)$ (i.e., $k^* = k-1$). Consequently, the displacement in Equation (31) becomes $\mathbf{d}_{ij}(k-1) = \sum_{l=k^*}^{k-1} \mathbf{v}_{ij}(l)T_s = \mathbf{v}_{ij}(k-1)T_s$.

6. DELAYS IN RANGE MEASUREMENTS ACQUISITION

One key point to have in mind when dealing with underwater networks is that acoustic communications may be subject to relatively large delays. In particular, communication latency is due to the physics of the communication channel as well as to the specific networking protocol employed. This latter component of the delay may be eventually reduced accepting higher packet loss probabilities. While details about the assessment of the communication latency are not addressed in this work, it should be noted that delays may be significant for larger distances and should be accounted for in the estimation framework. Indeed, this is the case within the approach described in this work where the delay needs to be known.

In this framework, the range measurements available during each step of the estimation process will be $y_{ij}(t-\tau_{ij})$, rather than $y_{ij}(t)$, having denoted with τ_{ij} the time delay in the measurement acquisition due to the acoustic communication network. This arises the problem of how it is possible to obtain the actual range $y_{ij}(t)$ from the knowledge of the delayed measurement $y_{ij}(t-\tau_{ij})$, and the time delay τ_{ij}, in order to properly perform the observer for the relative position estimation $\mathbf{z}_{ij}$. Let consider the intra-vehicle range $y_{ij}(t)$:

$$y_{ij}(t) = \mathbf{z}_{ij}(t)^\top \mathbf{z}_{ij}(t) \tag{45}$$

and its time derivative:

$$\dot{y}_{ij}(t) = 2\,\dot{\mathbf{z}}(t)_{ij}^\top\,\mathbf{z}_{ij}(t) = 2\,\mathbf{v}_{ij}(t)^\top\,\mathbf{z}_{ij}(t) \tag{46}$$

Equation (46) allows computing $y_{ij}(t)$ from the knowledge of $y_{ij}(t-\tau_{ij})$ and τ_{ij} as:

$$y_{ij}(t) = y_{ij}(t-\tau_{ij}) + \int_{t-\tau_{ij}}^{t} 2\,\mathbf{v}_{ij}(\tau)^\top\,\mathbf{z}_{ij}(\tau)\,d\tau \tag{47}$$

Exploiting Equation (47), time delays in the measurements are taken into account mitigating their effects on the estimation process. It is worth highlighting that the sampling time T_s of the Kalman filter should be properly chosen.

As a final remark, it is worth noting that it is not possible to implement Equation (47) as it is because the actual relative positions $\mathbf{z}_{ij}$ are not known, and we use the current estimations $\hat{\mathbf{z}}_{ij}$ instead:

$$y_{ij}(t) = y_{ij}(t-\tau_{ij}) + \int_{t-\tau_{ij}}^{t} 2\,\mathbf{v}_{ij}(\tau)^\top\,\hat{\mathbf{z}}_{ij}(\tau)\,d\tau. \tag{48}$$

The numerical integration of Equation (48) leads to the following discrete-time equation:

$$\begin{aligned} y_{ij}(k) \approx\ & y_{ij}(kT_s - \tau_{ij}) \\ & + \sum_{l=0}^{(\tau_{ij}/dT)} 2\,\mathbf{v}_{ij}(kT_s - \tau_{ij} + l\,dT)^\top\,\hat{\mathbf{z}}_{ij}(kT_s - \tau_{ij} + l\,dT)\,dT. \end{aligned} \tag{49}$$

where dT denotes the integration time. Notice that in spite of the lack of an analytical proof of convergence of Equation (49) to the true measurement y_{ij}, all the numerical results confirm the effectiveness of this approach.

The overall control and estimation procedure is illustrated in Algorithm 1. Summarizing, at each time step the last available constrained estimates of $\mathbf{z}_{ij}$ are used by the control law of each vehicle using Equation (16). Then, the measurements y_{ij} are acquired: in case of delays, the current y_{ij} is estimated through Equation (49). Finally, the observer updates the estimates of the variables $\mathbf{z}_{ij}$ using the constrained Kalman filter solution.

Algorithm 1 Control and estimation algorithm

Require: $\hat{\mathbf{z}}_{ij}(k|k), \boldsymbol{P}_{ij}(k|k), \hat{\mathbf{z}}_{pij}(k|k), \mathbf{Q}(k), R(k), y_{ij}(0), y_{ij}((k+1)T_s - \tau_{ij}), \tau_{ij} : (i,j) \in \mathcal{E}$

Ensure: $\hat{\mathbf{z}}_{ij}(k+1|k+1), \boldsymbol{P}_{ij}(k+1|k+1), \hat{\mathbf{z}}_p(k+1|k+1), \boldsymbol{P}_{pij}(k+1|k+1)$

$\mathbf{v}_i(k) \leftarrow K \sum_{h \in \mathcal{N}_i} \hat{\mathbf{z}}_{pih}(k|k)$

$\mathbf{v}_{ij}(k) \leftarrow K \left(\sum_{h \in \mathcal{N}_i} \hat{\mathbf{z}}_{pih}(k|k) - \sum_{\rho \in \mathcal{N}_j} \hat{\mathbf{z}}_{pj\rho}(k|k) \right)$

$\mathbf{d}_{ij}(k) \leftarrow \sum_{l=0}^{k} \mathbf{v}_{ij}(l) T_s$

if $\tau_{ij} \neq 0$

compute $y_{ij}(k+1)$ from (49)

end

$\bar{y}_{ij}(k+1) \leftarrow \frac{1}{2} \left[y_{ij}(k+1) - y_{ij}(0) + \|\mathbf{d}_{ij}(k)\|^2 \right]$

compute the KF estimation using (32-36): $\hat{\mathbf{z}}_{ij}(k+1|k+1), \boldsymbol{P}_{ij}(k+1|k+1) : (i,j) \in \mathcal{E}$

identify independent geometric constraints in terms of $\boldsymbol{D}$

project KF estimation on constraint equations using (43-44)

return $\hat{\mathbf{z}}_{ij}(k+1|k+1), \boldsymbol{P}_{ij}(k+1|k+1), \hat{\mathbf{z}}_p(k+1|k+1), \boldsymbol{P}_{pij}(k+1|k+1) : (i,j) \in \mathcal{E}$

7. COMMUNICATION PROTOCOLS

In this section, we describe the communication policy that we adopted to perform the range data exchange among the agents during the intersampling period. This step is instrumental to make the computation of Equations (32), (36) possible at each sampling time, and in turn the projection (Equations 43, 44).

Several approaches are possible; we propose two alternative solutions, which we refer to as *centralized* approach and *decentralized* approach. In the centralized approach, only one agent, a *leader* agent, is expected to perform the computation of Equations (32)–(36) and (43), (44) so that the communication policy is organized in order to make the data flow to the leader for the twofold task of reconstructing the topology of the RPMG established and collecting a complete set of range measurement to perform the estimation of Equations (32)–(36). If necessary or useful, the leader agent sends back the resulting estimated positions among the agents using the same scheme reversed. In the decentralized approach, all agents have the capability of performing the computation of (32)–(36) and (43), (44) and hence the communication policy is oriented to spread the range data among agents to distribute them to all, so that each agent performs the computation of the positions estimation. It is worth noting that the term centralized/decentralized is related to the computation of Equations (32)–(36), and hence to the fact that the "holder" of the estimated value is only one agent or any one of the network.

Regardless of the strategy adopted, an issue to consider is that the RPMG cannot be known in advance, and the topology identification of it is instrumental to the computation of Equations (32)–(36). In this respect, we assume that the number of the vehicles n and a preassigned labeling of the agents is known in advance, while the connection topology is unknown to any vehicle and it must be reconstructed as well using any approach.

We assume that all agents involved are equipped with synchronized clocks so as to use One-Way Travel Time (OWTT) range measurement schemas. Then, a Time-Division Multiple Access (TDMA) scheme can be employed to access the communication channel. Under this hypothesis, the communication among agents is unidirectional; this choice is conservative in order to avoid the chance of packet collisions and the management of the resulting loss of data. It should be noted that the duration of the time slots depends on the available bit rate and on the specific communication protocol. Examples of acoustic sensors commonly used in underwater environment are the middle frequency (MF) modems (18–34 kHz) by Evologics (Kebkal et al., 2017). They have been recently used for underwater positioning purposes during geotechnical survey experiments performed within a European project (Abreu et al., 2016). Such modems are characterized by a nominal bit rate in the range 3.10–3.85 kbps, hence compatible with the application at hand.

It is now worth mentioning that the amount of time needed for two agents to communicate using acoustic signals may be significant for large distances and in this paper it is accounted for as described in section 6. Indeed, sound speed underwater is approximately 1500 m/s, namely about six orders of magnitude lower than the speed of electromagnetic signals in air.

According to all the previous considerations, we established our communication policy under the following assumptions. We refer to **Figure 2** as to a description of the idea in the case of $n = 4$ and RPMG as in **Figure 3**. The two strategies are put in a pseudo-algorithm form (in the form of a flowchart) depicted in **Figures 4**, **5**.

We assume that vehicles are organized to send packets one by one. The agents are labeled from the beginning and they follow their labeling in order to send broadcast packets according to the agreed protocol (which depends on the type of approach, this is detailed in the following). Each packet is received only by the neighboring agents and it takes a non-zero travel time to reach the receiver, so we set equal to δt the maximum travel time (which depends on the sensor range and environmental parameters). All agent are aware of the starting time of the estimation procedure, say $\bar{t}$. Agent i sends its packet at time $\bar{t} + (i-1) \cdot \delta t$ and this packet reaches the agent j, $j \in \mathcal{N}_i$ within the time span $(\bar{t} + (i-1) \cdot \delta t, \bar{t} + i \cdot \delta t)$. This is periodically repeated at each $\bar{t} + \kappa T_s$.

In the centralized approach, the range data that are needed to run the filter can be distributed in the team of n members as illustrated in **Figure 4**: agents sequentially (one in each time slot) broadcast a data packet containing their identifying label, a time stamp, all the edges it is aware of, and all the corresponding range data and measurements delays. After all agents but one (i.e., after $n-1$ time slots) execute the protocol, the leader agent collects knowledge about the whole connection topology. Hence, the leader agent knows all the information, i.e., relative distances (6), required to solve the estimation problem taking into account the additional geometric constraints (11) associated with the connection topology. Once computed the estimates $\hat{\mathbf{z}}_p^*$ of the m relative positions using the collected information, the leader agent broadcasts to all agents a data packet containing

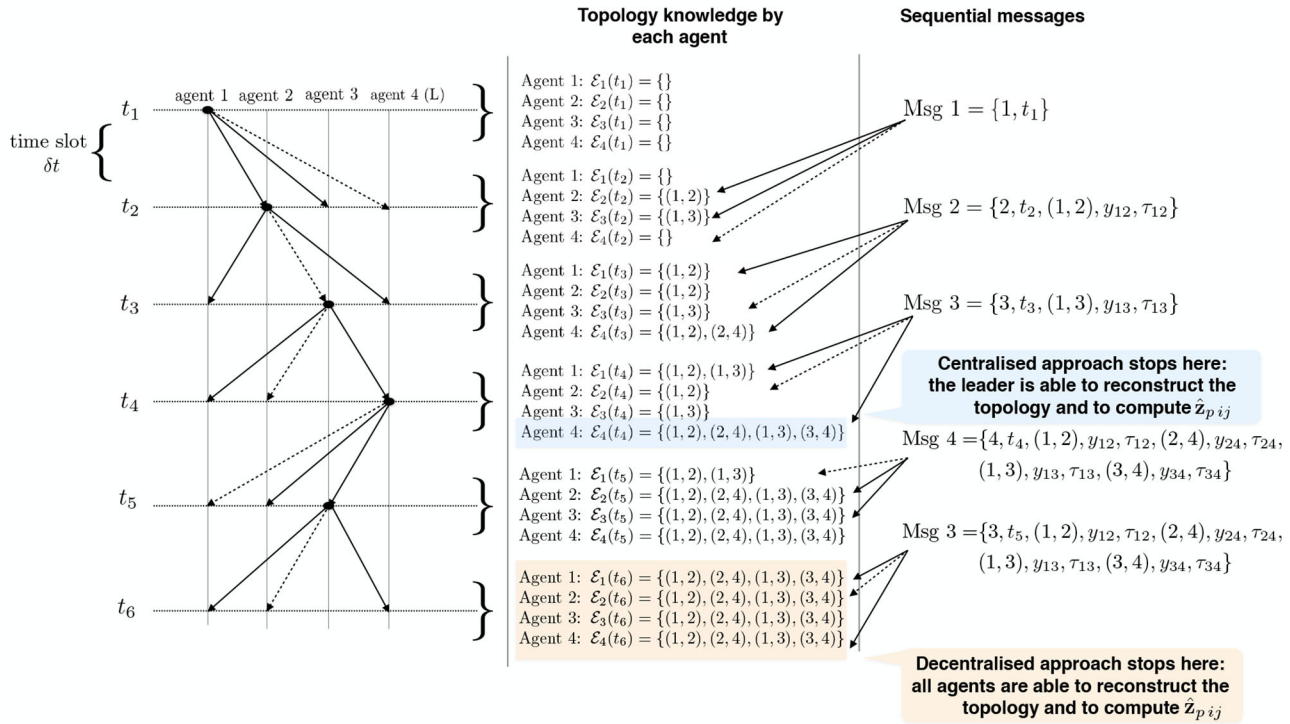


FIGURE 2 | Example of acoustic interrogation schema for acquiring range measurements and identify the connection topology for Relative Position Measurement Graph (RPMG) in **Figure 3**.

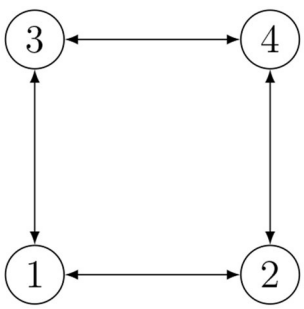


FIGURE 3 | Example of Relative Position Measurement Graph (RPMG) with 4 nodes and 4 edges.

such estimates. Overall, $2n - 2$ time slots are required to complete one estimation step. This kind of scaling appears to be most likely acceptable for most applications involving a limited number of vehicles.

In the decentralized approach, considering that the communication graph is unknown to the agents, the communication policy is implemented with the aim of retrieving the graph topology and spreading the range data to all agents. Each agent during its time slot broadcasts a data packet containing its label, a time stamp, the set of links already identified, and the corresponding range data and measurements delays. All agents receiving the ping, decode the data packet, and identify the link between itself and the transmitter agent. This is repeated until all agents collect knowledge about the whole connection topology. In the worst case, $2n - 2$ communications slots are required to ensure that all agents have identified the connection topology. At this point, each agent can perform the estimation $\hat{\mathbf{z}}_p^*$ of the relative positions.

The main differences between the two approaches can be deduced by the schemes in **Figures 4**, **5**, and we briefly comment them in the following. In the centralized approach, only one agent perform the elaboration, and it can be useful when the team is heterogeneous and some agents have higher computational capacity than others. However, the centralized approach requires a larger amount of communicated data when the estimated state is transmitted to all agents.

8. SIMULATIONS

The proposed range-based mutual localization for a team of underwater vehicles is here tested on the RPMG in **Figure 3** relative to a group of $n = 4$ agents and $m = 4$ communication links. The corresponding geometric constraints are as follows:

$$\boldsymbol{D}\mathbf{z}^* = \begin{bmatrix} \boldsymbol{I}_{3\times3} & \boldsymbol{I}_{3\times3} & \boldsymbol{I}_{3\times3} & -\boldsymbol{I}_{3\times3} \end{bmatrix} \begin{bmatrix} \mathbf{z}_{12} \\ \mathbf{z}_{24} \\ \mathbf{z}_{43} \\ \mathbf{z}_{13} \end{bmatrix} = \mathbf{0}_{3\times1}. \tag{50}$$

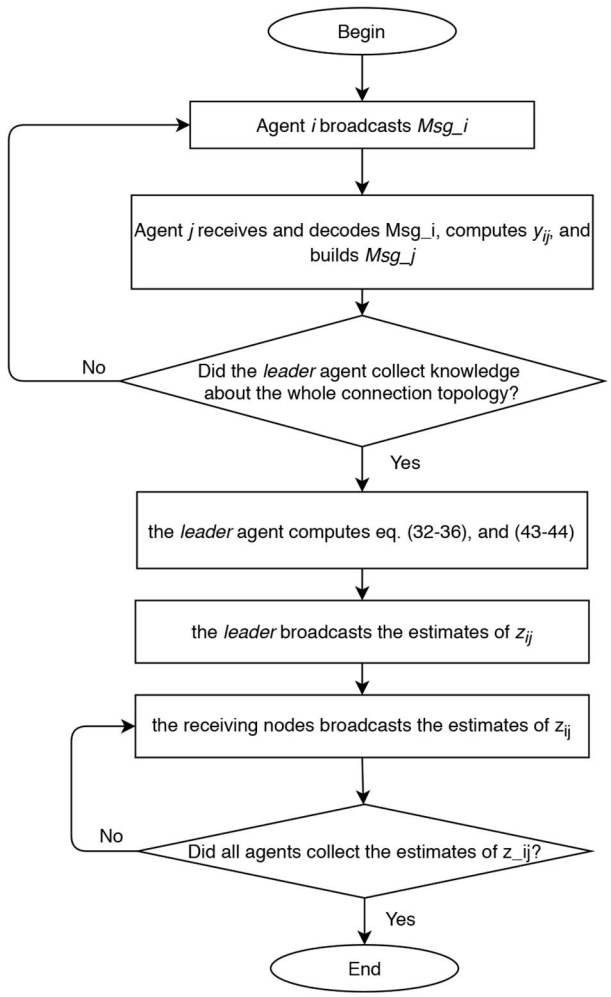


FIGURE 4 | Flow chart of centralized approach for range measurements acquisition and relative positions estimation.

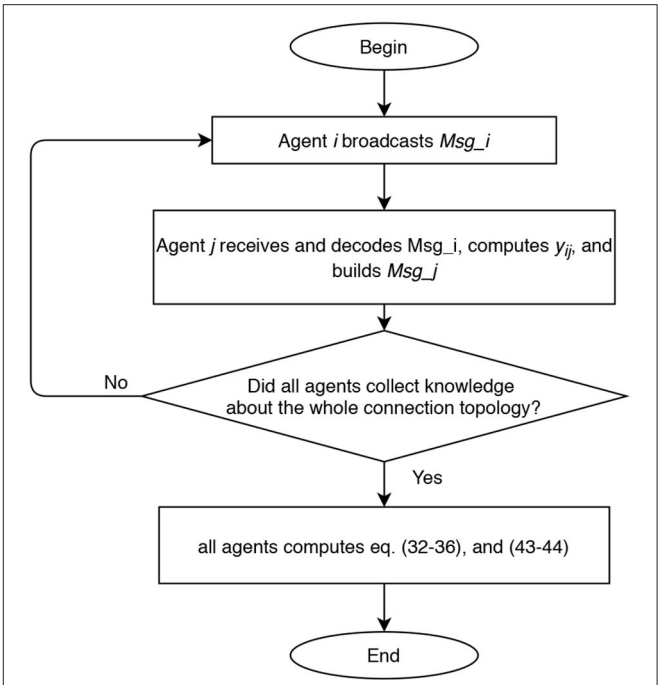


FIGURE 5 | Flow chart of decentralized approach for range measurements acquisition and relative positions estimation.

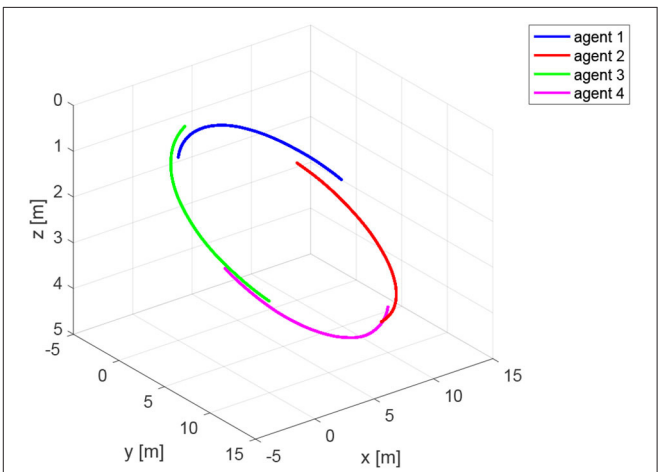


FIGURE 6 | Trajectories of the agents for the simulation based on Relative Position Measurement Graph (RPMG) in **Figure 3**.

The velocity inputs of each agent are assigned according to the localization-oriented control law in Equation (16) with $K = 0.1$:

$$\mathbf{v}_1(k) = -K(\mathbf{z}_{12}(k) + \mathbf{z}_{31}(k));$$
$$\mathbf{v}_2(k) = -K(\mathbf{z}_{12}(k) + \mathbf{z}_{24}(k));$$
$$\mathbf{v}_3(k) = -K(\mathbf{z}_{31}(k) + \mathbf{z}_{43}(k));$$
$$\mathbf{v}_4(k) = -K(\mathbf{z}_{24}(k) + \mathbf{z}_{43}(k)).$$

The agents are located in the following initial positions: $\mathbf{x}_1(0) = (0, 0, 1)^\top$m; $\mathbf{x}_2(0) = (10, 0, 2)^\top$m; $\mathbf{x}_3(0) = (0, 10, 3)^\top$m; $\mathbf{x}_4(0) = (10, 10, 4)^\top$m. Without loss of generality, the range measurements are assumed to be acquired with different time delays τ_{ij}, namely $\tau_{12} = 0.1$ s, $\tau_{24} = 0.2$ s, $\tau_{31} = 0.3$ s, $and \tau_{43} = 0.4$ s, whereas a sampling time $T_s = 0.4\,s$ has been considered. At each sampling time T_s, the actual range $y_{ij}(t)$ is derived from the knowledge of the delayed measurement $y_{ij}(t - \tau_{ij})$, and the time delay τ_{ij} according to Equation (49). The resulting trajectories are shown in **Figure 6**. It is worth remarking that the proposed agents velocities $\mathbf{v}_i$ guarantee the observability of the system (Equations 25 and 26). Indeed, it can be verified, by direct calculation, that the motion generated by the control law (16) verifies the full rank condition on the observability Gramian (27) of the system, i.e., $\text{rank}(\boldsymbol{G}) = 3\,m = 12$. **Figure 7** shows the rank of the Gramian along the trajectory, and a few range acquisitions are needed to get a full rank Gramian matrix. Therefore, given the observability of the system, the states $\mathbf{z}_{ij}$ can be estimated using the Kalman observer in Equations (32)–(36), with covariance $\boldsymbol{Q} = 0.9\,10^{-5} \cdot \text{diag}(1, 1, 1)\text{m}^2$, covariance $R = 0.25\text{m}^2$, and initial condition given by

$$\hat{\mathbf{z}}_{ij}(0) \sim \mathcal{N}(\mathbf{z}_{ij}(0), \boldsymbol{P}_{ij}(0)), \quad \boldsymbol{P}_{ij}(0) = 4 \cdot \text{diag}(1, 1, 1)\text{m}^2. \quad (51)$$

The estimate $\hat{\mathbf{z}}^* = (\hat{\mathbf{z}}_{12}^\top \ \hat{\mathbf{z}}_{24}^\top \ \hat{\mathbf{z}}_{43}^\top \ \hat{\mathbf{z}}_{13}^\top)^\top \in \mathbb{R}^{12}$ obtained using the Kalman filter is reported in **Figure 8A**. This estimate violates the equality constraint (50). A constrained state estimate can be obtained projecting the standard Kalman filter estimate $\hat{\mathbf{z}}^*$ onto the constraint surface through Equations (43), (44). This leads to the projected estimation illustrated in **Figure 8B**. **Figure 9** reports the norm of the equality constraints,

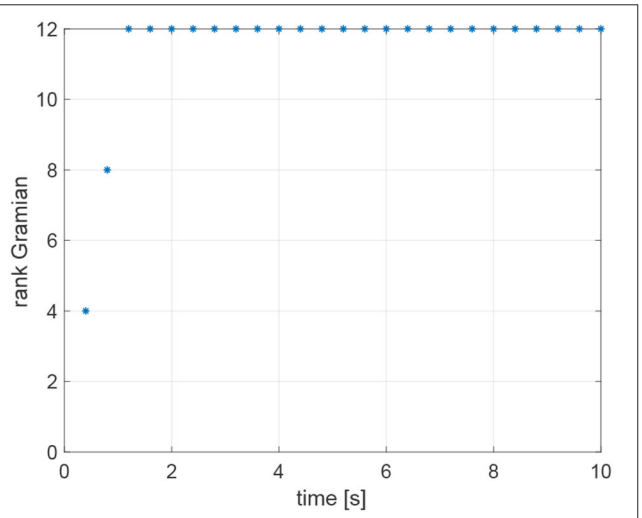


FIGURE 7 | Rank of the observability Gramian for the simulation based on Relative Position Measurement Graph (RPMG) in **Figure 3**.

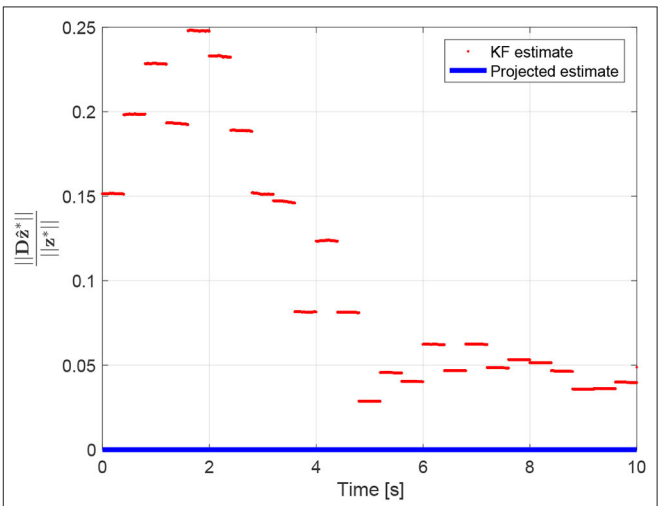


FIGURE 9 | Equality constraint $\frac{\|D\hat{\mathbf{z}}^*\|}{\|\mathbf{z}^*\|} = 0$.

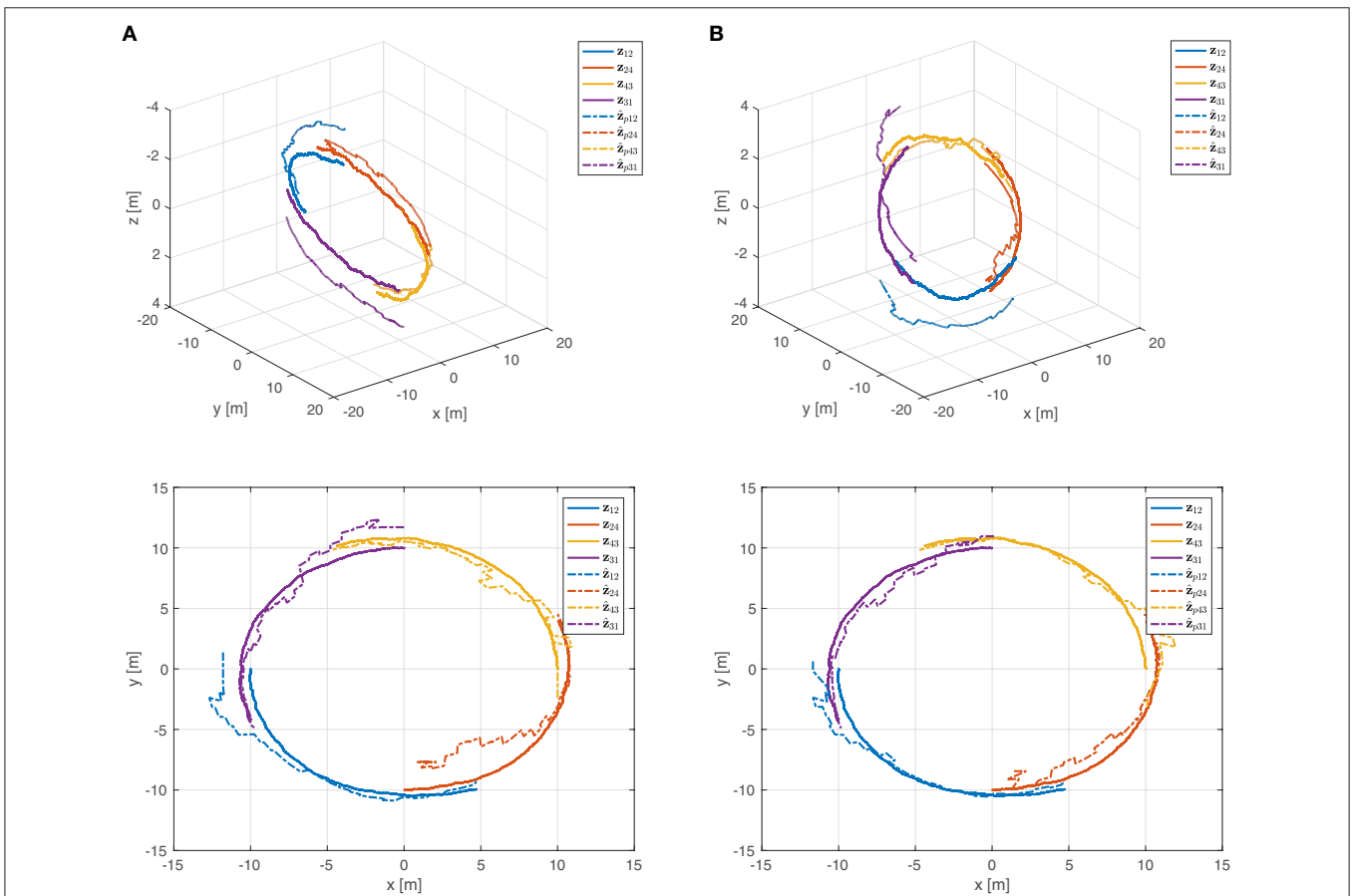


FIGURE 8 | (A) Estimation of the relative motions in 3D (top) and 2D (bottom). **(B)** Constrained estimation of the relative motions in 3D (top) and 2D (bottom).

$\frac{\|D\hat{\mathbf{z}}^*\|}{\|\mathbf{z}^*\|}$. It is worth noting that the unconstrained Kalman estimate (red line) does not satisfy exactly the constraint, whereas the constrained Kalman estimate (blue line) satisfies the equality constraint, namely $\frac{\|D\hat{\mathbf{z}}^*\|}{\|\mathbf{z}^*\|} = 0$. Moreover, as expected, the constrained estimate is also characterized by a reduced covariance, i.e., $\boldsymbol{P}_p^* - \boldsymbol{P}^* < 0$. Indeed, the maximum eigenvalue of the matrix $\boldsymbol{P}_p^* - \boldsymbol{P}^*$, shown in **Figure 10**, is always negative (not positive), confirming the improvement obtained by exploiting the additional information provided by the geometric constraints.

Note that regarding the norm of the estimation error, even if the weight $\boldsymbol{W}$ is chosen as $\boldsymbol{W} = \boldsymbol{P}^{*-1}(k)$ rather than $\boldsymbol{W} = \boldsymbol{I}$, the projected estimates still provide better results with respect to the corresponding unconstrained Kalman estimates. The norm of the estimation error for both estimates is reported in **Figure 10**.

The benefits of including the geometric constraints into the estimation framework are more evident in **Figure 11**, where the estimation error and the corresponding uncertainty region of each component of the state $\mathbf{z}^* \in \mathbb{R}^{12}$ are shown for both estimates, unconstrained and constrained. The estimation errors and the uncertainty region for the first component of the state $\mathbf{z}^* \in \mathbb{R}^{12}$ are also depicted in **Figure 12**. It is interesting to note that, as already highlighted, the uncertainty region of the constrained estimates is smaller than the one related to the standard Kalman estimate. As a final remark, it is worth noting that the estimates in **Figures 8**, **11**, **12**, as well as the equality constraint in **Figure 9**, have sharp leaps whenever a range update is processed, i.e., every T_s seconds. Moreover, the management of the delays in range measurements as described in section 6 allows to correctly process the measurements without compromise the convergence of the estimations.

A further simulation is undertaken on the more complex RPMG illustrated in **Figure 13** relative to a group of $n = 4$ agents

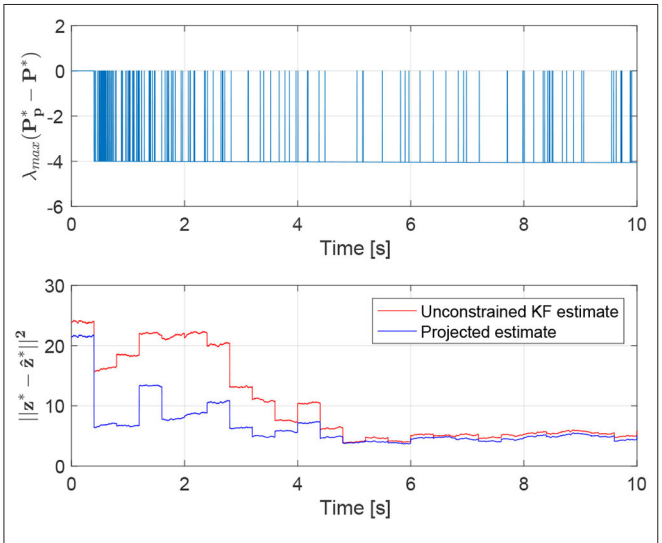


FIGURE 10 | Maximum eigenvalue of the matrix $\boldsymbol{P}_p^*(k) - \boldsymbol{P}^*(k)$ **(Top)** and norm of the estimation errors **(Bottom)**.

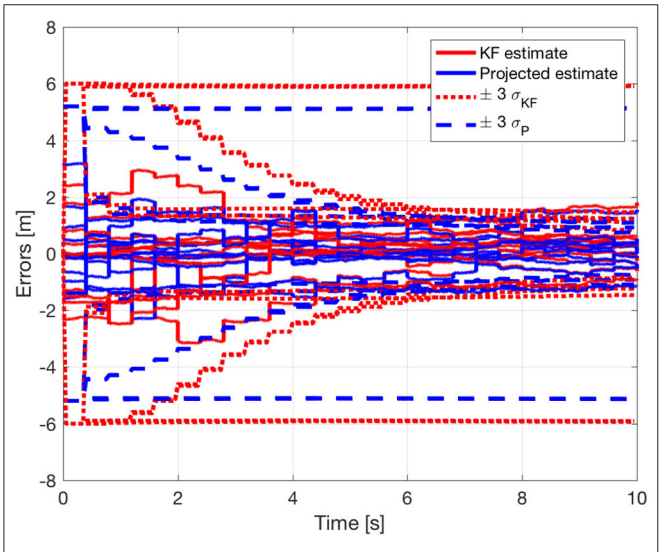


FIGURE 11 | Estimation errors and uncertainty regions for the relative positions estimations $\mathbf{z}^*$.

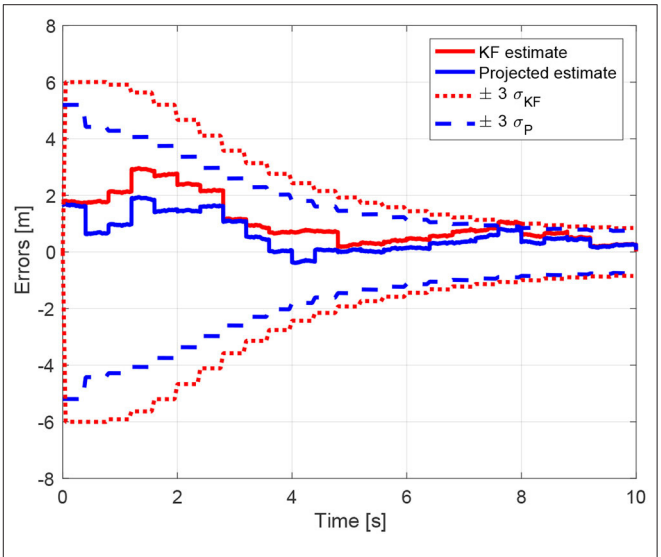


FIGURE 12 | Estimation errors and uncertainty regions for the relative positions estimations of the first component of $\mathbf{z}^*$.

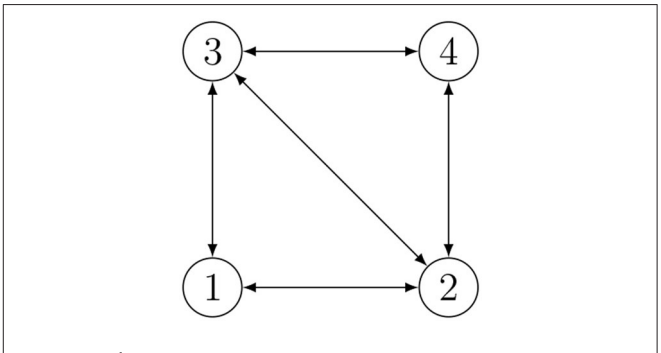


FIGURE 13 | RPMG with $n = 4$ agents and $m = 5$ communication links.

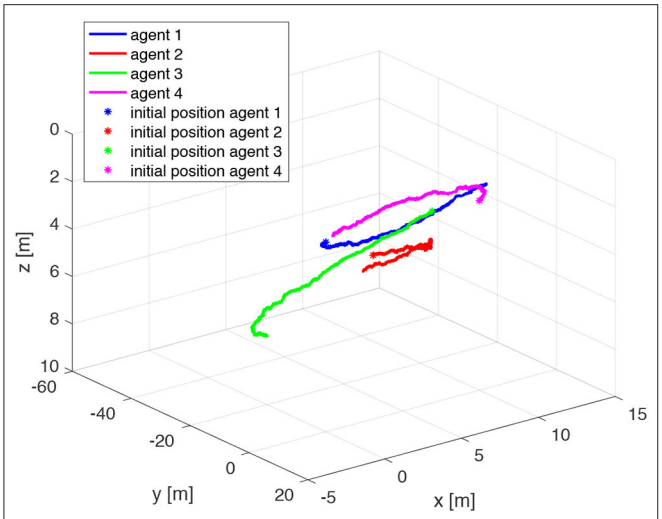


FIGURE 14 | Trajectories of the agents for the simulation based on Relative Position Measurement Graph (RPMG) in **Figure 13**.

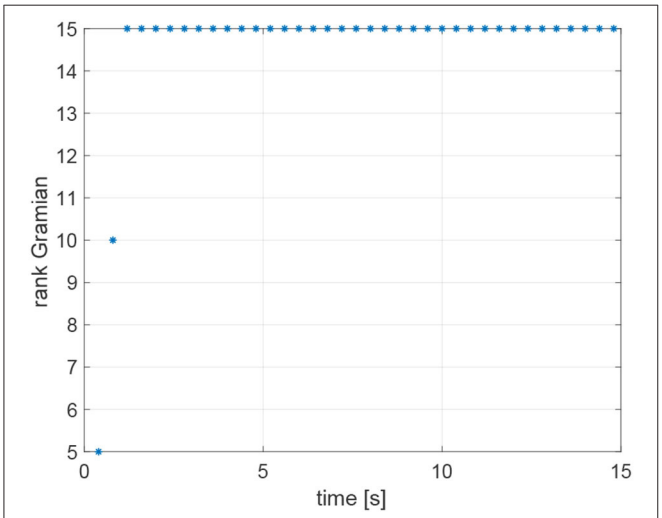


FIGURE 15 | Rank of the observability Gramian for the simulation based on Relative Position Measurement Graph (RPMG) in **Figure 13**.

and $m = 5$ communication links. The velocity inputs of each agent are assigned according to the localization-oriented control law in Equation (16) with $K = 0.1$:

$$\mathbf{v}_1(k) = -K(\mathbf{z}_{12}(k) + \mathbf{z}_{31}(k));$$
$$\mathbf{v}_2(k) = -K(\mathbf{z}_{12}(k) + \mathbf{z}_{24}(k) + \mathbf{z}_{32}(k));$$
$$\mathbf{v}_3(k) = -K(\mathbf{z}_{31}(k) + \mathbf{z}_{43}(k) + \mathbf{z}_{32}(k));$$
$$\mathbf{v}_4(k) = -K(\mathbf{z}_{24}(k) + \mathbf{z}_{43}(k)).$$

The initial positions of the agents are $\mathbf{x}_1(0) = (0, 0, 2)^\top$ m; $\mathbf{x}_2(0) = (5, -10, 4)^\top$ m; $\mathbf{x}_3(0) = (5, 10, 1)^\top$ m; $\mathbf{x}_4(0) = (10, 0, 2)^\top$ m. The range measurements are assumed to be acquired with different time delays τ_{ij}, namely $\tau_{12} = 0.1$ s, $\tau_{24} = 0.2$ s, $\tau_{31} = 0.3$ s, $\tau_{23} = 0.3$ s, $\tau_{43} = 0.4$ s, whereas a sampling time $T_s = 0.4\,s$ has been considered. The resulting trajectories are shown in **Figure 14**. Notice that the observability Gramian (27) of the system is full rank along the trajectory of the vehicles, indeed $\text{rank}(\boldsymbol{G}) = 3m = 15$ as shown in **Figure 15**. Therefore, given the observability of the system, the states $\mathbf{z}_{ij}$ can be estimated using the Kalman observer in Equations (32)–(36). The covariances of the state $\mathbf{z}_{ij}$ and the output $\bar{y}$ employed in the Kalman filter are $\boldsymbol{Q} = 0.9\,10^{-5} \cdot \text{diag}(1, 1, 1)\,\text{m}^2$ and $R = 0.25\,\text{m}^2$, respectively. The initial Kalman filter state estimate is given by

$$\hat{\mathbf{z}}_{ij}(0) \sim \mathcal{N}(\mathbf{z}_{ij}(0), \boldsymbol{P}_{ij}(0)), \quad \boldsymbol{P}_{ij}(0) = 9 \cdot \text{diag}(1, 1, 1)\,\text{m}^2, \quad (52)$$

namely, $\mathbf{z}_{ij}(0)$ is the initial true state and the initial condition $\hat{\mathbf{z}}_{ij}(0)$ of the filter is assigned randomly with covariance $\boldsymbol{P}_{ij}(0)$.

Regardless of the specific approach adopted for the acoustic communications among agents, i.e., centralized or decentralized, the estimation of the relative positions $\mathbf{z}^* = (\mathbf{z}_{12}^\top\ \mathbf{z}_{24}^\top\ \mathbf{z}_{43}^\top\ \mathbf{z}_{13}^\top; \mathbf{z}_{23}^\top)^\top \in \mathbb{R}^{15}$ is based on the Algorithm 1. **Figure 16** reports the ultimate constrained Kalman filter estimate $\hat{\mathbf{z}}_p^*$ of the relative motions. It is worth remarking that, given the global observability of the motion, even if the estimations are initialized with a value far from the real one (see Equation 52), the resulting $\hat{\mathbf{z}}_{pij}$ converge to $\mathbf{z}_{ij}$. This is an interesting feature because the proposed localization-oriented control law can actually be activated when the relative localization accuracy of agents is poor. Indeed, adopting such control strategy the whole agents network improves significantly its formation accuracy.

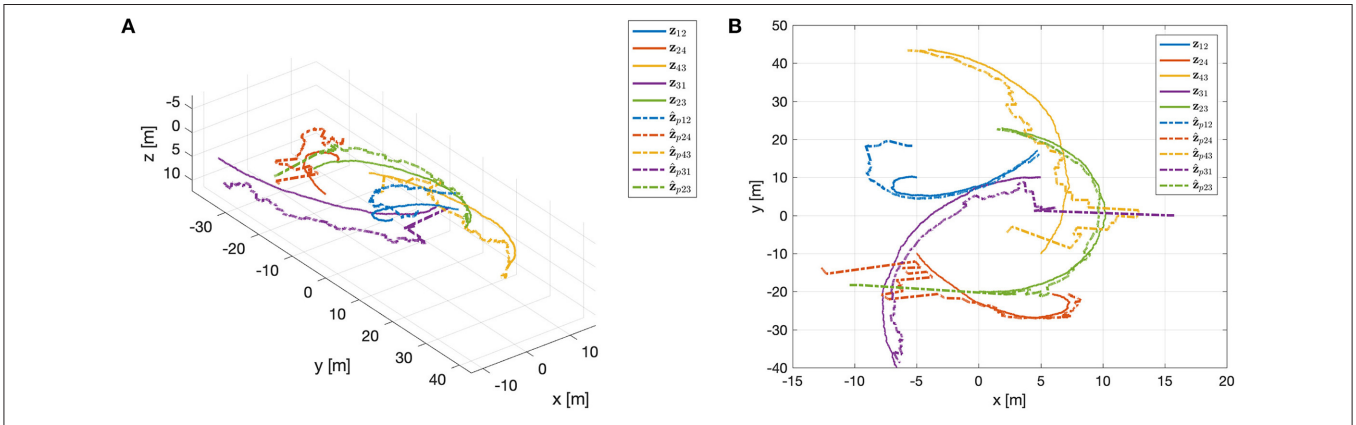


FIGURE 16 | Constrained Kalman filter estimation of the relative motions: **(A)** 3D and **(B)** 2D.

9. CONCLUSIONS

In this paper, the relative localization estimation problem for a team of vehicles is studied based on the knowledge of the range between agents of a subset of the participants. One main peculiarity of the proposed approach is that the relative velocity between agents, which is a fundamental data to solve the problem, is not assumed to be known in advance neither directly communicated. For this reason, a collaborative control protocol is designed in order to encapsulate the velocity data in the motion of each vehicle as a parameter through a dedicated control protocol, so that it can be inferred from the motion of the neighbor agents. Moreover, some suitable geometrical constraints associated with the agents' (unknown) positions are built and explicitly accounted for in the estimation schema providing a more accurate estimate. The issue of possible delays in the transmitted signals is also studied and two possible solutions are provided explaining how it is possible to get a reasonable range data exchange to get the solution both in a centralized fashion and in a decentralized one. Finally, the validity of the proposed approach is shown through numerical simulations.

DATA AVAILABILITY STATEMENT

The datasets generated for this study are available on request to the corresponding author.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct and intellectual contribution to the work, and approved it for publication.

FUNDING

This work was partially supported by the European Union's Horizon 2020 research and innovation program under the project EUMarineRobots: Marine robotics research infrastructure network, grant agreement N.731103 (call H2020-INFRAIA-2017-1-two-stage).

REFERENCES

Abreu, P., Antonelli, G., Arrichiello, F., Caffaz, A., Caiti, A., Casalino, G., et al. (2016). Widely scalable mobile underwater sonar technology: an overview of the H2020 WiMUST project. *Mar. Technol. Soc. J.* 50, 42–53. doi: 10.4031/MTSJ.50.4.3

Allotta, B., Costanzi, R., Meli, E., Pugi, L., Ridolfi, A., and Vettori, G. (2014). Cooperative localization of a team of AUVs by a tetrahedral configuration. *Robot. Auton. Syst.* 62, 1228–1237. doi: 10.1016/j.robot.2014.03.004

Antonelli, G., Arrichiello, F., Caiti, A., Casalino, G., De Palma, D., Indiveri, G., et al. (2018). ISME activity on the use of autonomous surface and underwater vehicles for acoustic surveys at sea. *Acta IMEKO* 7, 24–31. doi: 10.21014/acta_imeko.v7i2.539

Arai, T., Pagello, E., and Parker, L. E. (2002). Editorial: advances in multi-robot systems. *IEEE Trans. Robot. Automat.* 18, 655–661. doi: 10.1109/TRA.2002.806024

Bahr, A., Leonard, J. J., and Fallon, M. F. (2009). Cooperative localization for autonomous underwater vehicles. *Int. J. Robot. Res.* 28, 714–728. doi: 10.1177/0278364908100561

Batista, P., Silvestre, C., and Oliveira, P. (2011). Single range aided navigation and source localization: observability and filter design. *Syst. Control Lett.* 60, 665–673. doi: 10.1016/j.sysconle.2011.05.004

Bayat, M., Crasta, N., Aguiar, A., and Pascoal, A. (2016). Range-based underwater vehicle localization in the presence of unknown ocean currents: theory and experiments. *IEEE Trans. Control Syst. Technol.* 24, 122–139. doi: 10.1109/TCST.2015.2420636

Cao, M., Yu, C., and Anderson, B. D. O. (2011). Formation control using range-only measurements. *Automatica* 47, 776–781. doi: 10.1016/j.automatica.2011.01.067

De Palma, D., Arrichiello, F., Parlangeli, G., and Indiveri, G. (2017). Underwater localization using single beacon measurements: observability analysis for a double integrator system. *Ocean Eng.* 142, 650–665. doi: 10.1016/j.oceaneng.2017.07.025

De Palma, D., Indiveri, G., and Parlangeli, G. (2015). "Multi-vehicle relative localization based on single range measurements," in *IFAC-PapersOnLine, Proceedings of the 3rd IFAC Workshop on MultiVehicle System - MVS 2015* (Genova), 17–22. doi: 10.1016/j.ifacol.2015.06.457

De Palma, D., Indiveri, G., and Parlangeli, G. (2019). "Multi-vehicle cooperative range based navigation," in *Cooperative Localization and Navigation: Theory, Research, and Practice* eds C. Gao, G. Zhao, and H. Fourati (Boca Raton, FL: Taylor & Francis Ltd.), 636. doi: 10.1201/9780429507229-13

Fallon, M. F., Papadopoulos, G., Leonard, J. J., and Patrikalakis, N. M. (2010). Cooperative AUV navigation using a single maneuvering surface craft. *Int. J. Robot. Res.* 29, 1461–1474. doi: 10.1177/0278364910380760

Ferri, G., Munafó, A., Tesei, A., Braca, P., Meyer, F., Pelekanakis, K., et al. (2017). Cooperative robotic networks for underwater surveillance: an overview. *IET Radar Sonar Navigat.* 11, 1740–1761. doi: 10.1049/iet-rsn.2017.0074

Gadre, A. S., and Stilwell, D. J. (2004). "Toward underwater navigation based on range measurements from a single location," in *Proceedings of IEEE International Conference on Robotics and Automation, 2004 (ICRA 2004)* (New Orleans, LA), 4472–4477. doi: 10.1109/ROBOT.2004.1302422

Godsil, C. D., and Royle, G. (2001). *Algebraic Graph Theory, Vol. 207*. New York, NY: Springer. doi: 10.1007/978-1-4613-0163-9

Halsted, T., and Schwager, M. (2017). "Distributed multi-robot localization from acoustic pulses using Euclidean distance geometry," in *2017 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)* (Los Angeles, CA: IEEE), 104–111. doi: 10.1109/MRS.2017.8250938

Hermann, R., and Krener, A. J. (1977). Nonlinear controllability and observability. *IEEE Trans. Automat. Control* 22, 728–740. doi: 10.1109/TAC.1977.1101601

Indiveri, G., De Palma, D., and Parlangeli, G. (2016). Single range localization in 3-D: observability and robustness issues. *IEEE Trans. Control Syst. Technol.* 24, 1853–1860. doi: 10.1109/TCST.2015.2512879

Jouffroy, J., and Reger, J. (2006). "An algebraic perspective to single-transponder underwater navigation," in *Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control* (IEEE), 1789–1794. doi: 10.1109/CACSD-CCA-ISIC.2006.4776912

Kebkal, K., Kebkal, O., Kebkal, V., Sebastião, L., Pascoal, A., Ribeiro, J., et al. (2017). "Performance assessment of underwater acoustic modems operating simultaneously at different frequencies in the presence of background impulsive noise emitted by a sparker," in *Proc. UACE 2017 4th Underwater Acoustics Conference and Exhibition* (Skiathos), 315–323.

Leonard, N., Paley, D., Lekien, F., Sepulchre, R., Fratantoni, D., and Davis, R. (2007). Collective motion, sensor networks and ocean sampling. *Proc. IEEE* 95, 48–74. doi: 10.1109/JPROC.2006.887295

Mourikis, A. I., and Roumeliotis, S. I. (2006). Performance analysis of multirobot cooperative localization. *IEEE Trans. Robot.* 22, 666–681. doi: 10.1109/TRO.2006.878957

Olfati-Saber, R., and Murray, R. M. (2004). Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. Automatic Control* 49, 1520–1533. doi: 10.1109/TAC.2004.834113

Ren, W., and Beard, R. (2007). *Distributed Consensus in Multi-vehicle Cooperative Control: Theory And Applications*. London: Springer-Verlag. doi: 10.1007/978-1-84800-015-5

Ren, W., and Cao, Y. (2010). *Distributed Coordination of Multi-Agent Networks: Emergent Problems, Models, and Issues*. London: Springer Science & Business Media. doi: 10.1007/978-0-85729-169-1

Ross, A., and Jouffroy, J. (2005). "Remarks on the observability of single beacon underwater navigation," in *Int. Symp. on Unmanned Untethered Submersible Technology (UUST 05)* (Durham, NH).

Roumeliotis, S. I., and Bekey, G. A. (2000). "Collective localization: a distributed Kalman filter approach to localization of groups of mobile robots," in *IEEE International Conference on Robotics and Automation, 2000, ICRA'00* (San Francisco, CA: IEEE), 2958–2965. doi: 10.1109/ROBOT.2000.846477

Roumeliotis, S. I., and Bekey, G. A. (2002). Distributed multirobot localization. *IEEE Trans. Robot. Automat*. 18, 781–795. doi: 10.1109/TRA.2002.803461

Sanderson, A. C. (1997). A distributed algorithm for cooperative navigation among multiple mobile robots. *Adv. Robot.* 12, 335–349. doi: 10.1163/156855398X00235

Sarras, I., Marzat, J., Bertrand, S., and Piet-Lahanier, H. (2017). "Collaborative multi-vehicle localization with respect to static/dynamic target from range and velocity measurements," in *International Conference on Unmanned Aircraft Systems (ICUAS)* (Miami, FL) 850–859. doi: 10.1109/ICUAS.2017.7991392

Scherbatyuk, A. (1995). "The AUV positioning using ranges from one transponder LBL," in *OCEANS '95. MTS/IEEE. Challenges of Our Changing Global Environment* (San Diego, CA), 1620–1623. doi: 10.1109/OCEANS.1995.528728

Simetti, E., Wanderlingh, F., Casalino, G., Indiveri, G., and Antonelli, G. (2017). "Robust project: control framework for deep sea mining exploration," in *OCEANS 2017* (Anchorage), 1–5. doi: 10.1109/OCEANSE.2017.8084888

Simon, D. (2006). *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. John Wiley & Sons. doi: 10.1002/0470045345

Simon, D., and Chia, T. L. (2002). Kalman filtering with state equality constraints. *IEEE Trans. Aerospace Electron. Syst*. (Hoboken, NJ) 38, 128–136. doi: 10.1109/7.993234

Soares, C., Ji, P., Gomes, J., and Pascoal, A. (2017). "DIeSEL: distributed self-localization of a network of underwater vehicles," in *OCEANS 2017* (Anchorage), 1–6.

Soares, J. M., Aguiar, A. P., Pascoal, A. M., and Martinoli, A. (2013). "Joint ASV/AUV range-based formation control: theory and experimental results," in *2013 IEEE International Conference on Robotics and Automation (ICRA 2013)* (Karlsruhe), 5579–5585. doi: 10.1109/ICRA.2013.6631378

Webster, S. E., Walls, J. M., Whitcomb, L. L., and Eustice, R. M. (2013). Decentralized extended information filter for single-beacon cooperative acoustic navigation: theory and experiments. *IEEE Trans. Robot*. 29, 957–974. doi: 10.1109/TRO.2013.2252857

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Advantages of publishing in Frontiers

OPEN ACCESS
Articles are free to read for greatest visibility and readership

FAST PUBLICATION
Around 90 days from submission to decision

HIGH QUALITY PEER-REVIEW
Rigorous, collaborative, and constructive peer-review

TRANSPARENT PEER-REVIEW
Editors and reviewers acknowledged by name on published articles

Frontiers
Avenue du Tribunal-Fédéral 34
1005 Lausanne | Switzerland

Visit us: www.frontiersin.org
Contact us: frontiersin.org/about/contact

REPRODUCIBILITY OF RESEARCH
Support open data and methods to enhance research reproducibility

DIGITAL PUBLISHING
Articles designed for optimal readership across devices

FOLLOW US
@frontiersin

IMPACT METRICS
Advanced article metrics track visibility across digital media

EXTENSIVE PROMOTION
Marketing and promotion of impactful research

LOOP RESEARCH NETWORK
Our network increases your article's readership