

Custom PX4 firmware for autonomous hybrid aerial-marine missions

Andrea Capuozzo, Fabio Ruggiero, Vincenzo Lippiello

Abstract—Tasks aimed at mapping and monitoring aquatic environments often involve multiple operational phases that can be performed more efficiently, safely, and cost-effectively through the use of autonomous hybrid aerial–amphibious drones. This paper presents an enhanced version of the PX4 autopilot firmware specifically tailored for such hybrid platforms, enabling operators to plan and execute missions, including aerial flight and water-surface navigation, in a fully autonomous manner. The proposed firmware extension introduces two dedicated control modes for manual and autonomous marine navigation while preserving full compatibility with the standard PX4 flight modalities and incorporating the required safety mechanisms. Integration with QGroundControl has also been ensured to support seamless mission planning and mode management. The effectiveness of the proposed functionalities has been validated through simulated case studies. The code is freely released for the community.

I. INTRODUCTION

Mapping and monitoring aquatic environments are costly operations that require skilled personnel, specialized equipment, dedicated support infrastructures, and significant logistical resources. In addition, traditional inspection campaigns expose operators to safety risks and can negatively impact the investigated ecosystems, for instance, through water and noise pollution caused by support vessels. Advances in robotics [1], [2] helped mitigating these limitations by allowing operations to be conducted in safer, more efficient, and less invasive ways.

Hybrid aerial–amphibious drones, also referred to as unmanned aerial–aquatic vehicles (UAAVs), represent a versatile class of robotic platforms operating both in flight and on the water surface within a unified design. UAAVs can combine aerial inspection of target areas with water-based operations, including landing to deploy probes, collect samples, perform close-range measurements, and navigate on the surface.

Despite the increasing interest in hybrid robotic platforms, available open-source autopilot software—such as PX4 [3], ArduPilot [4], iNav [5], or Betaflight [6]—primarily targets single-domain vehicles and supports aerial and underwater systems as separate configurations. As a result, their capability to seamlessly integrate aerial flight and water-surface navigation within a unified control and mission-execution architecture remains limited.

This paper addresses this limitation by introducing custom modules and control modes within the PX4 firmware, enabling full integration with its existing functionalities while preserving the standard control architecture. The PX4¹ was selected due to its modular structure, compatibility with multiple flight-controller boards, and permissive Berkeley Software Distribu-



Fig. 1: Rendering of the UAAV reference model used in this work.

tion (BSD) license, which allows firmware extensions without mandatory upstream contribution. The proposed code is freely available to the community²

The remainder of this paper is organized as follows. Section II discusses related work on marine survey drones and autopilot firmware frameworks. Section III presents the firmware customization and navigation mode design. Section IV describes the control strategies for hybrid aerial–marine operations. Section V reports results from simulated scenarios, while Section VI discusses the findings. Finally, Section VII concludes the paper and outlines future work.

II. RELATED WORKS

Several approaches have been proposed in the literature to employ aerial and surface robotic platforms in marine inspections. Unmanned aerial vehicles (UAVs) are widely adopted as rapid and cost-effective tools for large-scale inspection and mapping of marine habitats [7], leveraging their ability to efficiently cover extensive areas from an elevated viewpoint. Complementarily, unmanned surface vehicles (USVs) and unmanned underwater vehicles (UUVs) are used to perform close-range inspection and data collection tasks. For instance, [8] presents a cooperative system composed of a UAV, a UUV, and a USV to inspect floating structures from multiple perspectives before allowing manned vessels to approach.

Open-source USV platforms based on frameworks such as ArduPilot have demonstrated improvements in efficiency and safety for surface water sampling tasks [9], [10]. These systems highlight the potential of robotic platforms to reduce operational costs while maintaining high-quality environmental

The authors are with the PRISMA Lab, Department of Electrical Engineering and Information Technology, University of Naples Federico II, Via Claudio 21, Naples, 80125, Italy. Corresponding author's email: andrea.capuozzo@unina.it

¹The custom version presented in this work is based on the release v1.15.0

²The link to the repository will be disclosed after acceptance. [\[FR: È stato accettato, quindi serve il link!\]](#)

data acquisition [10]. More generally, the combination of aerial and surface viewpoints has been recognized as beneficial for enhancing situational awareness and navigation performance during marine inspection missions [12], [13].

Despite these advances, most existing solutions rely either on multi-vehicle cooperation—where distinct platforms operate under separate control architectures—or on single-domain autopilot configurations tailored specifically for aerial or surface navigation. Even in the case of hybrid platforms, such as the multicopter–hovercraft system presented in [14], the aerial and marine functionalities are handled by separate PX4-based flight-control architectures. While effective, this approach introduces architectural fragmentation, increased system complexity, and the absence of a unified mission execution framework capable of seamlessly managing transitions between aerial and surface phases.

Overall, although the relevance of hybrid aerial–marine operations is increasingly recognized, a firmware-level integration that unifies flight and marine navigation within a single control architecture remains largely unexplored. This motivates the development presented in this work.

A. Contributions

To the best of the authors’ knowledge, the development of a unified PX4 firmware extension enabling both aerial flight and marine surface navigation has not yet been explicitly addressed in the existing literature.

The main contributions of this paper are (i) the design and implementation of the logic and software architecture required to safely and seamlessly integrate the novel marine navigation control module within the standard PX4 framework; (ii) the extension of the PX4 navigator module to accommodate the execution of hybrid aerial–marine operations; (iii) the development of dedicated control strategies for both manual and autonomous water-surface navigation; (iv) the integration of the proposed modes into QGroundControl³ (QGC) [15] achieved by introducing an updated user interface that enables intuitive mission planning and clear differentiation between aerial and marine waypoints.

III. FIRMWARE ARCHITECTURE

The proposed PX4 firmware extension introduces a dedicated marine navigation module supporting both manual and autonomous operation. The module manages marine-specific navigation logic, mode transitions, and thruster actuation. In addition, the PX4 Navigator module has been extended to support hybrid aerial–marine mission execution. Supporting interfaces ensure that the new modes are properly exposed within QGroundControl, allowing intuitive mode selection and clear distinction between aerial and marine waypoints.

³QGC is the default ground control station for PX4, which provides users with tools to manage settings, parameters, sensors, and logs, as well as to plan missions through a graphical interface

A. Marine navigation module

The marine navigation module is the key component of the extended PX4 firmware and acts as a manager between the manual and autonomous marine navigation modes. Following the conventional structure of PX4 modules⁴, which is not discussed here for brevity, it is implemented as a class inheriting the following standard PX4 base classes:

- *ModuleBase*, which provides the core module functionalities;
- *ModuleParams*, which enables the definition and access of control parameters;
- *ScheduledWorkItem*, which schedules the execution of the module main function `run(·)` at a frequency of 50 Hz.

All the information required for the module operation is exchanged through uORB⁵ topics.

The state diagram of the module is shown in Fig. 2a. When the system is started, the marine navigation module remains in a waiting state until a `vehicle_control_mode` uORB message signals the need for a navigation mode switch. The transition can be manually triggered by the pilot to enter or exit the marine manual navigation mode, or automatically requested by the mission module when a marine segment is required during mission execution, activating the autonomous marine navigation mode. During autonomous navigation, the pilot can always retake control—either aerial or marine manual navigation—through a safety switch. Otherwise, autonomous navigation continues until the marine path is completed.

In this work, the manual navigation mode is primarily intended as a safety mechanism that allows the operator to retake control of the UAAV during autonomous missions and is therefore designed for limited use.

In manual navigation mode, the custom module reads the inputs from the radio controller (RC) and directly converts them into pulse-width modulation (PWM) signals for thruster control. In autonomous navigation mode, path planning and path-following algorithms are executed to compute the thrust and torque commands to be applied to the UAAV, which are transformed into PWM signals for thruster actuation through the appropriate control allocation matrix. The adopted control strategies for both modes are detailed in Section IV.

B. Navigator module

The PX4 navigator module is responsible for autonomous flight modes and mission execution. The proposed customization enables the integration of marine segments within the mission execution pipeline. The main modifications were applied to the following files: `mission_base.h`, `mission_base.cpp`, `mission.h`, `mission.cpp`, and `mission_block.cpp`. Through these changes, the behavior described by the state diagram managing marine paths in Fig. 2b, is implemented.

Mission waypoints, selected through QGC, are classified as either aerial or marine based on a user-defined altitude

⁴<https://docs.px4.io/main/en/concept/architecture>

⁵<https://docs.px4.io/main/en/middleware/uorb>

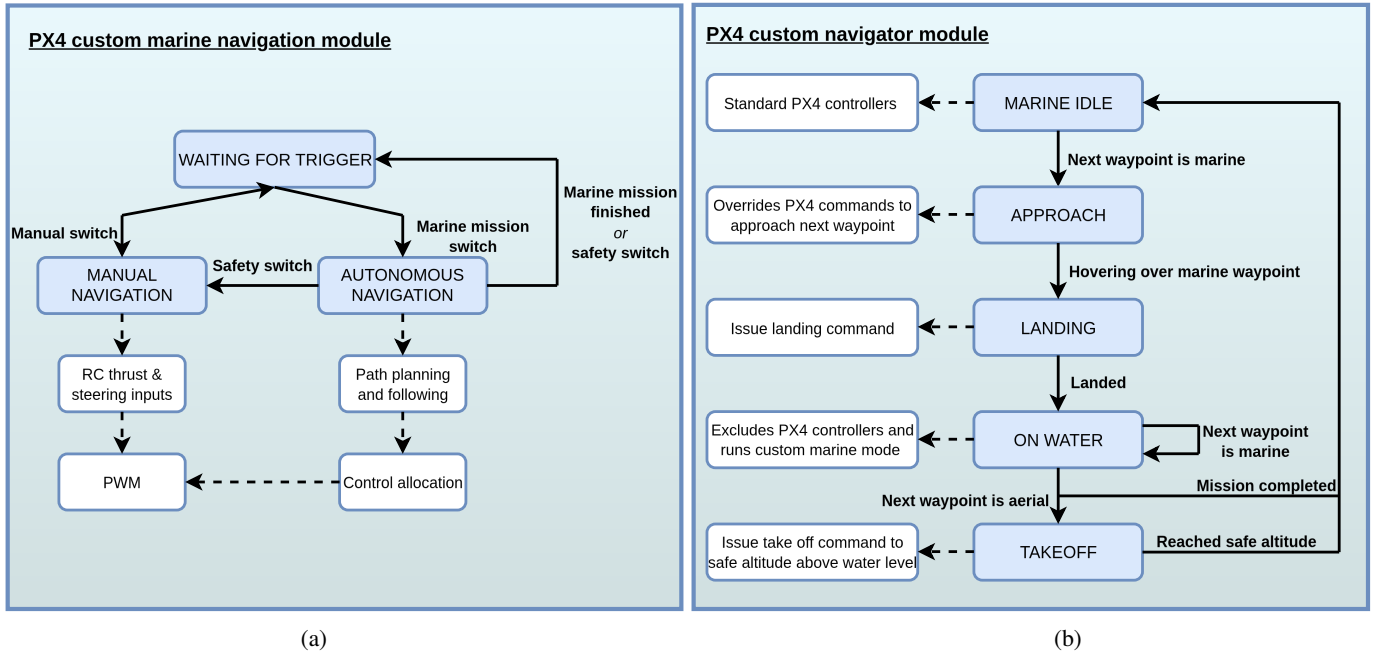


Fig. 2: [FR: Perché da MANUAL NAVIGATION a WAITING FOR TRIGGER c'è una freccia doppia?][AC: perché puoi fare lo switch anche al contrario](a) State diagram of the marine navigation module logic. (b) State diagram of the customized Navigator module handling marine mission branches. States are represented by blue boxes, and actions by white boxes; continuous lines denote state transitions, while dashed lines represent actions performed within each state.

threshold; waypoints below this altitude are considered marine. When an aerial waypoint is processed by the navigator module, the marine navigation logic remains idle and the standard PX4 controllers manage flight execution. Conversely, when a marine waypoint is received, the navigation logic transitions to an approach phase and temporarily overrides the PX4 controllers to guide the vehicle toward the destination waypoint while maintaining the previous altitude.

Once the waypoint is reached, a landing command is issued and the UAAV performs a water landing. During this phase, the logic ensures that the vehicle is oriented toward the direction of the subsequent waypoint in order to avoid unnecessary maneuvers on the water surface. If the following waypoint is also classified as marine, the navigation mode switches to the autonomous marine mode described in Section III-A and remains active until a new aerial waypoint is received or the pilot retakes manual control, returning the state to idle (this transition may occur from any state).

If the next waypoint is aerial, a takeoff command is issued and the UAAV climbs to a user-defined safe altitude above sea level before proceeding toward the waypoint, after which the logic returns to the marine idle state. The transition to the idle state also occurs when the mission is completed and no additional aerial or marine waypoints remain.

C. QGroundControl interface

The QGroundControl customizations introduced in this work⁶, made publicly available to the community⁷, ensure that these core functionalities, described in Section II-A are preserved.

The first customization enables the selection of the newly introduced flight modes directly from the navigation mode menu, as shown in Fig. 3: *PRISMA Auto Marine* for autonomous marine navigation and *PRISMA Manual TS* for manual marine navigation. This functionality is implemented on the PX4 side rather than within QGC itself, by assigning valid `nav_state` identifiers⁸ to the navigation modes and ensuring that the corresponding mode names are correctly communicated to QGroundControl⁹.

The second customization allows users to quickly distinguish between aerial and marine waypoints. When QGroundControl is launched and a mission is uploaded, waypoint altitudes are evaluated and, if a value falls below the threshold defined in Section III-B, the waypoint color is changed to blue, as illustrated in Fig. 4.

IV. METHODOLOGY

Before discussing the mathematical details of how the marine navigation control modes are implemented, it is necessary

⁶The custom version presented in this work is based on the release v4.3.0

⁷The link to the repository will be disclosed after acceptance. [FR: Immagino sia diverso da quello di prima perché coinvolge QGroundControl, ma comunque manca.]

⁸https://docs.px4.io/main/en/msg_docs/VehicleStatus

⁹Handled through the file `PX4-Autopilot/src/lib/modes/ui.hpp`

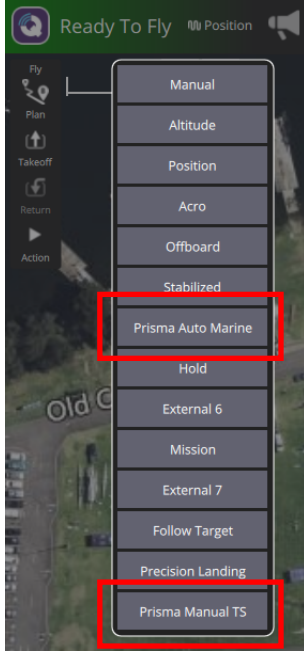


Fig. 3: QGC navigation mode selection showing the marine modes *PRISMA Auto Marine* and *PRISMA Manual TS*.

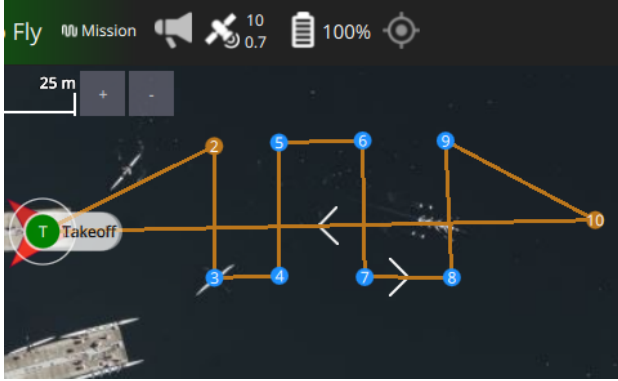


Fig. 4: QGC mission plan with aerial and marine (in blue) waypoints.

to state some preliminary assumptions.

A. UAAV specification

The UAAV reference platform considered in this work is illustrated in Fig. 1. The vehicle consists of a quadrotor equipped with two buoyant hulls and two fixed, non-steerable marine propellers for surface navigation. A dedicated payload bay is located on the upper structure, allowing the integration of mission-specific equipment and sensing tools.

B. Control allocation for marine vessels

In low-speed applications, a three-degrees-of-freedom model is sufficient to describe the UAAV motion, accounting for surge (motion along the x -axis of the body frame), sway (motion along the y -axis of the body frame), and yaw (rotation about the z -axis of the body frame denoted by the angle

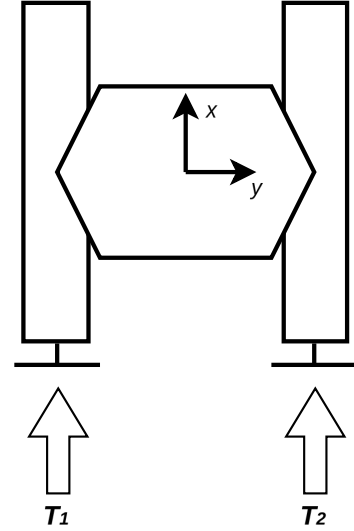


Fig. 5: Simplified UAAV scheme showing the body reference frame and the thrust generated by the marine propellers.

$\psi \in [-\pi, \pi)$), leading to the generalized force vector $\tau = [f_x, f_y, n]^T \in \mathbb{R}^3$, where $f_x, f_y \in \mathbb{R}$ denote the forces applied along the x - and y -axis of the body frame, respectively, and $n \in \mathbb{R}$ denotes the torque about the z -axis of the body frame. In [16], it is shown that, in the general case of a vessel equipped with $k > 0$ marine propellers—either steerable, allowing the thrust direction to be varied, or fixed—each generating a thrust $T_i > 0$, with $i = 1, \dots, k$, and located at position $p_i = [\ell_{i,x}, \ell_{i,y}]^T \in \mathbb{R}^2$, the generalized forces can be expressed as

$$\tau = \begin{bmatrix} \cos \alpha_1 & \dots & \cos \alpha_k \\ \sin \alpha_1 & \dots & \sin \alpha_k \\ \ell_1 & \dots & \ell_k \end{bmatrix} T, \quad (1)$$

where $\alpha_i \in [-\pi, \pi)$ is the orientation of the propeller p_i , $T = [T_1, \dots, T_k]^T \in \mathbb{R}^k$, and $\ell_i = \ell_{i,x} \sin \alpha_i - \ell_{i,y} \cos \alpha_i$.

As shown in Fig. 5, where the configuration of the reference UAAV platform is illustrated, the vessel layout includes only two non-steerable marine propellers. This simplifies (1) into

$$\bar{\tau} = \begin{bmatrix} 1 & 1 \\ -\ell_{1,y} & -\ell_{2,y} \end{bmatrix} T, \quad (2)$$

with $\bar{\tau} = [f_x, n]^T \in \mathbb{R}^2$.

C. Manual navigation control

The manual navigation control is implemented as a feed-forward strategy that modulates the thrust generated by each propeller, providing the pilot with the perception of independent control over total thrust and a virtual rudder. This design enables precise maneuvering, including pivoting without requiring forward motion.

The controller reads the radio control (RC) inputs through the uORB topic `manual_control_setpoint`, specifically the `throttle` and `roll` fields of the received message, interpreting them as total thrust t and steering s values,

respectively, with $t, s \in [-1, 1]$. In particular, $s \in [-1, 0]$ corresponds to left steering, whereas $s \in [0, 1]$ corresponds to right steering. The thrust commands to be generated by each propeller are then computed as

$$\begin{aligned} T_1 &= \begin{cases} t & \text{if } s \geq 0 \\ t(1+2s) & \text{if } s < 0 \end{cases}, \\ T_2 &= \begin{cases} t(1-2s) & \text{if } s > 0 \\ t & \text{if } s \leq 0 \end{cases}. \end{aligned} \quad (3)$$

Note that $T_1, T_2 \in [-1, 1]$, as they represent normalized thrust values for the propellers.

D. Autonomous navigation control

The autonomous navigation controller adopts a path-following strategy for underactuated marine vehicles based on the hand-position framework and input-output feedback linearization presented in [17]. The controlled output is a virtual point $h = [\xi_1, \xi_2]^T \in \mathbb{R}^2$, referred to as the hand-position point, located at a fixed distance l ahead of the UAAV center of mass $p = [x, y]^T \in \mathbb{R}^2$ along the x axis of the body frame

$$\begin{aligned} \xi_1 &= x + l \cos \psi, \\ \xi_2 &= y + l \sin \psi. \end{aligned} \quad (4)$$

The choice of controlling the hand-position point introduces an additional controllable output: a surge actuation applied at p generates a surge motion of h , while a yaw actuation induces a lateral motion of h proportional to the distance l . This property enables the application of input-output feedback linearization to underactuated marine platforms.

By applying the coordinate transformation and input mapping described in [17], and denoting by $\xi_3, \xi_4 \in \mathbb{R}$ the relative velocity components of h , by $V_x, V_y \in \mathbb{R}$ the unknown ocean current components expressed in the world frame, and by $u_1, u_2 \in \mathbb{R}$ the virtual control inputs, the external dynamics can be written as

$$\begin{aligned} \dot{\xi}_1 &= \xi_3 + V_x, \\ \dot{\xi}_2 &= \xi_4 + V_y, \\ \dot{\xi}_3 &= u_1, \\ \dot{\xi}_4 &= u_2. \end{aligned} \quad (5)$$

The virtual inputs are related to the generalized force vector $\bar{\tau}$ through the feedback-linearization mapping

$$\bar{\tau} = \begin{bmatrix} \cos \psi & -l \sin \psi \\ \sin \psi & l \cos \psi \end{bmatrix}^{-1} \begin{bmatrix} u_1 - F_{\xi,1}(\cdot) \\ u_2 - F_{\xi,2}(\cdot) \end{bmatrix}, \quad (6)$$

where $F_{\xi,1}(\cdot), F_{\xi,2}(\cdot) \in \mathbb{R}$ collect the nonlinear drift dynamics of the hand-point acceleration.

The virtual inputs u_1 and u_2 are designed for a generic C^2 path $\gamma(s) = (x(s), y(s))$, where $s \geq 0$ denotes the arc-length parameter of the path, and include feedforward terms accounting for the local curvature of the path [17]. In this work, the marine trajectory is defined as a sequence of straight-line segments connecting consecutive waypoints, resulting in a path with zero curvature and constant tangent direction.



Fig. 6: Example of simulation with UAAV mockup landed on water surface

Under these assumptions, curvature-dependent feedforward terms vanish, leading to the following simplified choice of the virtual inputs

$$\begin{aligned} u_1 &= -k_{vx}(\xi_3 - \dot{x}(s)) - k_{px}(\xi_1 - x(s)) - k_{Ix}I_x, \\ u_2 &= -k_{vy}(\xi_4 - \dot{y}(s)) - k_{py}(\xi_2 - y(s)) - k_{Iy}I_y, \end{aligned} \quad (7)$$

where $\dot{x}(s) = \dot{s}t_x$ and $\dot{y}(s) = \dot{s}t_y$, with $[t_x, t_y]$ denoting the constant unit tangent of the active path segment, and the constants $k_{vx}, k_{vy}, k_{px}, k_{py}, k_{Ix}, k_{Iy} \in \mathbb{R}$ as the control gains [FR: Se queste sono positive puoi mettere > 0 o $\mathbb{R}^+$ in luogo di $\mathbb{R}$]. The integral terms are defined as

$$\begin{aligned} I_x &= \int_0^t (\xi_1(\tau) - x(s(\tau))) d\tau, \\ I_y &= \int_0^t (\xi_2(\tau) - y(s(\tau))) d\tau, \end{aligned} \quad (8)$$

and provide steady-state rejection of constant environmental disturbances such as ocean currents.

Once the virtual inputs are computed, the generalized forces $\bar{\tau}$ are obtained from (6) and subsequently mapped to the individual propeller commands through the inversion of (2).

To account for steady ocean currents in the implementation, the hand-point velocities are evaluated using water-relative kinematics. Specifically, the measured ground velocity is corrected by subtracting an estimate of the current velocity $\hat{v}_c \in \mathbb{R}^2$

$$\begin{bmatrix} \dot{x}_r \\ \dot{y}_r \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} - \hat{v}_c. \quad (9)$$

Accordingly, the relative velocity components of the hand-position point are computed as

$$\begin{aligned} \xi_3 &= \dot{x}_r - l \sin \psi \dot{\psi}, \\ \xi_4 &= \dot{y}_r + l \cos \psi \dot{\psi}. \end{aligned} \quad (10)$$

V. SIMULATIONS

A. Simulation set up

The simulations were performed on a standard personal computer equipped with an *Intel Core i7* (11th generation) processor and 32 GB of RAM, running *Ubuntu 22.04* as operative system.

The marine environment was simulated using the *Virtual RobotX* (VRX) simulation environment [18], executed on *Gazebo Garden v7.9.0* (see Fig. 6), with modifications to account for damping effects in drone–water interactions [19]. Without these adjustments, the buoyant force would behave as an elastic force, causing the UAAV to exhibit continuous oscillatory motion upon contact with the water surface. The UAAV dynamics were simulated using the *PX4 Autopilot* firmware operating in software-in-the-loop (SITL) mode.

Two different case studies are presented in which the UAAV executes the same hybrid aerial mission: taking off from land, reaching a designated location over water, landing, and navigating along a boustrophedon path defined by six waypoints—selected through QGC—with altitude values below the threshold specified in Section III-B, thereby simulating a search mission on the sea surface. Once the path is completed, the drone returns to the starting point. The case studies differ in environmental conditions: in the first case, the sea is calm, while in the second, a wave field is present, modeled as a superposition of Gerstner waves [20].

The analysis focuses on the marine portion of the mission, evaluating performance and the UAAV’s ability to remain on track, thereby validating the proposed controllers.

The produced video¹⁰ shows the complete execution of the first case study mission, including a safety recovery with pilot-initiated manual takeover.

B. Case study 1)

Figure 7 shows the executed trajectory along with the corresponding off-track error norm, defined as the distance between the actual UAAV position and the position it should ideally occupy to follow the prescribed path over the entire duration of the marine phase of the mission. During this phase, the UAAV navigates between eight waypoints defining a boustrophedon pattern. Along the straight segments, the tracking error remains within acceptable ranges [21], indicating stable path-following performance, demonstrating the effectiveness of the controller in maintaining alignment with the reference path. Conversely, during turning maneuvers the vessel temporarily deviates from the desired trajectory, as reflected by the peaks in Fig. 7b, due to inertial and hydrodynamic effects that limit rapid reorientation. Nevertheless, precise tracking is rapidly re-established once the turn is completed, which is reflected by the localized spikes in the error norm occurring only at the transition points between consecutive segments.

This behavior can be further interpreted by examining Fig. 8, where the forces and torques generated by the autonomous navigation controller, along with the corresponding

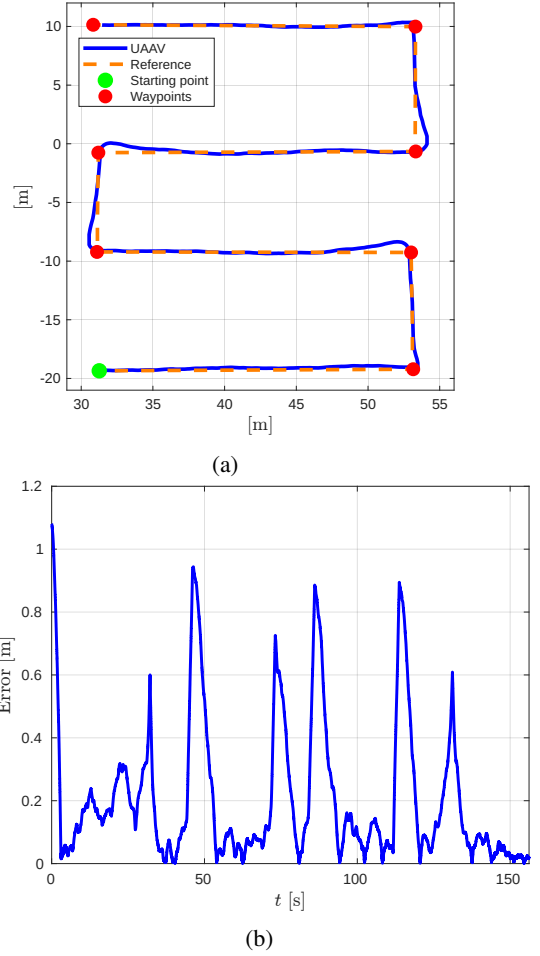


Fig. 7: *Case study 1*): (a) comparison between executed marine path and reference; (b) off-track error norm.

propeller control inputs, are shown. As the UAAV approaches the end of each marine segment, the controller progressively reduces forward thrust to ensure an accurate convergence toward the waypoint. At the initiation of motion toward the subsequent waypoint, a stronger torque is commanded to promptly align the vessel with the new heading direction. This sequence of deceleration, reorientation, and re-acceleration explains the transient tracking error during turns and the subsequent recovery of nominal path-following performance.

C. Case study 2)

In this scenario, the total wave field is composed of two incident component waves with a period of 4 s and an angular separation of 0.4 rad, with a median wave amplitude of 1.5 m. In Fig. 9, the executed path and the corresponding reference trajectory are shown. The overall performance remains acceptable; however, a higher trajectory error is observed, including along the linear segments, as can be seen from the error plot when compared with that of *case study 1*).

The presence of the wave field also increases the instability of the UAAV during pivoting maneuvers, further amplifying deviations from the prescribed path at the turning points. In

¹⁰<https://youtu.be/DNcr2hE-144>

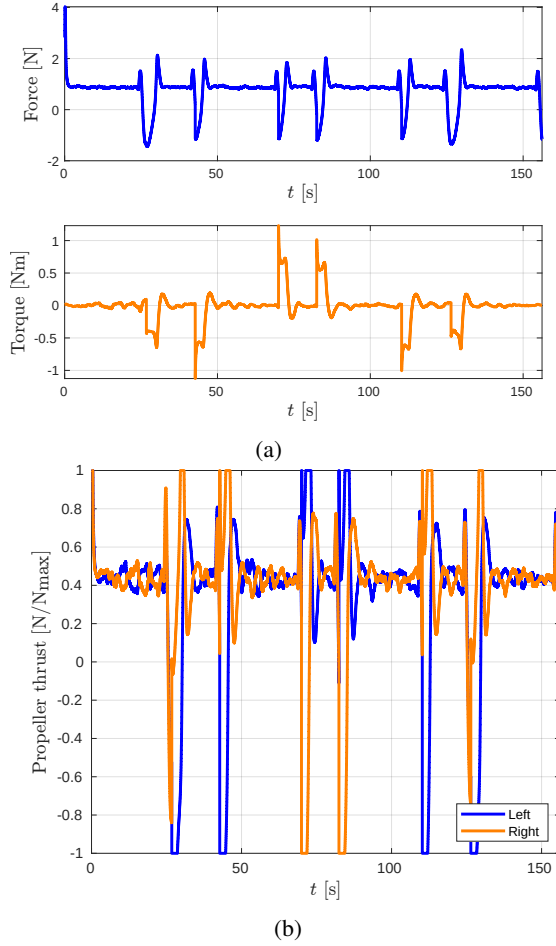


Fig. 8: *Case study 1*: (a) computed force and torque controls; (b) left and right normalized propeller control inputs.

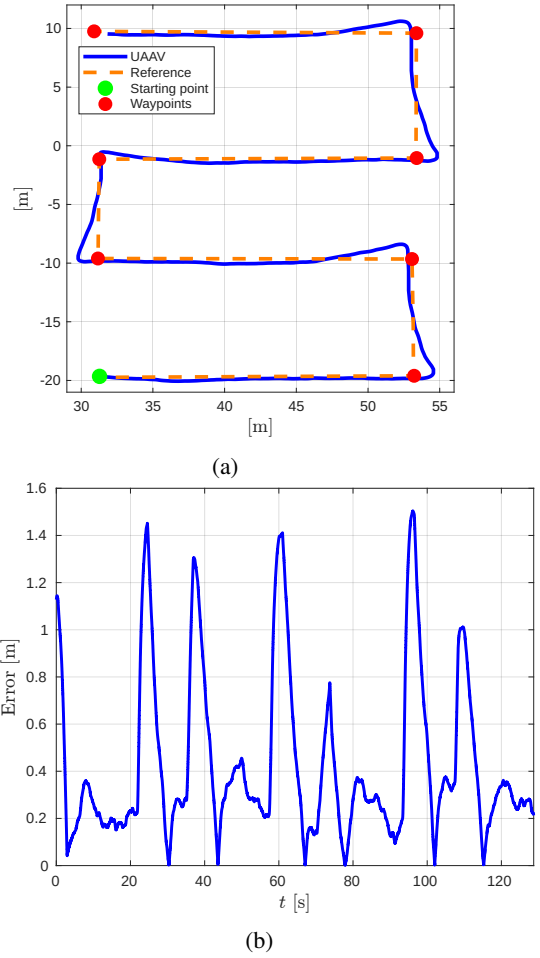


Fig. 9: *Case study 2*: (a) comparison between executed marine path and reference; (b) off-track error norm.

Fig. 10, the computed force and torque commands together with the propeller control inputs are displayed. The wave-induced disturbances affect these quantities, which exhibit a more pronounced oscillatory behavior compared to the previous case.

VI. DISCUSSION

The results presented in Section V require a careful analysis highlighting both the strengths and the limitations of the proposed extended firmware and of the autonomous marine navigation control.

The modified firmware demonstrates the capability to seamlessly manage hybrid missions and marine navigation, handling their execution while enforcing the necessary safety mechanisms. In particular, it allows the operator to rapidly switch between navigation modes, ensuring that control can be promptly regained in emergency situations. However, when hybrid missions are planned, no mechanism is currently available to verify mission feasibility prior to execution—such as checks based on battery state—or to adapt the mission online, for example by issuing an automatic return-to-home command when the energy level becomes insufficient. Furthermore, the

range of marine tasks supported by the navigation framework remains limited, and future developments should consider extending the mission directives beyond pure navigation, enabling more complex surface-operation activities.

From the perspective of the control algorithm, the simulation results indicate the capability in maintaining stable navigation performance, particularly under non-ideal environmental conditions, while keeping propeller control inputs within acceptable operational bounds. Nevertheless, improvements can be pursued in the path planning stage, where the adoption of continuous trajectory generation could provide smoother transitions between waypoints and reduce the transient deviations observed during maneuvering.

At last, the manual navigation control mode could also be improved, since its purely feedforward structure requires skilled operators for effective maneuvering. Future developments should therefore aim at introducing assistance mechanisms that reduce pilot workload and allow non-expert users to reliably control the UAAV in water-surface operations.

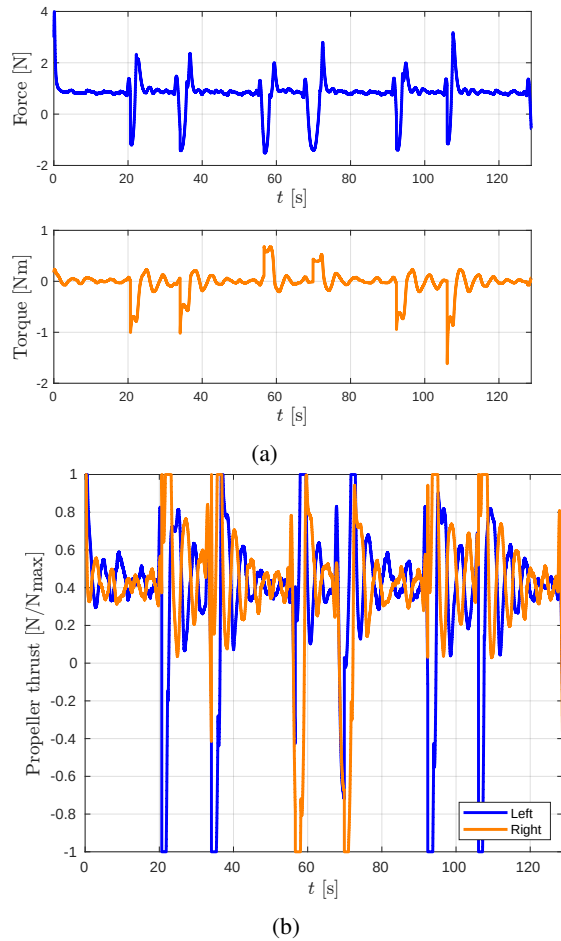


Fig. 10: Case study 2): (a) computed force and torque controls; (b) left and right normalized propeller control inputs.

VII. CONCLUSION AND FUTURE WORK

This paper proposes a new version of the PX4 firmware for UAAVs operating in hybrid aerial–marine autonomous missions. The primary objective is to introduce a unified and integrated hybrid control framework for such operations, ensuring safe execution for the pilot, the vehicle, and the surrounding environment. The introduced marine control modes enable both manual and autonomous navigation while preserving the standard user interface, thereby maintaining ease of use. Simulation results confirm the effectiveness of the extended firmware and highlight its potential for future developments.

This work can be further extended by enhancing the firmware capabilities and safety mechanisms through mission feasibility checks, online replanning, and the inclusion of additional specifications for hybrid mission execution. Furthermore, the marine control framework can be expanded by improving path planning strategies and by designing a more accessible manual control mode.

ACKNOWLEDGMENTS

The research leading to these results has been partially supported by *Centro Nazionale 5 (CN5) - National Biodiversity Future Center (NBFC) - SPOKE 1* (Codice progetto MUR: CN0000033 CUP UNINA: E63C22000990007). This research was also partially supported by the INVITALIA Project NEMESI. CUP C67G22000420008. The authors are solely responsible for its content.

The rendering of Fig. 1 has been obtained using AI tool such as OpenAI ChatGPT (v. 5.2) with simple prompt commands.

REFERENCES

- [1] R. Bogue, "The role of robots in environmental monitoring", *Industrial Robot: the international journal of robotics research and application*, vol. 50, n. 3, 2023, pp. 369–375
- [2] D. Ventura, A. Bonifazi, "Unmanned Aerial Systems (UASs) for Environmental Monitoring: A Review with Applications in Coastal", *Aerial Robots: Aerodynamics, Control and Applications*, 2017
- [3] PX4 Autopilot, available: <https://px4.io/>
- [4] ArduPilot, available: <https://ardupilot.org/>
- [5] iNav, available: <https://github.com/iNavFlight/inav>
- [6] Betaflight, available: <https://www.betaflight.com/>
- [7] D. Ventura, A. Bonifazi, M. F. Gravina, A. Belluscio, G. Ardizzone, "Mapping and Classification of Ecologically Sensitive Marine Habitats Using Unmanned Aerial Vehicle (UAV) Imagery and Object-Based Image Analysis (OBIA)", *Remote Sensing*, vol. 10, n. 9, 2018
- [8] J. Lindsay, J. Ross, M. L. Seto, E. Gregson, A. Moore, J. Patel, R. Bauer, "Collaboration of Heterogeneous Marine Robots Toward Multidomain Sensing and Situational Awareness on Partially Submerged Targets", *IEEE Journal of Oceanic Engineering*, vol. 47, n. 4, 2022, pp. 880–894
- [9] D. F. Carlson, S. Akbulut, J. F. Rasmussen, C. S. Hestbech, M. H. Andersen, C. Melvad, "Compact and modular autonomous surface vehicle for water research: The Naval Operating Research Drone Assessing Climate Change (NORDACC)", *Hardware X*, vol. 15, 2023
- [10] P. Gogendeau, S. Bonhommeau, H. Fourati, M. Julien, M. Contini, T. Chevrier, A. E. Nieblas, S. Bernard, "An autonomous surface vehicle for acoustic tracking, bathymetric and photogrammetric surveys", *Ocean Engineering*, vol. 331, 2021
- [11] P. Gogendeau, S. Bonhommeau, H. Fourati, M. Julien, M. Contini, T. Chevrier, A. E. Nieblas, S. Bernard, "An autonomous surface vehicle for acoustic tracking, bathymetric and photogrammetric surveys", *Ocean Engineering*, vol. 331, 2025
- [12] X. Xiao, J. Dufek, T. Woodbury, R. Murphy, "UAV assisted USV visual navigation for marine mass casualty incident response", *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017
- [13] T. Huang, Z. Chen, W. Gao, Z. Xue, Y. Liu, "USV-UAV Cooperative Trajectory Planning Algorithm with Hull Dynamic Constraints", *Sensors*, vol. 23, n. 4, 2023
- [14] D. Manoharan, C. Gajendran, M. K. Padmanabhan, S. Vignesh, S. Rajesh, G. D. Bhuvaneshwaran, "Design and development of autonomous amphibious unmanned aerial vehicle for in situ water quality assessment and water sampling", *Journal of Unmanned Vehicle Systems*, vol. 9, n. 3, 2021, pp. 182–204
- [15] QGroundControl, available <https://qgroundcontrol.com/>
- [16] T. A. Johansen, T. P. Fuglseth, P. Tøndel, T. I. Fossen, "Optimal constrained control allocation in marine surface vessels with rudders", *Control engineering practice*, vol. 16, n. 4, 2008, pp. 457–464
- [17] C. Paliotta, E. Lefeber, K. Y. Pettersen, J. Pinto, M. Costa and J. T. de Figueiredo Borges de Sousa, "Trajectory Tracking and Path Following for Underactuated Marine Vehicles", *IEEE Transactions on Control Systems Technology*, vol. 27, n. 4, 2019, pp. 1423–1437
- [18] B. Bingham, C. Agüero, M. McCarrin, J. Klamo, J. Malia, K. Allen, T. Lum, M. Rawson, R. Waqar, "Toward Maritime Robotic Simulation in Gazebo", *MTS/IEEE OCEANS Conference*, 2019

- [19] M. B. Kramer, J. Andersen, S. Thomas, F. B. Bendixen, H. Bingham, R. Read, N. Holk, E. Ransley, S. Brown, Y. H. Yu, T. T. Tran, J. Davidson, C. Horvath, C. E. Janson, K. Nielsen, m C. Eskilsson, "Highly Accurate Experimental Heave Decay Tests with a Floating Sphere: A Public Benchmark Dataset for Model Validation of Fluid–Structure Interaction", *Energies*, vol. 14, n. 2, 2021
- [20] J. Tessendorf, "Simulating Ocean Water", *Simulating nature: realistic and interactive techniques*, vol. 1, 2001
- [21] J. Ghommam, S. El Ferik, M. Saad, "Robust adaptive path-following control of underactuated marine vessel with off-track error constraint", *International Journal of Systems Science*, vol. 49, n.7, 2018, pp. 1540-1558