Journal of Intelligent & Robotic Systems (2024) 110:59
https://doi.org/10.1007/s10846-024-02095-2

REGULAR PAPER

ROV-Based Autonomous Maneuvering for Ship Hull Inspection with Coverage Monitoring

Alexandre Cardaillac[1] · **Roger Skjetne**[1] · **Martin Ludvigsen**[1]

Received: 4 August 2023 / Accepted: 28 March 2024

Abstract
Hull inspection is an important task to ensure sustainability of ships. To overcome the challenges of hull structure inspection in an underwater environment in an efficient way, an autonomous system for hull inspection has to be developed. In this paper, a new approach to underwater ship hull inspection is proposed. It aims at developing the basis for an end-to-end autonomous solution. The real-time aspect is an important part of this work, as it allows the operators and inspectors to receive feedback about the inspection as it happens. A reference mission plan is generated and adapted online based on the inspection findings. This is done through the processing of a multibeam forward looking sonar to estimate the pose of the hull relative to the drone. An inspection map is incrementally built in a novel way, incorporating uncertainty estimates to better represent the inspection state, quality, and observation confidence. The proposed methods are experimentally tested in real-time on real ships and demonstrate the applicability to quickly understand what has been done during the inspection.

Keywords Computer vision · Underwater navigation · Underwater robotics · Visual inspection · Sonar processing · Robotic perception · Coverage mapping

1 Introduction

Inspections are essential for good maintenance procedures, to establish the integrity and status of the ship hulls. Robotic documentation methods can improve both efficiency and safety for maritime operations. Shipping activities can range from transport of goods, extraction and production of resources, and tourism and recreation. The structures involved are all affected by the dynamic and rough oceanic environmental conditions and structural damages can significantly impact the operations. The presence of fouling has an impact on the fuel consumption; it generates frictional resistance [1]. Other defects can appear such as cracks and corrosion. The necessary repairs cause loss in time and money [2]. Recurring inspections are thus required by international regulations and often reveal necessary maintenance needs. Traditionally, on ships and floating vessels, these tasks are carried out through dry-docking, which can immobilize the vehicle for a long period of time. Prior inspection and maintenance operations are commonly performed by trained divers while the ship is still in water, but docked [3]. This is a dangerous, challenging, and non-exhaustive activity. Recently, inspectors have started to take advantage of modern technologies to reduce cost and time. By making use of Remotely Operated underwater Vehicles (ROVs) [4], they can operate efficiently and quickly to record evidence of the visual inspection without endangering human life. ROVs can be efficient but require skilled operators and constant attention. The quality of the documentation often varies and relies on the performance of the human in the loop.

An autonomous system can improve the quality of the inspection by providing accurate and repeatable results by maneuvering for the underwater vehicle precisely relative to the vessel. Data interpretation for ship hull inspection can also be automated. A major progress towards this direction is presented in [5], presenting a case with inspection missions for anti-terrorism and force protection, which consists of sonar based investigations using guidance and control system

✉ Alexandre Cardaillac
alexandre.cardaillac@ntnu.no

Roger Skjetne
roger.skjetne@ntnu.no

Martin Ludvigsen
martin.ludvigsen@ntnu.no

[1] Department of Marine Technology in the Faculty of Engineering, Norwegian University of Science and Technology (NTNU), Trondheim, Norway

Published online: 11 April 2024

relative to the ship hull and with global synchronisation using a GPS. The inspection vehicle follows horizontal and vertical slices pointing a Doppler Velocity Log (DVL) towards the hull to keep the vehicle normal to the hull's surface. The onboard sensors are actuated to adapt to the orientation of the hull sections during the inspection mission.

In [6], the inspection vehicle is capable of drift-free self-localisation relative to the ship hull using only onboard sensors. It relies mostly on sonar scans from an imaging sonar; however, when the vehicle is approaching more complex region of the ship hull, the acoustic lens needs to be changed in order to transform the imaging sonar into a profiling sonar. This work is extended in [7] where planning routines are developed for efficient visual coverage of the complex region linked to a waypoint following method for navigation. The work presented in [8] also implements waypoints tracking to follow a predefined vertical lawn-mower pattern. During the operation, user inputs from an onshore joystick are incorporated to adapt the heading of the vehicle and its distance to the hull enabling a semi-autonomous mode of operation.

All the presented previous works require prior knowledge or prior processing in order to efficiently perform the inspection. Many of them depend on data post-processing for the interpretation of the results which would require the operation to be finished before having meaningful results. Also, not all solutions can adapt autonomously to the shape of the hull while navigating, making human intervention necessary. A small, low-cost, and energy-efficient vehicle should be used to provide cost-effective logistics and access to confined volumes while adhering to class society regulations.

When considering the side of a ship, the hull can be considered as locally flat, and represented as a set of planes. Navigation along it can be viewed as a wall following problem. Wall detection in an underwater environment is often resolved using range measurements from an acoustic sensor. In [9], a Mechanically Scanned Imaging Sonar (MSIS) is deployed. After applying a standard Hough transform combined with an improved sonar model to detect line features, a wall is tracked by an Extended Kalman Filter (EKF). A method to detect the wall and estimate the pose of the vehicle is proposed using the Random Sample Consensus (RANSAC) [10] using measurements received from a multibeam imaging sonar [11]. Recently, an approach using reinforcement learning was studied in [12], where a set of ranging sensor is used and efficiently manipulated to allow an underwater vehicle to navigate along the wall. However, the proposed methods often lack robustness and cannot adapt well to shape changes, especially in the presence of more objects than the wall or acoustic noise.

Generally, for guidance and path following, line-of-sight (LOS) guidance laws [13, Chapter 12] are employed. It is mainly used for paths composed of straight lines. For more complex parameterised path, guidance laws are often developed in a Serret-Frenet framework [13, Chapter 12]. In [14], a LOS-based guidance law is employed to inspect aquaculture net pens and combined with the beams from a DVL to approximate the geometry of the local region facing the ROV as a plane. Path tracking and following for an underactuated inspection class ROV is studied in [15]. Inspection of cylindrical structures are considered and the lookahead-based LOS guidance law was chosen to follow a path parameterised as a 3D spiral. In [16], profile following is performed using a sonar that construct a local representation of the environment and linear regression is utilised to estimate the local profile. Profile following is then achieved with behavior-based controllers. Although the mentioned solutions work well for their respective applications, they do not allow an advanced parameterisation of the speed along the path, and cannot follow the path with very high accuracy and including the constraints.

A fully autonomous hull inspection system should be aware of its inspection coverage and, therefore, be able to detect holes and gaps in the data set. Common approaches to this problem work directly with the original point cloud, [17] presented hole boundary detection using the method presented in [18] by projecting the 3D point cloud on the 2D plane Oxy. A growth function is then applied to extract the boundaries. In [19], the boundaries are extracted using 2D phase information by detecting phase jumps, and the holes are then repaired with an algorithm based on Structure From Motion (SFM) and point cloud registration. The method by [20] converts first the point cloud into a voxel grid based on the same methods as previously mentioned to detect boundaries. However, to detect height jumps and vertical away holes, thresholding operations are performed.

The work presented in this paper aims at developing the basis for an end-to-end autonomous solution for ship hull inspection by focusing on real-time navigation and control, and inspection monitoring using a small and low-cost ROV. Human intervention is only required for deployment, monitoring, and retrieval of the underwater vehicle. The navigation and control system consists of following a vertical lawnmower pattern generated from the main dimension of the ship hull provided as a priori user input. The approach is model free, and no Computer Aided Designs (CAD) or hull drawings are required. The vehicle adapts and autonomously updates the mission plan based on quality and progress of the mapping data. The inspection pattern is adapted in real-time based on measurements from a forward looking multibeam sonar. Processing of the sonar data results in orientation changes and translations of the desired vehicle path to maximise the visual coverage of the ship while keeping the underwater vehicle normal to the hull's surface. An inspection map is constructed online to keep track of the areas inspected. The map incorporates uncertainty measurements

and hole detection features to provide relevant and understandable information to the operator and inspector. The main contribution is a resilient navigation system adapting the mission of the underwater vehicle using observations from the onboard sonar and navigation sensors. This is made possible by using a new approach to the interpretation of the sonar measurements to not only update the current path but also to estimate the shape of the structure over time. Holes in the coverage map of the structures are detected through a proposed processing pipeline suited to this ship hull scenarios. Finally, the occupancy map is augmented with uncertainty estimations to quantify the inspection quality and confidence.

The proposed solution is extensively tested, including field trials with six different ships distributed in three different locations. The inspections are performed in harbors and shipyards on docked ships and taking into account the rules and regulations proposed by the class societies.

This article is based on previous work published by the authors to build a complete robotic solution for ship hull inspection. The path planner in [21] is augmented with a new component tailor-made for online adaptation of the path segments to allow an efficient navigation along the ship hull. The localisation method employs the framework in [22], setup to account for the disturbances that exist in a harbor. In this work, the path following method described in [23] was used for the transit part of the mission. These methods are combined with inspection pattern generation, guidance for inspection paths, generation of voxel coverage map and sonar based relative navigation to form an integrated solution.

To summarise, the following main contributions are proposed:

1. A maneuvering-based guidance strategy for precise inspection with online adaption and continuously updated constraints.
2. Robust line detection in sonar imagery with low-level image processing approach.
3. Online acoustic inspection map generation with self-monitoring functions and uncertainty estimation.
4. An integrated and functioning system for ship hull inspection, compatible with the international regulations.

2 Vehicle Setup

The proposed solution is developed using the X3 small ROV produced by the company Blueye Robotics[1] as shown in Fig. 1(a). The default integrated sensor payload consists of two Inertial Measurement Units (IMUs), a pressure sensor providing depth measurements, and a camera inside a glass dome with ∼48 degrees vertical Field Of View (FOV) and ∼77 degrees horizontal FOV. Three external sensors are connected to the vehicle's guest ports: a GPS on a stick for synchronisation with the global navigation frame, a forward-looking multibeam sonar (FL-MBS), and a DVL oriented towards the sea bottom to measure the speed over ground in the vehicle's frame. The FL-MBS has a configurable range and 130 degrees horizontal and 20 degrees vertical apertures, and it is pointed in the same direction as the camera with a slight vertical offset. The footprints of the camera and the forward-looking sonar are depicted in Fig. 1(b) with the Blueye ROV facing a ship hull. The acoustic beams from the DVL are not shown, but they are pointing down and towards the sea bottom. Also note that the tilt of the vehicle could have an impact on the inspection because of the sensor footprints but is very limited as long as the vehicle's tilt is not superior to the sonar vertical aperture. The vehicle is neutrally balanced and passively stable for roll and pitch with a large righting moment to avoid such situation.

The ROV is actuated in surge, sway, heave, and yaw. Processing of the localisation, guidance, and control algorithms are fully done onboard the vehicle, whereas the optical imagery and sonar data are processed on the operator's laptop connected to the surface unit via the umbilical and WiFi due to the limited computational capacity onboard the vehicle. The operator has the possibility to interact with the vehicle for safety reasons based on the online data feed and to provide high level input.

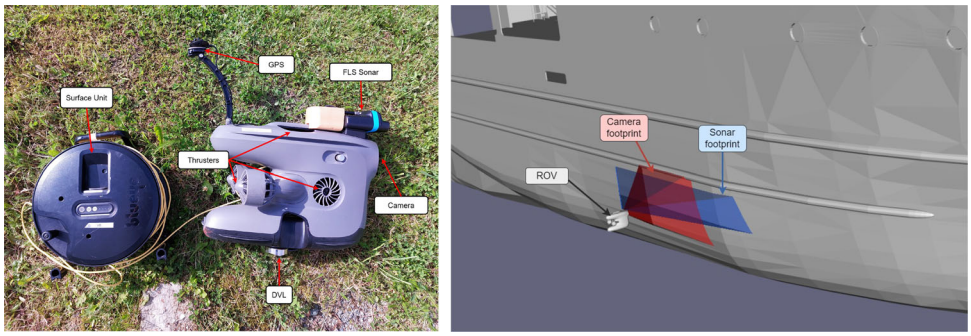
3 Mission Procedure and Manager

To autonomously perform a visual inspection of a submerged ship, the ROV must be able to move while facing the hull, which requires it to be fully actuated in a 4DOF configuration space. The operator starts the operation by deploying the drone nearby the target and start the inspection mission by sending the mission details to the ROV, such as a starting position nearby the hull and the area to be covered.

To initialise the state of the vehicle, the gravity direction is used to align the ROV navigation frame to the global frame when the ROV is assumed to be static. The initial yaw angle is estimated using the magnetometer. This is done once and away from the dock and any strong magnetic disturbances or metallic structures. Measurements from a GNSS are received while the vehicle is in the surface, allowing global positioning and yaw correction if needed. Satellite navigation is not available underwater, and the ROV returns to the surface to update the estimated vehicle pose when the standard deviation grows beyond a threshold value.

Once the initialization is done, the vehicle moves towards the vessel to the assigned starting position followed by the execution of the inspection mode. An inspection map is

[1] Blueye: https://www.blueyerobotics.com



(a) (b)

Fig. 1 (a) Blueye X3 ROV used for the experiments. (b) Representation of the field of view of the onboard camera and the forward looking sonar while the ROV is facing a ship hull

gradually built using the sonar, to keep track of the inspected areas. When the ROV has finished the inspection pattern, it returns to fill eventual detected gaps and uncertain areas in the inspection map.

As shown in Fig. 2, the embedded mission manager has control over all components of the autonomous system and handles their interactions. Although the operator should only interact with the mission manager, bridges are established with the other components for safety and debugging purposes.

Each module works at a different update rate based on the sensor and computing capacities. They are synchronized in each stack mechanism, where the system contains four stacks: control, guidance, navigation, and mapping. The update rate is controlled and constant for all stacks except for the localisation stack which runs at a free rate depending on the availability and rate of the navigation sensors. The system with the associated update rates are displayed in Fig. 3.

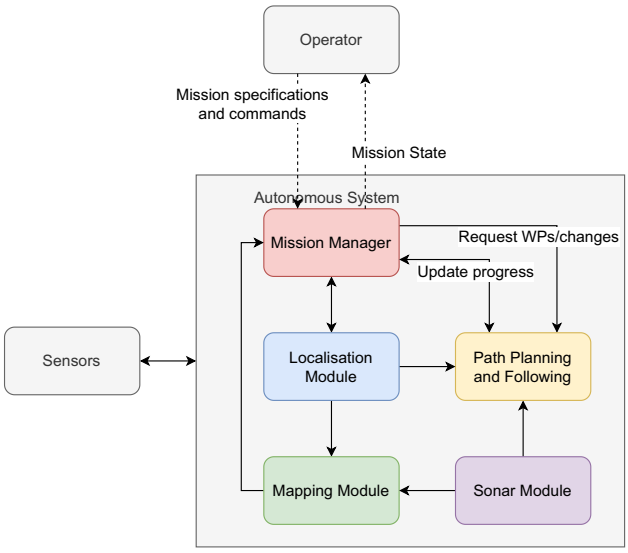


Fig. 2 System components and their interactions. The embedded mission manager has control over the other components

Figure 4 provides a schematic of the procedures and interactions between the navigation modules. The theory and implementation details are provided in the next sections.

4 Localisation Module

Commonly, underwater vehicles employ Kalman Filters or variations to perform multi-sensor fusion and estimate the localisation [24, 25]. However, their configurations are generally fixed and rigid.

In this article, the 6DOF pose of the vehicle is based on five sensors: IMU, Magnetometer, DVL, GPS, and Pressure sensor. They are listed with the associated rates and details in Table 1. To use and combine these sensors efficiently, the Modular and Robust Sensor-fusion (MaRS) framework [26] is employed and extended to work with underwater vehicles [22]. It is based on an error-state Extended Kalman Filter, which handles measurement outliers and outages as well as online extrinsic calibration of the sensors. The MaRS

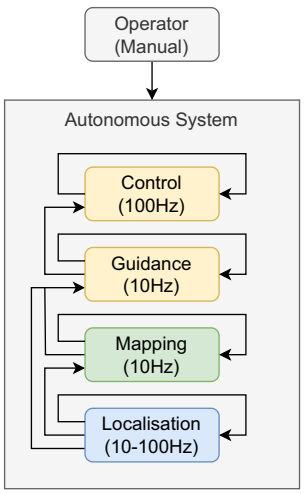


Fig. 3 System stacks, their interactions, and their approximate averaged rates

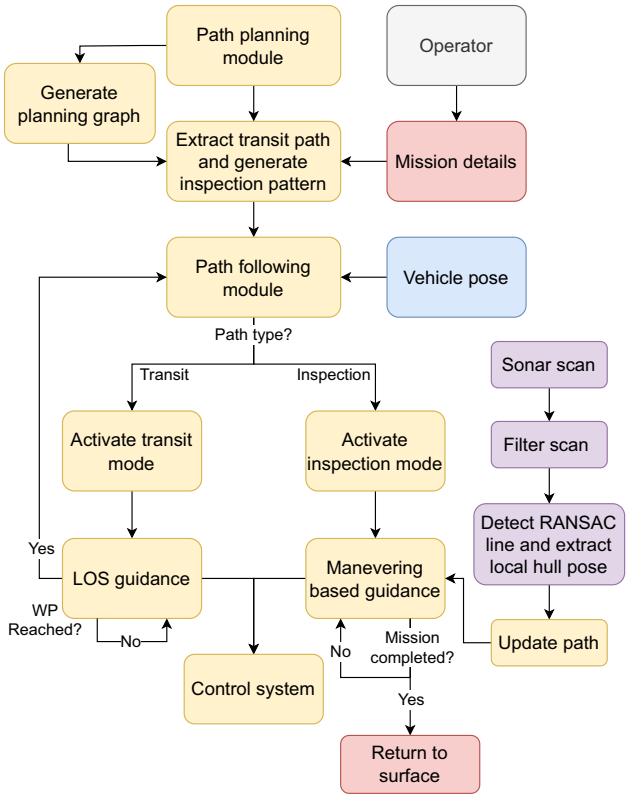


Fig. 4 The stateflow diagram representing the planning, guidance, and control procedures and interactions

framework estimates the core state

$$\mathrm{x}_{core} := \left[{}_N p_{NI}^\top,\ q_{NI}^\top,\ {}_N v_{NI}^\top,\ {}_I b_a^\top,\ {}_I b_\omega^\top \right]^\top \in \mathbb{R}^{16\times 1} \quad (1)$$

composed within the gravitationally-aligned navigation frame N of the position of the drone's IMU $p \in \mathbb{R}^3$, its orientation $q \in \mathbb{R}^4$, and velocity $v \in \mathbb{R}^3$. The biases $b_a \in \mathbb{R}^3$ and $b_\omega \in \mathbb{R}^3$ are also estimated for acceleration and angular velocities, respectively. The vehicle state is corrected using the following sensor measurement equations:

$$\mathrm{z}_{DVL} = R_{ID}^\top R_{NI}^\top {}_N v_{NI} + R_{ID}^\top \lfloor {}_I\breve{\omega}_{NI} - {}_I b_\omega \rfloor_\times {}_I p_{ID} \quad (2)$$

Table 1 Sensor list and details

Sensor	Rate (Hz)	Details
IMU	100.0	Acceleration readings
Magnetometer	100.0	Direction readings
DVL	5.0	Velocities and attitude
Pressure	45.0	Depth
GNSS	1.0	Global position

$$\mathrm{z}_{Pressure} = \|{}_N g\| \cdot \rho_{\text{water}} \cdot [0,0,1]\,({}_N p_{NI} + R_{NI}\, {}_I p_{IP}) \quad (3)$$

$$\mathrm{z}_{Pos} = {}_{G_0} p_{G_0 N} + R_{G_0 N}\,({}_N p_{NI} + R_{NI}\, {}_I p_{IG}) \quad (4)$$

$$\mathrm{z}_{Att} = R_{A_0 N} R_{NI} R_{IA}. \quad (5)$$

The calibration states ${}_N p_{ID}$, ${}_N p_{IP}$, and ${}_N p_{IG}$ describe respectively the translation between the IMU and the DVL, pressure, and position. The orientation offsets between the IMU and DVL and attitude sensor are described by R_{ID} and R_{IA}. Two constants are included, the norm of the gravitational acceleration $\|{}_N g\| = 9.81\text{m/s}^2$, and the density of the water $\rho_{\text{water}} = 997\text{kg/m}^3$. Finally, two additional reference frame are introduced, of the position sensor, G_0, and attitude sensor, A_0.

Probabilistic tests are performed on every sensor measurement received using a χ^2-test based on the prior sensor states and covariance. It is used within MaRS to detect and reject outliers like noisy or faulty measurements. The framework also provides the full state covariance matrix, which provides information of the navigation uncertainty.

Details on the implementation and performance of the method are provided in [22, 26].

5 Inspection Pattern Generation

Path planning is adaptive and done in multiple steps with minimum prior information. The first step is completed using the vessel length and draught. Here, the Parameterized Rapidly-exploring Random Graph (PRRG) method is employed [21], developed in previous work based on the Rapidly-exploring Random Tree (RRT) [27] for safe and dynamic navigation in multi-dimensional environments. The generated graph is designed to enable updates in real-time. It can be combined with a classical planner such as the Dijkstra algorithm [28] or a more specific planner such as D* Lite [29], which features embedded methods for obstacle avoidance. Rules can also be defined to allow dynamic node selection based on constraints. An extension is also proposed to generate a parallel custom graph composed of custom nodes. This enables a precise route creation and online update mechanism based on the mission details and findings.

During the operation, the first step is to initialize the workspace in the form of a 3D occupancy map based on the ship's particular dimensions and generate a planning graph inside it. Numerous collision free paths are then found as the map is being gradually built. An inspection pattern is generated to fit a classic lawnmower pattern and added to the graph. This allows the vehicle to efficiently find escape routes in case an obstacle is observed on the survey path, and to quickly return to it.

Since the precise ship geometry is not known, a classic vertical lawnmower pattern is first generated based on two inputs: height and width. The goal of this path is to provide full visual coverage. It can be formed in two ways, with vertical or horizontal segments which divide this inspection area in vertical or horizontal slices. Therefore, the area the camera can cover at a specific distance and the wanted visual overlap need to be considered. Each waypoint (WP) of the path P is defined by a vertical and horizontal distance, respectively Δ_v and Δ_h, to the starting position denoted WP_0, that is,

$$\mathrm{P} := [\mathrm{WP}_0, \mathrm{WP}_1, \mathrm{WP}_2, ..., \mathrm{WP}_n] \tag{6}$$

$$\mathrm{WP}_i := \left[\Delta_{v,i}, \Delta_{h,i}\right]^\top, \tag{7}$$

where i indicates the i$^{\text{th}}$ waypoint. We consider horizontal slices, but similar operations can be performed for a path based on vertical slices. The number of horizontal lines n_{hlines} is

$$n_{hlines} := \left\lceil \frac{\mathrm{H}}{\mathrm{Cvrg_v} - \mathrm{Ovrl} \cdot 2} \right\rceil, \tag{8}$$

where $\lceil \cdot \rceil$ is the *ceil* mathematical operation, H is the total height that needs to be visually covered, $\mathrm{Cvrg_v}$ (abbreviation of Coverage) is the vertical coverage of the camera, and Ovrl (abbreviation of Overlap) is the required visual overlap. Ovrl is multiplied by two to take into account the two overlapping areas, with the slices above and below. The horizontal and vertical distances are then calculated by

$$\Delta_{h,i} := \begin{cases} \mathrm{W}(i \bmod 2), & \text{if } (i \bmod 4) < 2 \\ \mathrm{W}\,|(i \bmod 2) - 1|\,, & \text{otherwise} \end{cases} \tag{9}$$

$$\Delta_{v,i} := -\left\lfloor \frac{i}{2} \right\rfloor (\mathrm{Cvrg_v} - \mathrm{Ovrl}) - \frac{\mathrm{Cvrg_v} - \mathrm{Ovrl}}{2}, \tag{10}$$

using $\lfloor \cdot \rfloor$ as the *floor* operation, and W is the total surface width that needs to be visually covered for each slice. The *mod* operation corresponds to the modulo operation returning the remainder of the division. Applying Eqs. 9 and 10 will results in a correct path. However, it may not be centered on the structure, which can present risks for the visual coverage. For this reason, horizontal and vertical shifts should be added, according to

$$\mathrm{Shift}_h := \frac{\mathrm{Cvrg_h}}{2} - \mathrm{Ovrl}(2 \cdot (i \bmod 2) - 1) \tag{11}$$

$$\Delta_{h,i} = \begin{cases} \Delta_{h,i} - \mathrm{Shift}_h, & \text{if } (i \bmod 4) < 2 \\ \Delta_{h,i} + \mathrm{Shift}_h, & \text{otherwise.} \end{cases} \tag{12}$$

The horizontal Shift_h is then applied to the horizontal distances. For the vertical Shift_v, we use

$$\mathrm{Shift}_v := \frac{1}{2}\left(-\mathrm{H} + n_{hlines}(\mathrm{Cvrg_v} - \mathrm{Ovrl}) + \mathrm{Ovrl}\right) \tag{13}$$

$$\Delta_{v,i} = \Delta_{v,i} + \mathrm{Shift}_v, \tag{14}$$

where $\mathrm{Cvrg_h}$ is the horizontal coverage of the camera. An example of a final generated pattern is depicted in Fig. 5(a) with the camera coverage of each horizontal slice in different colors to highlight the overlap. The next step is to rotate and translate the pattern to the vehicle's reference frame and align it with the ship.

6 Online Path Adaptation

As soon as the vehicle starts to follow the pattern to inspect the hull, the path needs to be updated for the drone to keep a constant distance to the hull. It needs to be constant to ensure good visual coverage. If the distance varies, spots will be missed if the drone is too close or details could be missed if it is too far away. Over time, with a constant distance, the pattern should adapt to the actual shape of the ship. A multibeam forward-looking sonar is deployed to achieve this, imposing the constraint to always be facing the hull. By assuming that most parts of the structure are locally flat, it becomes possible to perform line detection based the acoustic image from the sonar measurement and similar to a wall following problem.

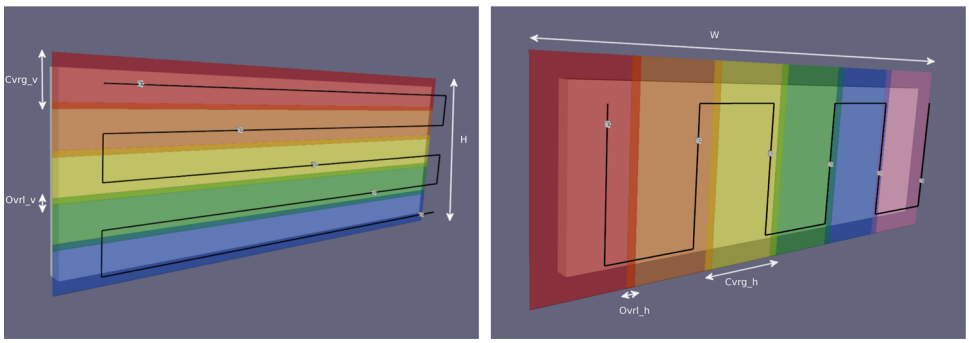


Fig. 5 (a) Vertical lawnmower pattern with horizontal slices generated to visually cover a plane of size 30 × 10 meters. Each horizontal slice is depicted with a different color to represent the coverage difference in each slice. In this case, a camera coverage of 3 × 2.5 meters with a required overlap of 0.3 meters was used and resulted in the creation of 5 horizontal slices. (b) Equivalent pattern with vertical slices

6.1 Wall Detection

Considering that the hull is locally flat, estimating the hull orientation and position with respect to the vehicle using line detection methods on the acoustic data remains the most efficient approach. A comparison of line detection methods is done in [30]. The main algorithms are described and tested on 2D data from a laser rangefinder. Methods based on the RANSAC algorithm have the important advantage of fitting in the presence of outliers. However, in the presence of multiple dense area of features, it can be largely biased and result in a high number of false positives. As a counter measure, it is necessary to sparsify these areas or apply a constraint to require better distribution of the features on the line. We propose to first apply an edge detection operator to address this issue. It has the effect of significantly reducing the number of features while keeping those of interest that are on the line. That way, RANSAC is less subject to bias, and the line features are highlighted and better distributed. The Canny edge detector [31] is employed to achieve this. Both techniques have been widely used in computer vision applications and are known for their efficiencies [32].

With the acoustic data from the sonar, the intensity map is first formed based on the ranges and bearings, displayed in Fig. 6(a). Threshold operations are performed before the Canny edge detector is applied. Pixels with intensity lower than T_{low} and higher than T_{high} are removed and set to 0. After processing the edge detector, including the Gaussian filter to smooth the image, all pixels considered as edge are stored and converted from Polar to Cartesian coordinates. It is now possible to fit a reliable line with the remaining points. Although there are outliers left, as observed in Fig. 6(b), they will be detected as such in the RANSAC process. The fitted line, displayed in Fig. 6(c), is visually correct and corresponds to the object's pose relative to the drone.

Using the line's geometric details, representing the wall locally, it is possible to place it in the ROV reference frame based on the vehicle's heading ψ, the detected line inclination α_l relative to the normal of the line of sight of the vehicle, and the forward distance d_w from the ROV to the object. The definition of the wall is based on the point w, the point on the wall the drone is looking at, given by

$$w = \begin{bmatrix} w_x \\ w_y \end{bmatrix} := \begin{bmatrix} p_x + \cos(\psi)\, d_w \\ p_y + \sin(\psi)\, d_w \end{bmatrix}, \tag{15}$$

where (p_x, p_y) is the drone's position. The orientation of the wall is estimated by

$$\alpha_w := \psi - \alpha_l + \frac{\pi}{2}. \tag{16}$$

Therefore, any point w_k on the line going through w and with orientation α_w is given by

$$w_k = \begin{bmatrix} w_{k,x} \\ w_{k,y} \end{bmatrix} := \begin{bmatrix} w_x \pm \cos(\alpha_w)\, c_k \\ w_y \pm \sin(\alpha_w)\, c_k \end{bmatrix}, \tag{17}$$

where $c_k > 0$ is an arbitrary constant that represents the distance to the main point w. The wall's coordinates are estimated in the 2D plane Oxy and positioned at the vehicle's depth, p_z, in the vertical direction. The geometry involved is depicted in Fig. 7 with examples of positions.

6.2 Path Update Mechanism

With the position of the ROV and the hull known, the initial path P can be updated to a new path P′ to adapt to the

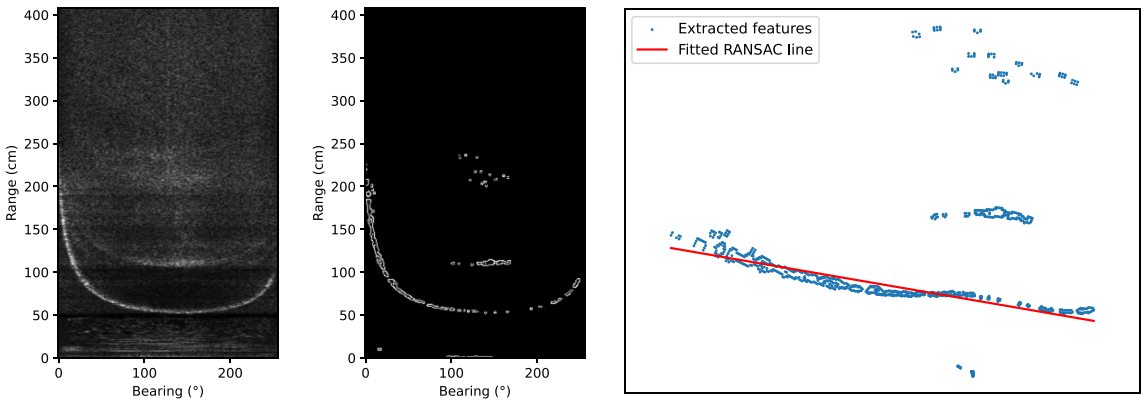


Fig. 6 Representation of the detection process of the local wall-shaped object, where (a) is the raw sonar data presented as an image, (b) depicts the results after the thresholding operations and the application of the Canny edge detector, and (c) displays the fitted line using RANSAC on the remaining features from (b) in Cartesian coordinates

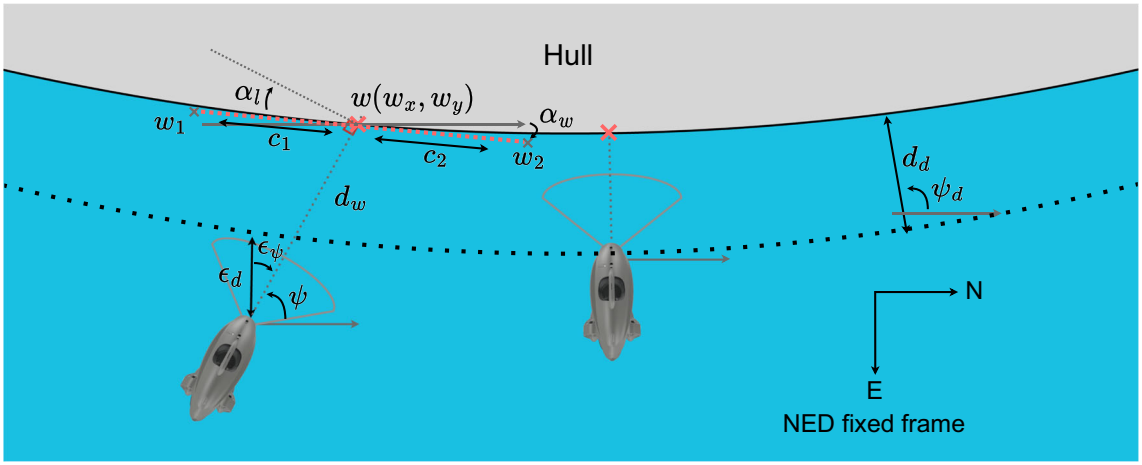


Fig. 7 Here is represented the geometry of the method. In the first case (drone on the left part of the image), the ROV with heading ψ has a forward distance to the wall of d_w, which creates the point located in $w(w_x, w_y)$. The angle between the abscissa axis and estimated local wall orientation is here very small and represented by α_w. With the constraint of facing the wall at a specific distance, ϵ_d and ϵ_ψ represent respectively the errors in distance and heading. The second drone satisfies the constraints

structure in front of the ROV. For each new sonar measurement, the path is re-positioned, i.e., translated and rotated. This is achieved by considering a desired distance d_d to the hull and a generic path variable θ, representing the progression on each path segment. d_d is provided by the user and is constant. θ is calculated based on the current position and the two waypoints $\{\text{WP}_{i-1}, \text{WP}_i\}$, the previous and current targets, that is,

$$\theta := \frac{||\text{Pr}(p,\ \text{WP}_{i-1},\ \text{WP}_i) - \text{WP}_{i-1}||}{||\text{WP}_i - \text{WP}_{i-1}||}. \tag{18}$$

The function Pr, projecting a point ξ onto a line going through p_1 and p_2, is defined according to

$$\text{Pr}(\xi,\ p_1,\ p_2) := p_1 + ((\xi - p_1) \cdot (p_2 - p_1))\frac{p_2 - p_1}{||p_2 - p_1||^2}. \tag{19}$$

By then projecting the ROV position on the estimated local wall, which corresponds to the closest point on the wall from the ROV, it becomes possible to create a parallel line at an offset distance d_d. However, because θ represents the normalized progression on the current path segment, i.e., $\theta \in [0, 1)$ between WP_{i-1} and WP_i, the general direction dir between the previous and next waypoints of the path, relative to the starting pose of the drone, must be taken into account. This corresponds to how the drone should be moving in its own frame until the next waypoint is reached. It has the three possible outcomes $[-1, 0, 1]$, which respectively correspond to a left-right direction, vertical direction, and right-left direction from the ROV's perspective. Therefore, it should respectively be moving either sideways to its right, or up/down vertically, or sideways to its left. The computation of this value depends on the type of inspection pattern. In case of a pattern with vertical slices, dir is given by

$$dir(i) := (i \bmod 2) - 1, \tag{20}$$

where i indicates the i$^{\text{th}}$ target waypoint. Similarly, in case of a pattern with horizontal slices, dir is calculated by

$$dir(i) := (i \bmod 4 - 2)(i \bmod 2). \tag{21}$$

With these definitions of directions, the vehicle is by default following the path at the start from left to right and from up to down. The new waypoints WP'_{i-1} and WP'_i are adapted, respectively, from WP_{i-1} and WP_i using the sonar measurements. They satisfy local constraints and become

$$\text{WP}'_{i-1} := p' - \theta\, dir(i) \begin{bmatrix}\cos\alpha_w \\ \sin\alpha_w\end{bmatrix} ||\text{WP}_i - \text{WP}_{i-1}|| \tag{22}$$

$$\text{WP}'_i := \text{WP}'_{i-1} + dir(i) \begin{bmatrix}\cos\alpha_w \\ \sin\alpha_w\end{bmatrix} ||\text{WP}_i - \text{WP}_{i-1}||, \tag{23}$$

with WP'_i defined based on WP'_{i-1} and the path segment length. p' is the position where the ROV should be with the new pair of constraints on P′. It corresponds to a point on the line going from the projected position of the drone on the wall p_{proj} to its own position p, that is,

$$p_{proj} := \text{Pr}(p,\ w_1,\ w_2) \tag{24}$$

$$p' := p_{proj} + d_d \begin{bmatrix}\cos\alpha_d \\ \sin\alpha_d\end{bmatrix}, \tag{25}$$

where w_1 and w_2 are defined using Eq. 17, and α_d is the angle

$$\alpha_d := \text{atan2}(p_y - p_{proj,y},\ p_x - p_{proj,x}). \tag{26}$$

The desired heading ψ_d satisfying the current constraint of facing the hull can now be obtained. Similar to the path, it is re-estimated for each sonar measurement, at approximately 10Hz. The frequency can be considered as relatively high but it has a natural averaging effect because of the control-guidance-localisation rate differences. As a result, if an error occur with a sonar measurement, it will be quickly eliminated. The new angle is derived from α_d and corresponds to its opposite angle that is

$$\psi_d := \alpha_d + \pi. \tag{27}$$

This path adaptation part is executed online for each estimated pair of constraints considered as valid. An example of the results is depicted in Fig. 8 with a drone having similar constraints and a wall that has a slight curvature, requiring the vehicle to adapt.

The new path P′, desired heading ψ_d, and position p are then passed as parameters to the guidance stack to solve the desired velocity and position along the path. The function sequence to produce these is provided as pseudocode in Algorithm 1.

Algorithm 1 Update path segment

input : 2D position of the ROV p, current path segment S, local wall segment W and its orientation α_w, desired distance d_d
output: updated path segment S' and desired heading ψ_d

$\theta \leftarrow$ get_segment_progress(p, S) Eq. 18
$p_{proj}, \alpha_d \leftarrow$ get_projected_pose(p, W) Eqs. 24 and 26
$p' \leftarrow$ offset_point(p_{proj}, d_d, α_d) Eq. 25
if *Pattern* **is** *vertical* **then**
  $dir \leftarrow$ get_current_vdir() Eq. 20
else
  $dir \leftarrow$ get_current_hdir() Eq. 21
$S' \leftarrow$ update_segment(p_{proj}, θ, α_w, S) Eqs. 22 and 23
$\psi_d \leftarrow$ get_desired_heading(α_d) Eq. 27

7 Path Following

With the drone being able to locate itself, generate path segments, and adapt them over time, all the tools that are necessary for efficient path following along the computed segments are established. To develop a fully autonomous system, two independent types of paths are defined: transit and inspection paths. The former provides a path from the deployment position to the starting position of the inspection. The latter defines the actual route for the drone to follow during the survey. The guidance method is also different for the two operation modes.

The vehicle is controllable in 4 DOF, where the heave motion is decoupled from the horizontal DOF and controlled using a vertical thruster. For the path-following problem, depth control is, hence, solved independently from the guidance problem in the horizontal plane.

7.1 Guidance for Transit Paths

To follow a transit path, a method using the Line Of Sight (LOS) steering laws based on [33] and [34] for straight-line following is developed and presented in deeper details in [23]. The purpose is to give inputs for heading control while following the path. The time-varying term $\Delta(y_e)$ [35] represents the line of sight distance, which is the distance to a target point on a tangent line to the path that the drone aims at. This term is designed to vary with the cross track error y_e to avoid

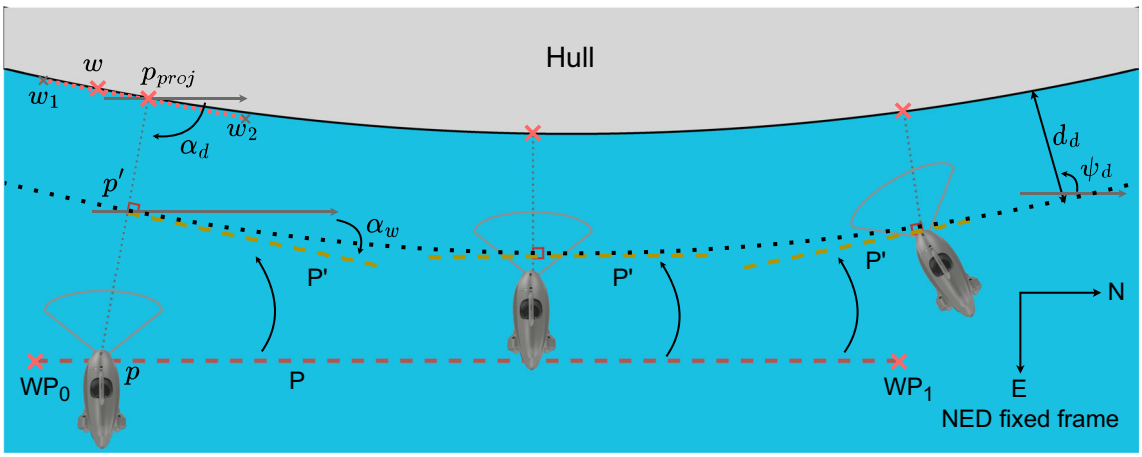


Fig. 8 The path adaptation along the wall is depicted here along with the geometry involved. P is the original path segment (red) and transformed into P' (yellow) for each new sonar measurement resulting in a new estimate of the wall orientation and its distance to the ROV

aggressive and oscillatory movements. When estimating the desired heading ψ_d, an integral action is also incorporated [36] to compensate for external disturbances, that is,

$$\psi_d = \gamma_p - \text{atan2}(y_e + \kappa y_{int}, \Delta(y_e)) \tag{28}$$

$$\dot{y}_{int} := \frac{U y_e}{\sqrt{\Delta(y_e)^2 + (y_e + \kappa y_{int})^2}}, \tag{29}$$

where γ_p is the horizontal path-tangential angle and U, the total velocity of the ROV. The integral action is tuned by $\kappa > 0$.

The vehicle has reached the waypoint when it has entered a circle of acceptance in the horizontal plane and its depth error is below a specific threshold.

7.2 Guidance for Inspection Paths

Although a method was developed in previous work for the inspection path guidance, the lack of precise control in slow speed inspection mode redirected the method to a more convenient solution for that scenario, maneuvering-based guidance [37, 38]. This method provides more convenient specification of the speed along the path as well as a better compensation of the heading and depth errors during the inspection. These two errors are especially important for the inspection procedure, and priority is given to them over positioning in the horizontal plane. When the errors are considered as too large, the inspection motion should be paused until they are corrected. This contributes to the inspection quality.

In the maneuvering problem [37], the goal is to converge to and follow a continuously parameterized path. The problem can be divided in two parts, to stay on and follow the path and to satisfy a speed assignment along the path. Therefore, a desired speed requirement is assigned to the vehicle if the path is being followed well, i.e., the vehicle is close to the path. If this is not the case, in order to better compensate for the path following errors, the speed should be adjusted to prioritize controlling the vehicle towards it.

To describe the desired position and speed along the path, each path segment are redefined as

$$\rho_i(\theta) := (1 - \theta) p_{i-1} + \theta p_i, \tag{30}$$

where the waypoints WP'_i estimated from the sonar are used as p_i in the horizontal plane. Therefore, p_{i-1} and p_i are respectively the previous and next waypoints in the set of waypoints $\{p_0, p_1, p_2, ..., p_n\}$. By definition, ρ has then the following limits: $\rho_i(0) = p_{i-1}$ and $\rho_i(1) = p_i$, since local $\theta \in [0, 1)$. Subsequently, a global dynamic path variable $s \in [0, n)$ can be defined, representing the global progression on the path. The following mapping,

$$i(s) = \lfloor s \rfloor + 1 \tag{31}$$

$$\theta(s) = s - \lfloor s \rfloor, \tag{32}$$

enables correspondences between the local and global progression and the index of the current path segment. The desired position is

$$p_d(s) := \rho_{i(s)}(\theta(s)), \tag{33}$$

which ensures a continuous and connected motion along all path segments. Similar to the path segments, but this time for the entire path, $p_d(0) = p_0$ and $p_d(k) = p_k$. We next design a speed assignment $v_s(s, t)$ for $\dot{s}$ along the path according to

$$v_s(s, t) := \frac{u_s(s)}{||p_d^s(s)||} u_d(t). \tag{34}$$

Here, $u_d(t)$ is the desired speed of the ROV along the path. This value can be set and updated manually or autonomously according to the inspection state. It is typically monitored by the ROV operator and inspector. Also note that $||p_d^s(s)||$ is the distance between p_i and p_{i-1} because the path is a set of straight line segments, and therefore $||p_d^s(s)|| = ||p_i - p_{i-1}||$ for $i = i(s)$. $u_s(s)$ is a speed modifier to help slow down the speed of the vehicle at corners, i.e., when close to a waypoint. It is using the saturation function $\text{sat}_1(x)$ to not exceed the bounds and is given by

$$\text{sat}_b(x) := \begin{cases} x, & \text{if } |x| \le b \\ \text{sgn}(x) b, & \text{otherwise} \end{cases} \tag{35}$$

$$u_s(s) := \begin{cases} \frac{\text{sat}_1(k_{slope}(s+1-i)) + u_0}{1+u_0}, & \text{if } s \in [i-1,\ i-\frac{1}{2}) \\ \frac{\text{sat}_1(k_{slope}(i-s)) + u_0}{1+u_0}, & \text{if } s \in [i-\frac{1}{2},\ i), \end{cases} \tag{36}$$

where $\frac{u_0}{1+u_0}$ is a regularization parameter and $\frac{k_{slope}}{1+u_0}$ becomes the convergence speed around the waypoints.

In order to compensate for the previously mentioned errors in depth and heading, the speed assignment should be modified to account for the errors ϵ_z and ϵ_ψ in, respectively, depth and heading. Accordingly, we define

$$\sigma_\delta(\epsilon) := \text{brelu}_1\left(k_\delta(\delta - |\epsilon|)\frac{1}{\delta} + 1\right) \tag{37}$$

$$\text{brelu}(x, b) := \min(\max(0, x), b). \tag{38}$$

The function Eq. 37 is a deactivation function with the effect that when $\epsilon > \delta$, i.e., when the threshold $\delta > 0$ is exceeded, the returned value decreases until it reaches 0. It can be seen as a gain in $[0, 1]$. Its slope at $|\epsilon|$ towards 0 is tuned by $k_\delta > 0$. To ensure that the ratio $\sigma_\delta(\epsilon)$ does not exceed the limits, the resulting value of the operation is passed through

the brelu(x, b) activation function which bounds the result in $[0, b]$, with in this case $b = 1$. Using Eqs. 37 and 38 on the two errors ϵ_z and ϵ_ψ, the modified speed assignment $v_s(s, t)$ becomes

$$v_s(s, t, \epsilon_\psi, \epsilon_z) = \sigma_{\delta_\psi}(\epsilon_\psi)\sigma_{\delta_z}(\epsilon_z)\frac{u_s(s)}{||p_d^s(s)||}u_d(t), \tag{39}$$

which will regulate the motions along the path according to the errors in heading and depth when these grow too large, that is, it will make the desired position $p_d(s)$ slow down and possibly stop until the errors are compensated. The desired velocity v_d is finally defined to make sure the vehicle reaches the waypoints. It is based on the constant bearing nonlinear approach from [39, 40]. It prevents the vehicle from deviating from the path segment by making it converge towards it, while maintaining its velocity along the path. To this end, the position error $e_1 := p - p_d(s)$ is included such that

$$v_d := p_d^s v_s - U_p \frac{e_1}{\sqrt{|e_1|^2 + \Delta_1^2}}, \tag{40}$$

where $U_p > 0$ is the approach speed, and Δ_1 is a tuning parameter. This approach also allows to robustify the path following system as the path as well as the vehicle evolve over time.

In this inspection mode, the desired heading ψ_d evolves over time and is set according to Eq. 27, as updated by the sonar measurements.

7.3 Depth System

The depth instructions are similar regardless of the navigation mode and waypoint positions. The desired depth z_d is always set to the next waypoint's depth, that is,

$$z_d := \mathrm{WP}_{i,z}. \tag{41}$$

Unlike the horizontal position of the waypoints, their depths are defined as constant. The desired vertical velocity w_d is designed to compensate the error in depth, such that

$$w_d := \mathrm{sat}_{u_z}(\kappa_z(z_d - p_z)), \tag{42}$$

where sat_{u_z} is defined in Eq. 35, u_z is the maximum desired vertical speed, and κ_z a gain setting the convergence rate.

7.4 Control System

To control the surge, sway, heave, and yaw motions, Proportional Integral Derivative (PID) controllers are employed. Each controller has its own set of gains K_p, K_i, and K_d, which also differs according to the navigation mode. The following errors $\epsilon_{p,1} \in \mathbb{R}^2$ and $\epsilon_{p,2} \in \mathbb{R}^2$ are respectively defined as state position error and velocity error in the 2D plane Oxy, that is,

$$\epsilon_{p,1} := R_\psi^\top(p_d - p(t)) = R_\psi^\top \begin{bmatrix} p_{d,x} - p_x(t) \\ p_{d,y} - p_y(t) \end{bmatrix} \tag{43}$$

$$\epsilon_{p,2} := v_d - R_\psi^\top \nu(t) = \begin{bmatrix} v_{d,x} - R_\psi^\top \nu_x(t) \\ v_{d,y} - R_\psi^\top \nu_y(t) \end{bmatrix}. \tag{44}$$

The load $\tau_p \in \mathbb{R}^2$, vector of desired forces and moments, is then given by

$$\dot{\xi}_p := \epsilon_{p,2} \tag{45}$$

$$-K_{p,i}\xi_p := \begin{cases} \tau_{p,\min}, & \text{if } -K_{p,i}\xi_p < \tau_{p,\min} \\ \tau_{p,\max}, & \text{if } -K_{p,i}\xi_p > \tau_{p,\max} \\ -K_{p,i}\xi_p, & \text{otherwise} \end{cases} \tag{46}$$

$$\tau_p := -K_{p,p}\epsilon_{p,1} - K_{p,i}\xi_p - K_{p,d}\epsilon_{p,2}, \tag{47}$$

with the addition of the anti-windup action on the integral term and the diagonal gain matrices $K_{p,p}, K_{p,i}, K_{p,d} \in \mathbb{R}^{2\times 2}$. Similar operations are performed with the errors in depth and heading,

$$\epsilon_{z,1} := z_d - p_z(t) \tag{48}$$

$$\epsilon_{z,2} := w_d - \nu_z(t) \tag{49}$$

$$\epsilon_{\psi,1} := \psi_d - \psi(t) \tag{50}$$

$$\epsilon_{\psi,2} := \dot{\epsilon}_{\psi,1} - \dot{\psi}(t). \tag{51}$$

A similar PID controller Eq. 47 is used for depth and heading control, using ϵ_z and ϵ_ψ instead of ϵ_p, resulting in respectively the force τ_z and moment τ_ψ. The gains are set individually for each controller.

8 Inspection Map

To keep track of the inspection progress during the operation, an inspection map is built online. Using sonar and navigation data, everything seen by the vehicle is registered with position, and point clouds are established to form an occupancy map. The map contains local uncertainty information that the operator can use to assess the reliability of the inspection by area. Automatic detection of coverage holes is added to allow the autonomous system to find them and (re)inspect the area. The map is here referred to as the inspection map, because it is solely used for the drone to keep track of the mission and not other tasks such as the ship hull reconstruction—which is an independent task.

8.1 Occupancy Map Generation

Although the line detection method presented in Section 6.1 provides good estimates of the structure in front of the vehicle, it is not accurate enough to build a map because the features of the sonar scans are not always aligned. Using the assumption that the surface in front of the ROV has a flat shape, the closest detected feature for each sonar beam provides a set of information about the surroundings that allows the creation of the inspection map. After processing the sonar image, it is possible to obtain features like in Fig. 9, and represent them in the vehicle's frame. The set of 3D points is given by

$$f^V := p + f^S R\left(\psi - \frac{\pi}{2}\right), \tag{52}$$

where f^V is in the vehicle's frame and extracted from the sonar points f^S converted to Cartesian coordinates. The original 2D sonar features f are transformed into f^S using the conversion

$$f_i^S := \begin{bmatrix} \sin\alpha_{f_i} \\ \cos\alpha_{f_i} \\ 0 \end{bmatrix} r_{f_i}. \tag{53}$$

For each beam i, the pair in polar coordinates (α_{f_i}, r_{f_i}) corresponds respectively to the horizontal beam angle and range of the measurement. The set f^V can then be used to build the 3D occupancy map. The grid is defined by its resolution, i.e., how many cells or voxels that are contained in one unit distance. A resolution too high may result in important memory consumption and heavy computational costs. Depending on the mapping strategy, missing parts in the grid may also happen. On the contrary, a resolution too low will have the consequence of imprecision and inconsistency in the map, but with lower computational costs. The sensors characteristics can also have an impact on the mapping task depending on the resolution [41]. The map resolution should therefore be chosen carefully. In this project, high resolution is not required. The resolution should match the actual information content.

Before building the map, the sonar features must be filtered for noise and outliers. To achieve this, a method based on averaged point distances is proposed. It measures the average distance between the surrounding points in a window for each point at the time. For present objects, corresponding feature points of consecutive beams should be close to each other. The window size $s \in \{2x \in \mathbb{Z}^+\}$ selects a total of $s+1$ consecutive beams, with $s/2$ beams before and after the focus point. The average distance value for a point becomes

$$\bar{d}_i^f = \frac{1}{s+1} \sum_{j=i-\frac{s}{2}}^{i+\frac{s}{2}} \sqrt{(f_{i_x}^S - f_{j_x}^S)^2 + (f_{i_y}^S - f_{j_y}^S)^2}, \tag{54}$$

where f_i^S is the focus point for all other points f_j^s surrounding it. The process is represented in Fig. 10. This method is more robust and less prone to true positive rejection compared to methods such as bin based evaluations [42] using the direct sonar distances to the points. In such methods, values at both end of the bin see their chances of being rejected greatly increase whereas with a moving window evaluating each neighborhood individually, the bias brought by the mean is vanished. Note, however, that a cluster of noisy points will anyway create a local bias and might increase rejection of true positives in that area. The rejection of the point is then

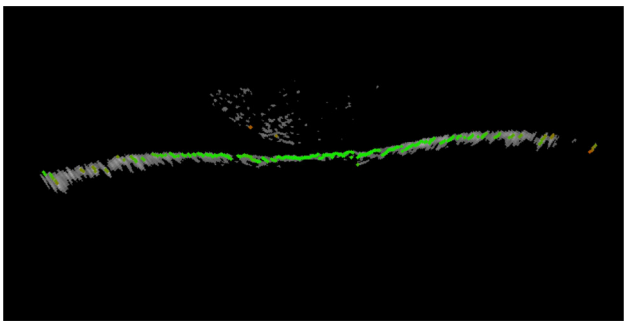


Fig. 9 The detected features on the sonar image are represented as green squares. There is only one feature per beam. The scan does not display perfectly linear features because of the estimation of the speed of sound. However, because of the sampling method, it become negligible, and the error is in the order of a millimeter

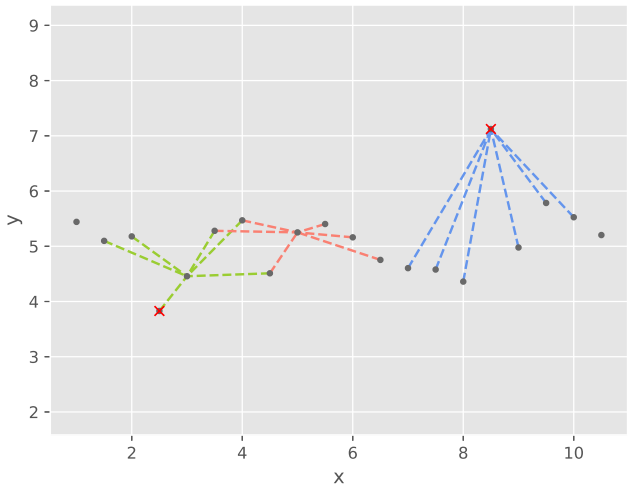


Fig. 10 The moving average outlier removal procedure with three windows displayed. Each window is represented by a different color (green, red and blue) and is centered on the starting point of all lines. In this case the window size is 7. With a threshold distance of 1.5, points marked with a red cross would be rejected, the blue window therefore evaluated the neighborhood as too far away

based on the threshold test

$$\begin{cases} \text{accept}, & \text{if } \bar{d}_i^f < T_f \\ \text{reject}, & \text{otherwise.} \end{cases} \tag{55}$$

Over time a dense point cloud is generated and enables the creation of the voxel map. A voxel is created only if there are enough reliable points inside the area of the voxel. Figure 11 presents both the dense point cloud and its conversion into a voxel map that was automatically processed during an autonomous inspection mission of a wall. In Fig. 11(a), it is possible to observe behind the wall of green points, points in shades of blue that are the rejected points. Note the sparse distribution of points at the left and right edges of the cloud, this is due to the configuration and resolution of the sonar that makes it less precise with beams at the edges of the sensor. In Fig. 11(b), the voxel map width is smaller that the wall of points because the sparse parts of the point cloud is not considered during the voxel creation.

8.2 Uncertainty Estimations

When performing the mapping task, there can be multiple sources of errors that can make the final map or some parts of it unreliable. Evaluating the uncertainty of the data instantly provides relevant information to the ship hull inspector, and poorly covered areas can then be re-inspected by the autonomous vehicle upon request. The main source of uncertainty of the map is the position inaccuracy of the drone, and errors in vehicle position estimates can greatly impact the final results, i.e., the position of the voxels. The second source of uncertainty considered here is the sonar accuracy, including acoustic conditions and the assumption of a flat surface. The latter is the first subject for a test to decide if the drone is actually looking at a wall. To achieve this, the distances $\bar{d}^f$ are fitted to a probability distribution whose parameters describe the shape of the sonar image features.

The fitness of distributions was tested experimentally and the average distances were found to fit best the Gamma distribution $\bar{d}^f \sim \Gamma(\alpha, \beta, \gamma)$ with a shape parameter α, a scale parameter β, and shift parameter γ (shifting the distribution to the right by γ). This was done by performing and comparing log-likelihood goodness-of-fit statistical hypothesis tests and by analysing the reliability functions. To first assess the type of object in front of the sonar, i.e. a wall or something else, the skewness w of the distribution is used as key value. It is determined by the shape and is given by

$$w := \frac{2}{\sqrt{\alpha}}. \tag{56}$$

Although the point distances follow a Gamma distribution even when a wall is not present, the shape is significantly impacted and makes it possible to create correspondences between the shape parameter α and what is in front of the sonar. The distribution is expected to be strongly positively skewed when all the points are aligned, indicating the presence of a flat surface. In this case, points are close to each other with only the bearings of the sonar beams affecting the distances between the points. Therefore, points with very low $\bar{d}_i^f$ will accumulate quickly, creating the skewness of the distribution. To determine if a flat surface is detected, the test

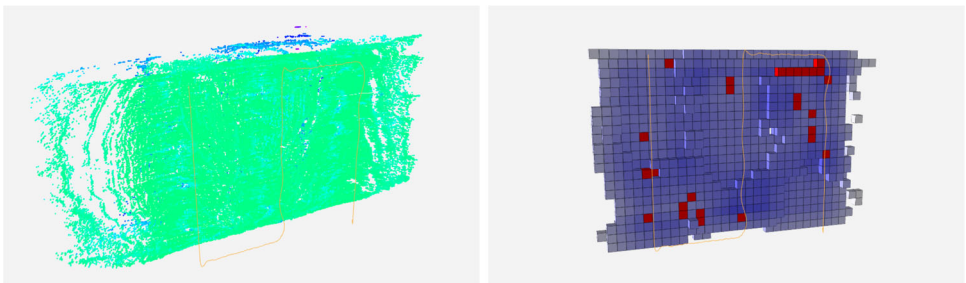
$$w > c, \tag{57}$$

is performed, where the value c is an arbitrary constant threshold value. Using $c = 2$ allows tolerant detection, to accept not exactly flat surfaces or unlinked flat surfaces. A higher value makes it stricter.

The uncertainty from the sonar is evaluated using the distribution's variance with a population mean given by the estimated true mean for the line inclination α_l, and forward distance d_w calculated in Section 6.1. For that purpose, the estimated true population is estimated by first generating a feature per sonar beam. The estimated true sonar points f^T in Cartesian coordinates are expressed as follows:

$$f_i^T := \begin{bmatrix} \sin\alpha_{f_i} \\ \cos\alpha_{f_i} \end{bmatrix} r_{f_i^T} \tag{58}$$

$$r_{f_i^T} := \frac{d_w \sin\beta_A}{\sin\beta_M}, \tag{59}$$



(a) (b)

Fig. 11 The vehicle path for an autonomous inspection of a wall displayed as an orange line in front of the mapped area. In (a), the dense point cloud is displayed with green points as inliers and the rest as outliers. (b) corresponds to the conversion of the point cloud to the voxel map. Red voxels are detected missing voxels as explained in the later Section 8.3

considering the three points O, A, M forming an ordinary triangle with the corresponding angles $\beta_O, \beta_A, \beta_M$ and $\beta_A = \frac{\pi}{2} + \alpha_l$. By then applying (54) to f^T, a new set of distances $\bar{d}^{f^T}$ is obtained, corresponding to how the data should look like. Calculating the variance of the set $\bar{d}^f$ therefore becomes

$$\text{var}(\bar{d}^f) = \frac{1}{n}\sum_{i}^{n}\left(\bar{d}_i^f - \mu_{\bar{d}^{fT}}\right)^2, \tag{60}$$

indicating how far the data is from the expected true mean, where $\mu_{\bar{d}^{fT}}$ is the mean of all averaged distances in $\bar{d}^{f^T}$. The main source of uncertainty, the position of the vehicle, is represented by the state covariance matrix. However, to better represent the uncertainty in the inspection map, a single value must be extracted from the covariance matrix. This representative value should be defined using an 1-homogeneous function S with a normalization constraint [43]. The approach chosen here is based on the point variance describing the location precision [44, 45],

$$S(\mathrm{K}_{\mathrm{x}_{core}}) := \frac{\text{trace}(\mathrm{K}_{\mathrm{x}_{core}})}{k}, \tag{61}$$

where $\mathrm{K}_{\mathrm{x}_{core}}$ is the covariance matrix of the drone's core state x_{core}. The trace or the equivalent sum of the matrix eigenvalues is divided by k, the total number of core components included in the matrix.

To mix and obtain only a single value from both sources, a weighted sum is applied to give the possibility to emphasize one source more than the other, that is,

$$u := S(\sigma_{\mathrm{x}_{core}}\mathrm{K}_{\mathrm{x}_{core}}) + \sigma_{\bar{d}^f}\,\text{var}(\bar{d}^f), \tag{62}$$

with u the final uncertainty estimate, and the weights $\sigma_{\mathrm{x}_{core}} \in \mathbb{R}^{k,+}$ and $\sigma_{\bar{d}^f} \in \mathbb{R}^+$. Calculated during the mapping process, the uncertainty can be visualized as a color shade based on u, as shown in Fig. 12. First, the uncertainty from the sonar Fig. 12(a) is displayed; secondly, the uncertainty of the position estimates is shown in Fig. 12(b); finally, both components are combined in Fig. 12(c). It is important to note that a red voxel does not always mean it is specifically uncertain, but instead that it is the most uncertain of the map. The pipeline of the voxel map creation is illustrated in Fig. 13 together with the uncertainty estimation.

8.3 Hole Detection

The last step of the mapping process is to detect holes in the voxel map for identification of potential missing coverage, indicating places that may need re-inspection.

Existing methods in literature, presented in Section 1, are not suited for the inspection map developed in this project. The shape of the structure being inspected makes it challenging to rely on height jumps alone or projections on 2D planes. The method proposed to perform hole detection is based on neighbors checking in a 3D moving kernel. With a cubic shape, the kernel moves in the existing voxels and first selects all empty cells around the voxel located in the center of the kernel and add them to the candidate list. The selected empty cells are then considered as part of a hole if they satisfy neighborhood constraints: the number of existing voxels around it is above a threshold and the number of lines involving symmetrical voxels with respect to the center of the local neighborhood is above a second threshold. For a line to exists and be considered, there must be at least one occupied cell on both sides of the line. Depending on the size of the window, there could be more than one voxel on both sides without being symmetrical. This will still be considered symmetrical since they are on a common line. This methods allows 3D structural constraints that can be adjusted with the thresholds. High thresholds will results in the detection of only closed spaces whereas low thresholds can work with

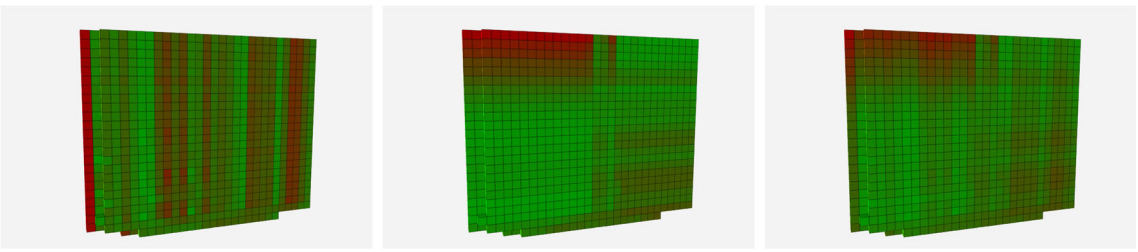


Fig. 12 Uncertainty visualisation in the voxel map. Note that the more a voxel tends to the red color, the more it is uncertain. In (a), the uncertainty in the sonar measurements is projected and in (b), the uncertainty in the core state of the vehicle is propagated to the voxel map. Both are combined in (c) with more weight on the uncertainty from the pose of the ROV

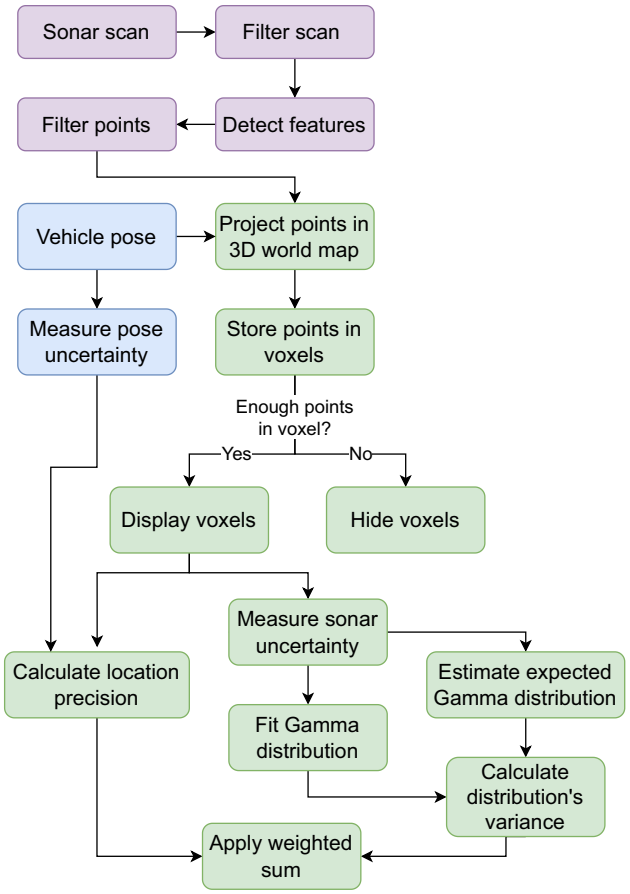


Fig. 13 The occupancy map creation pipeline with the uncertainty estimation

open spaces. It will also identify hollowed corners as missing cells and consider them as a parts of the map where points might be missing. The detected holes are then interpreted as potential missing parts in the inspection map. Examples of detected holes with several neighborhood configurations are presented in Fig. 14.

9 Component and System Integration Testing

Prior to the field trials, each component was extensively tested, in a simulated environment or in a test setup using previously logged sensor data. The *guidance* and *control* modules were developed using a simulated environment with static inspection paths to provide a minimum viable solution with stable control and appropriate response. Outputs from the *sonar* module were also simulated to test the path update mechanism and make sure the vehicle reacts safely to the changes. The *sonar* module was evaluated using real-world data acquired while operating the ROV manually for ship hull inspection. This enabled assessment of the pose estimation of the wall relative to the vehicle. Similarly, for the *mapping* module, the ROV was driven manually in front of a wall with known geometry which enabled efficient tuning of the parameters and assessment of the outlier rejection results. Finally, the integrated system was tested in an outdoor pool, by repeatedly inspecting a wall, simulating a ship hull. The system was ready for full-scale testing when all components had shown consistent performance for all testing cases.

10 Inspection of a Vessel

To assess and validate the proposed methods, full-scale ship hulls have been mapped. Mono-hull commercial and research vessels which present different constructive features were considered, including the main and most common hull types: displacement, semi-displacement, and planning hull types. Field trials were conducted in three shipyards and on ships of different types and sizes. The summary of the ships autonomously inspected using the presented solution is displayed in Table 2. In addition to repeated trials, this variety of vessels allowed to test the universality and repeatability of the proposed approach.

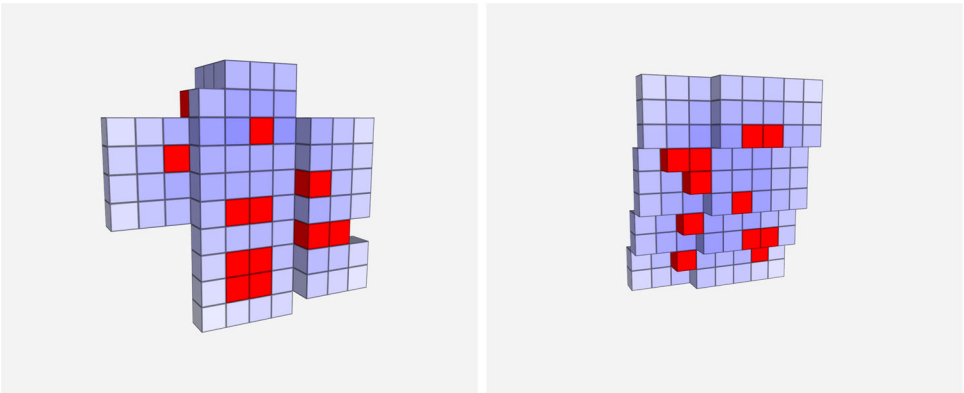


Fig. 14 Detection of potential missing voxels (red) in different scenario. In (a), holes of different sizes and shapes are placed on the structure and correctly detected. In (b), in addition to holes, red voxels are placed in corners as they are considered as potentially missing data

Table 2 Ship details

Location	Category	Hull Type	Size
Perama shipyard	Ro-ro cruise ferry	Displacement	$146 \times 26 \times 5.7m^3$
	Ro-ro cruise ferry	Displacement	$102 \times 21 \times 5.2m^3$
Klagenfurt lake	Lake ferry	Planning	$38 \times 8 \times 1.1m^3$
	Lake ferry	Planning	$40 \times 6 \times 1.1m^3$
Trondheim fjord	Research vessel	Displacement	$31 \times 10 \times 3.0m^3$
	Ro-ro ferry	Semi-displacement	$48 \times 14 \times 3.5m^3$

In this section, the inspection of a Ro-Ro cruise ferry from the port of Perama, near Athens in Greece, is considered in detail. The selected ferry is 146 m long, has a beam of 26 m, and draught of 5.7 m. It was repeatedly inspected to ensure the viability and consistency of the system. The additional details about the vessel are anonymised at the request of the shipowner. In this section, two surveys of pattern size 30×5 m^2 are considered. They consist of two autonomous inspections using different inspection patterns. The first survey contains horizontal slices spaced vertically 0.7 m apart and with 1.0 m distance requirement to the hull. The second survey contains vertical slices with 2.0 m horizontal distance using 1.3 m distance to the hull. In both cases, the inspection start at the water surface and ends at the keel. The two surveys are summarized in Table 3.

The operator's laptop was a standard consumer laptop, containing an Intel Core i7 vPRO CPU and 16GB RAM. Parts of the implementation of the methods are done in Python and others in C++ for optimization and compatibility purposes. Computer vision operations are performed using the OpenCV library[2].

10.1 Real-Time Maneuvering Evaluation

First, the maneuvering is evaluated by measuring the position, heading, and distance errors over time as the difference between the actual states and desired states. The results are shown in Fig. 15. Both path types had characteristic navigation error modes related to the heading. Because of the required motions in the second survey, which contains more diverse and simultaneous motions, combined with the thruster setup of the ROV, yaw motions are more sensitives and can be amplified by the surge motions. Also, since in this second survey the drone is regularly going back to the water surface, the impact of waves are more significant on the positioning than in the first survey. The sonar measurements have an impact on the estimated errors as for each new sonar measurements received, the path is re-adapted and therefore a new desired position is estimated. This is done on average at a rate of 10Hz during the experiment. Globally, the navigation and maneuvering errors remain in an acceptable range and do not have an impact on the visual and acoustic coverage of the hull.

A comparison between the proposed localisation filter, MaRS, explained in Section 4, and a basic Kalman Filter fusing the measurements from the DVL and IMU is done to highlight the relevance of the framework. The results can be visually compared in Fig. 16. Although it is not possible to quantity how accurate the localisation system is, it is clear that the MaRS framework outperforms the embedded localisation filter. The drift in attitude and the depth measurement errors are apparent and easily identifiable, since with such a trajectory the vehicle would have hit the hull multiple times. Furthermore, the trajectory produced by MaRS can be verified using the 3D inspection map as it would expose the errors and contain inconsistencies. The two trajectories drifted from each other at a rate of 0.008m/s, ending with 5.6m position difference. Similarly, the heading drifted at a rate of 0.026°/s and the operation finished with a heading difference of 18.26 degrees.

For more comparisons and details on the performance of the MaRS framework, the reader is recommended to read [22, 26].

10.2 Inspection Map Generation

The generation of the inspection map of the hull section is done in real-time in both scenarios. The resolution of the 3D grid is purposely increased. This has the effect of increasing the number of small holes. Therefore, only large gaps are considered for re-mapping, indicating truly a missing coverage. The results are displayed in Fig. 17 with the uncertainty as color gradient. In both cases, the inspection operation started from the top right and finished at the bottom left. The observed results have significant differences, especially

Table 3 Survey details

Survey	Pattern type	Pattern size	Slice distance	Hull distance
1	Horizontal	30×5m^2	0.7m	1.0m
2	Vertical	30×5m^2	2.0m	1.3m

[2] OpenCV (Open Source Computer Vision Library): https://opencv.org

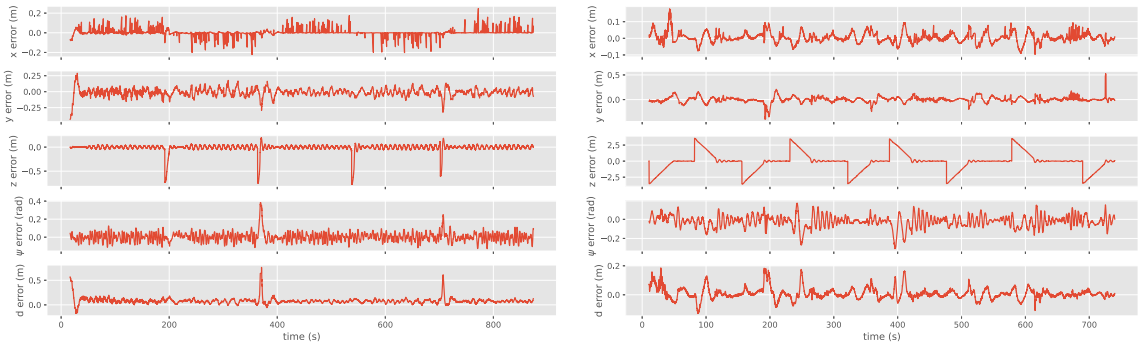


Fig. 15 Graph plots displaying the errors in position, heading, and distance to the hull, over time during two autonomous inspection. In (a), a survey with horizontal slices is performed, and with vertical slices in (b)

regarding the coverage. When following the path with horizontal lines, the sonar coverage is sparse and corresponds to each path segment. However, it does not mean the hull is not correctly visually documented. On the contrary, being able to observe and differentiate the path segments from the inspection map allows easy and intuitive observation of the camera coverage. This is also possible with the second survey but would require to additionally look at the trajectory of the ROV since the horizontal FOV of the camera and sonar are different.

In Fig. 17(b), it is possible to observe a large gap in the inspection map, which is enlargened in Fig. 18. To correct it, a request is sent to the ROV to cover this area and obtain better coverage. Once this area is visited, the section of the hull is considered as fully covered and the ROV returns to the surface to end the mission.

In both scenarios it was possible to achieve full visual coverage of the hull with full autonomy and no prior knowledge of the hull's geometry. The hull shape estimated during the mission using the sonar is close to reality and allows a better understanding of the local inspection area. Together with the online visual documentation, this provides relevant information and details to a ship inspector supervising the operation.

As mentioned earlier, the resolution of the grid needs to be chosen carefully and the parameters must be tuned to represent the hull with high enough resolution to recognise hull features while at the same time avoiding large holes due to the resolution. The first survey was used to represent the trade-off between the resolution, the minimum number of points required to create a voxel, i.e. the threshold, and detected holes, and is plotted in Fig. 19. The number of voxels quickly

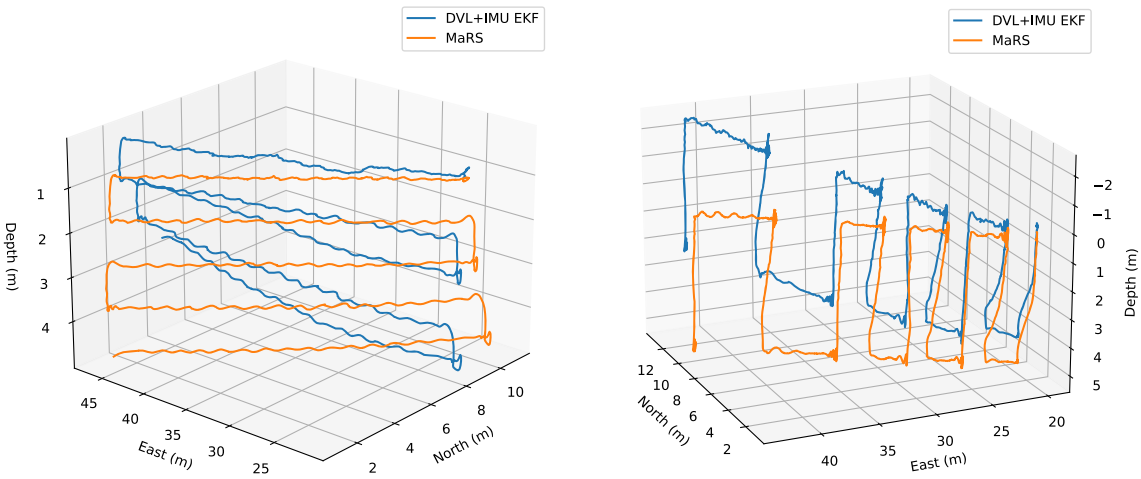


Fig. 16 3D plots of the vehicle's trajectory with two localisation strategies. In (a), a survey with horizontal slices is performed, and with vertical slices in (b)

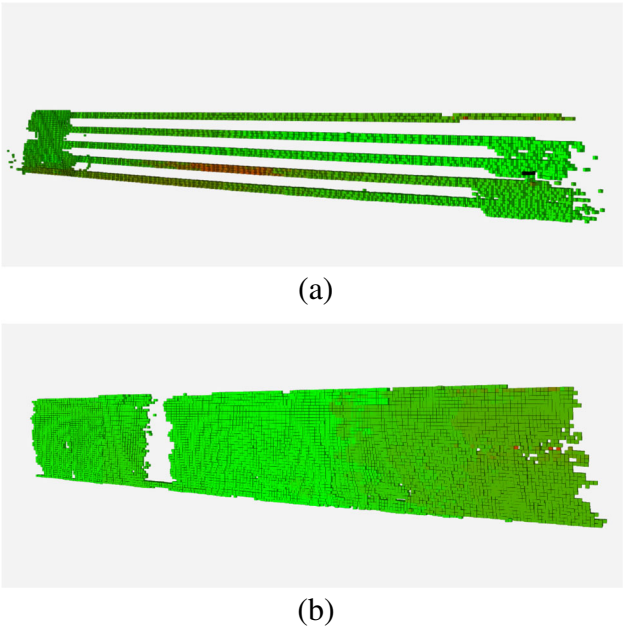


Fig. 17 Resulting 3D inspection maps from two inspection missions. They are represented as 3D voxel maps and were generated in real-time. The color correspond to the uncertainty with in green the most reliable voxels and in red the least ones. In (a), horizontal slices are considered. The mission ends with the drone going back to the water surface, top left. In (b), vertical slices are considered

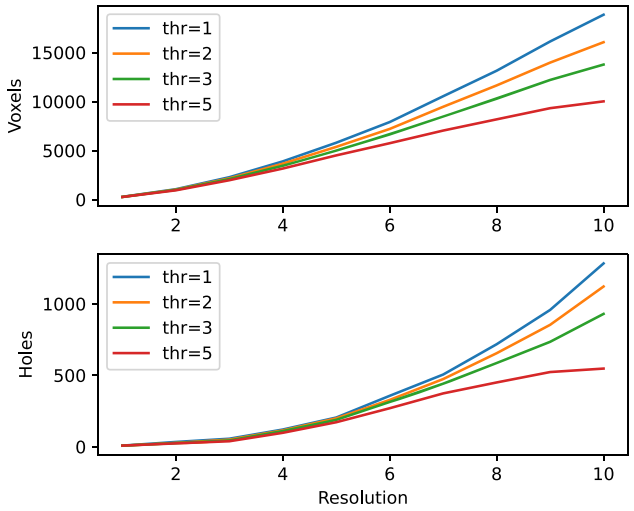


Fig. 19 The number of cells and detected coverage holes are reported after generating the 3D grid of the first survey with a set of parameters, including the resolution and threshold for the required number of sonar ranges to create a voxel

grows with the resolution, however decreases with a higher threshold since more points are required to create a voxel. This indicates the potential existence of places with sparser acoustic coverage, which are more likely to contain holes. These places typically appear at the edges of the covered areas. Therefore, a combination of high resolution with very low threshold should be avoided. However, a place with a sparse point cloud still indicates that the corresponding hull section was observed. This should be represented in the map, requiring the threshold to not be too high.

The accuracy of the manoeuvring performance during the surveys is summed up in Table 4 to provide a better understanding of the capabilities of the system. The guidance errors are represented in the table, i.e, how well the vehicle follows the path. The good navigation performance propagates to the *mapping* module, which leads to higher expected accuracy for the survey results. If the navigation errors are beyond our level of acceptance, re-calibration at the surface can be done. For both surveys, the vehicle could provide full visual coverage of the hull sections larger than 200m^2 in less than 15 minutes. The capabilities of the ROV used allow faster inspection. However, slow speed maneuvering was preferred in order to achieve a full, accurate and consistent documentation of the hull, allowing further data processing.

10.3 Complementary Case

To further demonstrate the consistency and robustness of the solution, a smaller ro-ro ferry was inspected in the Trondheim fjord, Norway. The results displayed in Fig. 20 were collected while following an inspection pattern with vertical slices of size $15 \times 2m^2$ and a required distance to the hull of $1.3m$. Similar to the previous case, the autonomous

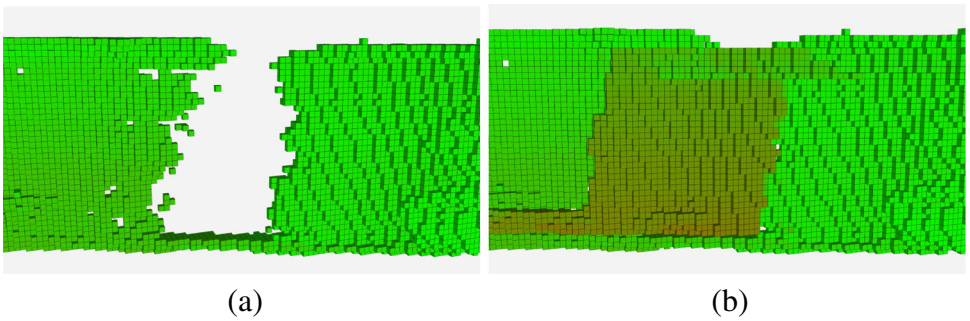


Fig. 18 Visualisation of the gap in the inspection map in (a), and how it is repaired, (b). Based on the colors of the voxels representing the uncertainty, it is possible to observe that the drone comes from the left part of the image and scan the missing area from the bottom to top

Table 4 Maneuvering and inspection performance of the surveys

Survey	Time	Surface covered	Estimate	Mean error	Std. Error
1	857s	201m^2	Position (m)	0.06	0.07
			Heading (deg)	3.39	2.88
			Distance (m)	0.05	0.05
2	729s	224m^2	Position (m)	0.09	0.10
			Heading (deg)	4.20	3.25
			Distance (m)	0.05	0.06

inspection performed well, with an observed mean position error of 0.06m, heading error of 3.11°, and distance error of 0.05m. The 3D voxel map of the inspected section of the ferry was also visually validated.

With the collected visual data, the inspector can create reports to document the state of the hull using automatically generated visual products such as mosaics. Mosaics are an effective way of combining images of areas that are of particular interest to the inspector. In spite of sea water turbidity and limited visual range, an overview of the local area can be established without loosing important visual details. A sample set of mosaics is displayed in Fig. 21. These mosaics also allow to show the local visual coverage capabilities of the solution.

11 Operation Remarks and Discussions

During the experiments, the sea condition was a challenge, especially when navigating close to the water surface. Although positioning is mainly based on acoustics at close range, which significantly limits the effects of the environmental conditions, since the orientation is based on inertial measurements, rough weather conditions can lead to numerous small and contradictory motions that can be difficult to observe and estimate, which will result in orientation precision errors. Also, the light conditions brought issues. Even though the focus of this work is not on visual image processing, the visual image acquisition is an important step. During the operations, a lot of sunlight flickering occurred, making

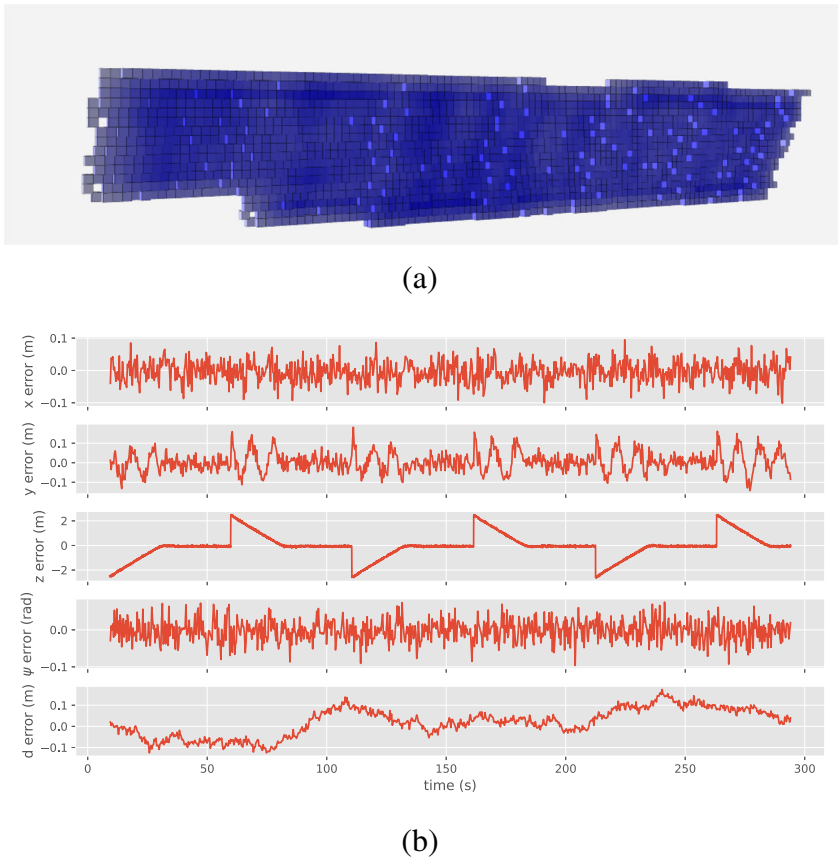


Fig. 20 The results of another autonomous inspection are displayed with in (a), the 3D voxel map and in (b), the maneuvering errors

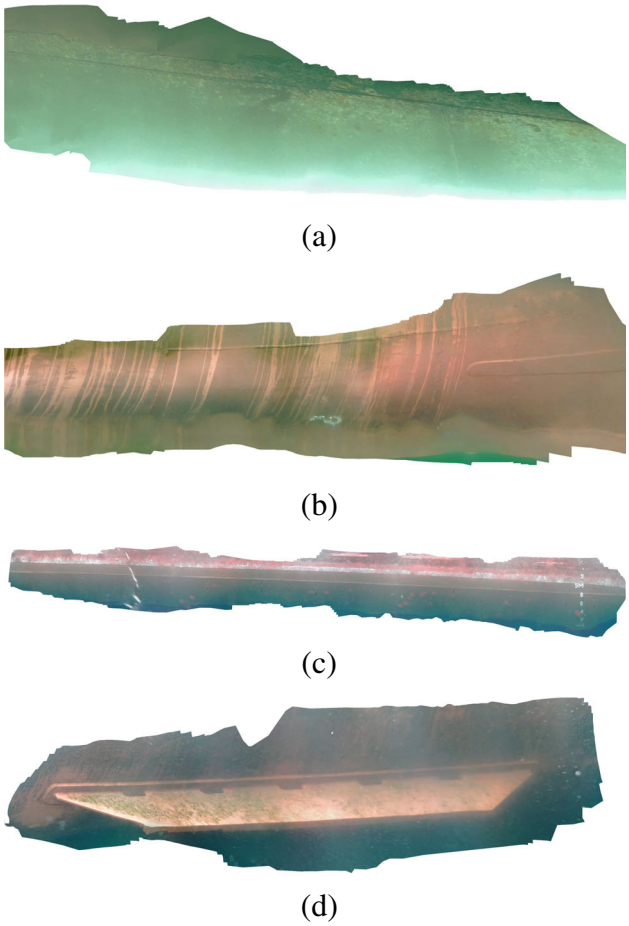


Fig. 21 Visual orthomosaics of parts of the hull are generated. They can hold relevant inspection findings which can be useful additions to the inspector reports. In (a) it is possible to observe a large amount of paint peel while in (b), the mosaic can be used to document how dirty the hull is. (c) shows the status of the hull at the waterline and (d) reports the condition of a bildge keel

difficult further processing of the images. Finally, the ROV's structure and physical constraints are not optimal for this type of tasks. Moving transversely to make sure visual documentation is enabled, also results in large drag and energy consumption. With a more adequate sensor setup, i.e., with the camera and sonar pointing sideways, the vehicle would be more stable and efficient.

The assumption of the locally flat surface when performing the inspection implies constraints. It is possible only because the sides of the ships are considered and can be visualised as a set of planes. This assumption would not hold anymore for the more complex parts of the ship, such as the propellers. Also, damaged hulls can still be considered as locally flat if a plane can be fitted in the sonar footprint. Therefore, the capture area does not need to be strictly flat for the solution to work. However, if the ship has received significant damage on a very large area which results in important deformation of the geometry, the proposed solution will not work optimally.

The proposed approach is efficient and consistent regardless of the ship type and size, and does not need further tuning for each specific ship. This is possible because the solution was designed to be adaptable and tested repeatedly on numerous vessels. Furthermore, application to aquaculture net pen inspection was explored in [46], showing the adaptability of the method.

The proposed navigation solution and procedures will work in high turbidity water condition because it does not rely on optic sensors for the real-time operation. Opposed to previous approaches [6–8, 47], the proposed solution provides and guarantee full visual documentation in addition to acoustic coverage. Earlier reported solutions have not defined full visual coverage as an objective for the inspection mission. The focus was on the methods rather than the inspection results, leading to a different set of objectives. Although local visual maps are generated as a mean to show the direct results of the developed methods, and that full coverage remains technically feasible, it is not discussed and therefore not addressed. A complete visual coverage is mandatory for a thorough documentation and the detection and registration of faults such as marine growth, paint peel, and corrosion as indicated by the American Bureau of Shipping[3] (ABS). Coverage monitoring should be done in real-time, and dynamically taken into account by a mission manager to obtain a resilient and robust behavior of the vehicle as shown in this work. Furthermore, the proposed setup is less expensive and smaller, which makes it more accessible, easy to deploy and manipulate, hence, the overall operation runs smoother. However, the proposed solution works best if the ship inspection is divided in sections that are individually inspected. This is because of the localisation drift over time in the horizontal plane. The vehicle cannot perform the inspection as it should when the position drifts. However, it can keep running at correct depth, with correct heading and with correct distance to the hull. Hence the image quality and visual coverage will remain acceptable, while the integrity of the coverage mapping will deteriorate with the navigation drift. A comparison of the existing ship hull inspection solution is proposed in Table 5, comparing the setups, objectives, and the type of results. It highlights the proposed approach in this paper as an alternative solution with complementary results.

The inspection map will show which areas were inspected, and it can be used as a support for documentation while the inspector is looking at the video feed. This enables compliance with the guidelines of the ABS for remote operations which require real-time streaming of digital data to help the assessment of the structure integrity while identifying

[3] ABS: https://ww2.eagle.org

Table 5 Comparison of the solutions

Approach	Setup	Objectives	Results
Vaganay et al. [5]	AUV + DVL + imaging sonar	Autonomous navigation + 100% acoustic coverage	DVL-based hull-relative multi-level control system + post-processed coverage area
Hover et al. [7]	AUV (optional tether) + DVL + camera + imaging sonar (with profiling lens)	Autonomous navigation + limited drift + 3D modelling	DVL-based hull-relative control + camera-sonar combination for drift correction + post-processed 3D model
Kim and Eustice [47]	AUV (optional tether) + camera	Monocular SLAM	Robust real-time localisation with geometrical priors + long term operations
Hong et al. [8]	AUV (optional tether) + DVL + stereo camera + acoustic altimeter	Autonomous navigation + stereo SLAM	DVL-based localisation with manually-aided control system + hull-relative visual pose estimation + visual mosaics
Ours	ROV + DVL + camera + imaging sonar	Autonomous navigation + 100% visual coverage + real-time monitoring	Sonar-based hull-relative control + online generated acoustic inspection map + automatically generated visual mosaics

possible anomalies. Furthermore, the identification of the dimensions of the anomalies is strongly recommended, and using the camera only, the scale can be difficult to estimate underwater. Using the inspection map at the same time provides a meaningful scale to the ROV observations.

12 Conclusion and Future Work

In this paper, a new approach to real-time underwater ship hull inspection was studied. An autonomous maneuvering solution was proposed to efficiently navigate around a ship while keeping track of the state of the hull, i.e., its position and orientation, and condition. The sonar scans are processed in a novel way for that purpose, to not only obtain the drone's orientation relative to the hull, but also to generate an estimate of the shape of the hull in the form of an inspection map. The proposed methods allowed to provide the operator/inspector with a better understanding of the inspection procedure. This includes uncertainty estimation of the coverage and missing area detection that can indicate the quality of the inspection locally and overall. The results were successfully demonstrated in real field experiments with repeated inspections, to demonstrate the viability and robustness of the solution. Additionally, they proved it is possible to achieve high autonomy with a simplistic sensor setup and drone configuration. The ROV efficiently adapted to the shape of the hull in real-time, which was estimated with a level of accuracy high enough to recognize places. We believe the proposed methods presented in this paper contribute to the development of the basis for an end-to-end autonomous solution for ship hull inspection.

Future works include developing methods towards more real-time operation such as online reconstruction of the model using cloud-based techniques [48]. Adding in the loop the detection of faults on ship hull using previous work by [49] would allow to go a step further in the end-to-end autonomy of ship hull inspection. Expanding the autonomy to the inspection of more parts of the ship such as propellers [50] would enable to obtain a complete inspection map. Further work on the sonar is also considered, including intrinsic calibration to estimate biases relative to bearings as well as intrinsic calibration for better synchronisation with the vehicle odometry. This would increase greatly the accuracy of the sonar points. Also, we believe the gamma distribution as used in this paper hold geometric information and descriptions of what the sonar sees. Further analysis in that direction could results in better scene understanding using acoustic images. The application of machine leaning for autonomous control will also be explored. Previous works [51, 52] showed great potential, including in underwater inspection scenarios [12]. Finally, since most inspections occur in harbors, the use of external sensors installed within the inspection area to extend the inspection efficiency and capabilities will be explored.

Acknowledgements The authors would like to thank the collaborators within the BugWright2 projects as well as Borja Serra from Blueye Robotics for their continuous technical support within this project.

Author Contributions The first author (A. Cardaillac) performed the research and experiments. The second and third authors contributed to the study conception and design (R. Skjetne and M. Ludvigsen).

Funding Open access funding provided by NTNU Norwegian University of Science and Technology (incl St. Olavs Hospital - Trondheim University Hospital). This work was supported by the BugWright2 EU H2020-Project [Grant No. 871260]; and by the Research Council of

Norway through the Centre of Excellence NTNU AMOS [Grant No. 223254].

Data Availability The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

Declarations

Conflicts of interest The authors have no conflict of interest to declare.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

1. Hedgpeth, J.W.: Marine fouling and its prevention. Science **118**(3061), 257–257 (1953). https://doi.org/10.1126/science.118.3061.257.a
2. Boon, B., Brennan, F., Garbatov, Y., Ji, C., Parunov, J., Rahman, T., Rizzo, C., Rouhan, A., Shin, C., Yamamoto, N.: Condition assessment of aged ships and offshore structures. In: International Ship and Offshore Structures Congress, **2**, 313–365 (2009)
3. Mittleman, J., Swan, L.: Underwater inspection for welding and overhaul. Naval Eng. J. **105**(5), 37–42 (1993). https://doi.org/10.1111/j.1559-3584.1993.tb02755.x
4. Lynn, D.C., Bohlander, G.S.: Performing ship hull inspections using a remotely operated vehicle. In: Oceans '99. MTS/IEEE. Riding the Crest Into the 21st Century. Conference and Exhibition. Conference Proceedings (IEEE Cat. No.99CH37008), **2**, pp. 555–5622 (1999). https://doi.org/10.1109/OCEANS.1999.804763
5. Vaganay, J., Elkins, M.L., Willcox, S., Hover, F.S., Damus, R.S., Desset, S., Morash, J.P., Polidoro, V.C.: Ship hull inspection by hull-relative navigation and control. In: Proceedings of OCEANS 2005 MTS/IEEE, pp. 761–7661 (2005). https://doi.org/10.1109/OCEANS.2005.1639844
6. Kaess, M., Johannsson, H., Englot, B., Hover, F.S., Leonard, J.J.: Towards autonomous ship hull inspection using the bluefin hauv. In: 9th International Symposium on Technology and the Mine Problem. Conference Proceedings, pp. 1–10 (2010)
7. Hover, F.S., Eustice, R.M., Kim, A., Englot, B., Johannsson, H., Kaess, M., Leonard, J.J.: Advanced perception, navigation and planning for autonomous in-water ship hull inspection. Int. J. Robotics Res. **31**(12), 1445–1464 (2012). https://doi.org/10.1177/0278364912461059
8. Hong, S., Chung, D., Kim, J., Kim, Y., Kim, A., Yoon, H.K.: In-water visual ship hull inspection using a hover-capable underwater vehicle with stereo vision. J Field Robotics **36**(3), 531–546 (2019). https://doi.org/10.1002/rob.21841
9. Kazmi, W., Ridao, P., Ribas, D., Hernandez, E.: Dam wall detection and tracking using a mechanically scanned imaging sonar. In: 2009 IEEE International Conference on Robotics and Automation, pp. 3595–3600 (2009). https://doi.org/10.1109/ROBOT.2009.5152691
10. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981). https://doi.org/10.1145/358669.358692
11. Karras, G.C., Bechlioulis, C.P., Abdella, H.K., Larkworthy, T., Kyriakopoulos, K., Lane D.: A robust sonar servo control scheme for wall-following using an autonomous underwater vehicle. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3893–3898 (2013). https://doi.org/10.1109/IROS.2013.6696913
12. Wang, X., Zhang, G., Sun, Y., Wan, L., Cao, J.: Research on autonomous underwater vehicle wall following based on reinforcement learning and multi-sonar weighted round robin mode. Int. J. Adv. Robotic Syst. **17**(3) (2020). https://doi.org/10.1177/1729881420925311
13. Fossen T.: Handbook of Marine Craft Hydrodynamics and Motion Control, pp. 331–387. John Wiley & Sons, Ltd (2011). Chap. 12. https://doi.org/10.1002/9781119994138
14. Amundsen, H.B., Caharija, W., Pettersen, K.Y.: Autonomous rov inspections of aquaculture net pens using dvl. IEEE J. Oceanic Eng. **47**(1), 1–19 (2022). https://doi.org/10.1109/JOE.2021.3105285
15. Arnesen, B.O., Lekkas, A.M., Schjølberg I.: 3D Path Following and Tracking for an Inspection Class ROV. International Conference on Offshore Mechanics and Arctic Engineering, vol. Volume 7A: Ocean Engineering, pp. 07–06019 (2017). https://doi.org/10.1115/OMAE2017-61170
16. Galceran, E., Palomeras, N., Carreras, M.: Profile following for inspection of underwater structures. Paladyn, J. Behavioral Robotics **4**(4), 211–222 (2013) https://doi.org/10.2478/pjbr-2013-0019
17. Nguyen, V.S., Trinh, T.H., Tran, M.H.: Hole boundary detection of a surface of 3d point clouds. In: 2015 International Conference on Advanced Computing and Applications (ACOMP), pp. 124–129 (2015). https://doi.org/10.1109/ACOMP.2015.12
18. Nguyen, V.-S., Bac, A., Daniel, M.: Boundary extraction and simplification of a surface defined by a sparse 3d volume. In: Proceedings of the Third Symposium on Information and Communication Technology. SoICT '12, pp. 115–124. Association for Computing Machinery, New York, NY, USA (2012). https://doi.org/10.1145/2350716.2350735
19. Gai, S., Da, F., Zeng, L., Huang, Y.: Research on a hole filling algorithm of a point cloud based on structure from motion. J. Opt. Soc. Am. A **36**(2), 39–46 (2019). https://doi.org/10.1364/JOSAA.36.000A39
20. Fiolka, T., Rouatbi, F., Bender, D.: Automated Detection and Closing of Holes in Aerial Point Clouds Using AN Uas. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences **42W6**, 101–107 (2017) https://doi.org/10.5194/isprs-archives-XLII-2-W6-101-2017
21. Cardaillac, A., Ludvigsen, M.: Ruled path planning framework for safe and dynamic navigation. In: OCEANS 2021: San Diego - Porto, pp. 1–7 (2021). https://doi.org/10.23919/OCEANS44145.2021.9705699
22. Scheiber, M., Cardaillac, A., Brommer, C., Weiss, S., Ludvigsen, M.: Modular multi-sensor fusion for underwater localization for autonomous rov operations. In: OCEANS 2022, Hampton

Roads, pp. 1–5 (2022). https://doi.org/10.1109/OCEANS47191.2022.9977298
23. Cardaillac, A., Ludvigsen, M.: Path following for underwater inspection allowing manoeuvring constraints. In: Petrovic I., Menegatti, E., Marković, I. (eds.) Intelligent Autonomous Systems 17, pp. 867–880. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-22216-0_58
24. Maurelli, F., Krupiński, S., Xiang, X., Petillot, Y.: Auv localisation: a review of passive and active techniques. Int. J. Intell. Robotics Appl. **6**(2), 246–269 (2022). https://doi.org/10.1007/s41315-021-00215-x
25. Gómez-Espinosa, A., Cuan-Urquizo, E., González-García, J.: Autonomous underwater vehicles: Localization, navigation, and communication for collaborative missions. Appl. Sci. **10**, 1256 (2020). https://doi.org/10.3390/app10041256
26. Brommer, C., Jung, R., Steinbrener, J., Weiss, S.: Mars: A modular and robust sensor-fusion framework. IEEE Robotics Automation Lett. **6**(2), 359–366 (2021). https://doi.org/10.1109/LRA.2020.3043195
27. LaValle, S.M.: Rapidly-exploring random trees: a new tool for path planning. Technical report, Computer Science Dept, Iowa State University (1998)
28. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numer. Math. **1**(1), 269–271 (1959). https://doi.org/10.1007/BF01386390
29. Koenig, S., Likhachev, M.: Fast replanning for navigation in unknown terrain. IEEE Trans. Robotics **21**(3), 354–363 (2005). https://doi.org/10.1109/TRO.2004.838026
30. Nguyen, V., Martinelli, A., Tomatis, N., Siegwart, R.: A comparison of line extraction algorithms using 2d laser rangefinder for indoor mobile robotics. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1929–1934 (2005). https://doi.org/10.1109/IROS.2005.1545234
31. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Machine Intell. **PAMI-8**(6), 679–698 (1986) https://doi.org/10.1109/TPAMI.1986.4767851
32. Davies, E.R.: Chapter 1 - the dramatically changing face of computer vision. In: Davies E.R., Turk M.A. (eds.) Advanced Methods and Deep Learning in Computer Vision. Computer Vision and Pattern Recognition, pp. 1–91. Academic Press, UK (2022). https://doi.org/10.1016/B978-0-12-822109-9.00010-2
33. Breivik, M., Fossen, T.I.: Principles of guidance-based path following in 2d and 3d. In: Proceedings of the 44th IEEE Conference on Decision and Control, pp. 627–634 (2005). https://doi.org/10.1109/CDC.2005.1582226
34. Breivik, M., Fossen, T.I.: Guidance laws for autonomous underwater vehicles. In: Inzartsev A.V. (ed.) Underwater Vehicles, pp. 51–76. IntechOpen, Rijeka (2009). Chap. 4. https://doi.org/10.5772/6696
35. Lekkas, A.M., Fossen, T.I.: A time-varying lookahead distance guidance law for path following. IFAC Proceedings Volumes, 9th IFAC Conference on Manoeuvring and Control of Marine Craft. **45**(27), 398–403 (2012) https://doi.org/10.3182/20120919-3-IT-2046.00068
36. Lekkas, A.M., Fossen, T.I.: Integral los path following for curved paths based on a monotone cubic hermite spline parametrization. IEEE Trans. Control Syst. Technol. **22**(6), 2287–2301 (2014). https://doi.org/10.1109/TCST.2014.2306774
37. Skjetne, R.: The maneuvering problem. PhD thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway (2005)
38. Hauser, J., Hindman, R.: Maneuver regulation from trajectory tracking: Feedback linearizable systems*. IFAC Proceedings Volumes, 3rd IFAC Symposium on Nonlinear Control Systems Design 1995, Tahoe City, CA, USA, 25-28 June 1995. **28**(14), 595–600 (1995). https://doi.org/10.1016/S1474-6670(17)46893-5
39. Sørensen, M.E.N., Breivik, M., Skjetne, R.: Comparing combinations of linear and nonlinear feedback terms for ship motion control. IEEE Access **8**, 193813–193826 (2020). https://doi.org/10.1109/ACCESS.2020.3033409
40. Breivik, M., Strand, J., Fossen, T.: Guided dynamic positioning for fully actuated marine surface vessels. In: 7th IFAC Conference on Manoeuvring and Control of Marine Craft, pp. 1–6 (2006)
41. Dia, R., Mottin, J., Rakotovao, T., Puschini, D., Lesecq, S.: Evaluation of occupancy grid resolution through a novel approach for inverse sensor modeling. IFAC-PapersOnLine, 20th IFAC World Congress. **50**(1), 13841–13847 (2017) https://doi.org/10.1016/j.ifacol.2017.08.2225
42. Cheng, C., Wang, C., Yang, D., Liu, W., Zhang, F.: Underwater localization and mapping based on multi-beam forward looking sonar. Frontiers in Neurorobotics **15** (2022). https://doi.org/10.3389/fnbot.2021.801956
43. Paindaveine, D.: A canonical definition of shape. Stat. & Probability Lett. **78**(14), 2240–2247 (2008). https://doi.org/10.1016/j.spl.2008.01.094
44. Tyler, D.E.: A Distribution-Free M-Estimator of Multivariate Scatter. Ann. Stat. **15**(1), 234–251 (1987). https://doi.org/10.1214/aos/1176350263
45. Dümbgen, L.: On tyler's m-functional of scatter in high dimension. Ann. Institute Stat. Math. **50**(3), 471–491 (1998). https://doi.org/10.1023/A:1003573311481
46. Cardaillac, A., Amundsen, H.B., Kelasidi, E., Ludvigsen, M.: Application of maneuvering based control for autonomous inspection of aquaculture net pens. In: 2023 8th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS), pp. 44–51 (2023). https://doi.org/10.1109/ACIRS58671.2023.10239708
47. Kim, A., Eustice, R.M.: Real-time visual slam for autonomous underwater hull inspection using visual saliency. IEEE Trans. Robotics **29**(3), 719–733 (2013). https://doi.org/10.1109/TRO.2012.2235699
48. Cardaillac, A., Ludvigsen, M.: A communication interface for multilayer cloud computing architecture for low cost underwater vehicles*. IFAC-PapersOnLine, 11th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2022. **55**(14), 77–82 (2022). https://doi.org/10.1016/j.ifacol.2022.07.586
49. Waszak, M., Cardaillac, A., Elvesæter, B., Rødølen, F., Ludvigsen, M.: Semantic segmentation in underwater ship inspections: Benchmark and data set. IEEE J. Oceanic Eng. 1–12 (2022) https://doi.org/10.1109/JOE.2022.3219129
50. Cardaillac, A., Ludvigsen, M.: Camera-sonar combination for improved underwater localization and mapping. IEEE Access **11**, 123070–123079 (2023). https://doi.org/10.1109/ACCESS.2023.3329834
51. Lu, W., Cheng, K., Hu, M.: Reinforcement learning for autonomous underwater vehicles via data-informed domain randomization. Appl. Sci. **13**(3) (2023). https://doi.org/10.3390/app13031723
52. Hadi, B., Khosravi, A., Sarhadi, P.: Deep reinforcement learning for adaptive path planning and control of an autonomous underwater vehicle. Appl. Ocean Res. **129**, 103326 (2022). https://doi.org/10.1016/j.apor.2022.103326

Alexandre Cardaillac received his MSc degree in Artificial Intelligence with Speech and Multimodal Interaction from the Heriot-Watt University in 2020 and received the award of the best MSc dissertation for his work on uncertainty estimation in deep neural networks. He completed his PhD degree in engineering with the Department of Marine Technology in 2024 at the Norwegian University of Science and Technology, as part of the Applied Underwater Robotics Laboratory. Alexandre is currently a Postdoctoral Research Associate in Underwater Computational Imaging at the University of Sydney. He is part of the ARIAM Research Hub and the Robotic Imaging Lab. His main research interests include underwater computational imaging using acoustic and optical data, scene understanding and situation awareness for underwater robotic systems.

Roger Skjetne received his MSc degree in control engineering in 2000 from the University of California at Santa Barbara (UCSB) and corresponding PhD degree in 2005 from the Norwegian University of Science and Technology (NTNU). He holds an Exxon Mobil prize for his PhD thesis at NTNU. Prior to his studies, he worked as a certified electrician for Aker Elektro AS on numerous offshore installations for the North Sea, and in 2004-2009 he was employed in Marine Cybernetics AS, working on Hardware-In-the-Loop simulation for testing marine control systems. From August 2009 he has held the Kongsberg Maritime chair of Professor in Control Engineering at the Department of Marine Technology at NTNU. In 2017-2018, he was a visiting research scholar at the Center for Control, Dynamical-systems and Computation at the University of California at Santa Barbara. His research interests are within autonomous ship guidance and control, dynamic positioning of marine vessels, control and optimization of shipboard hybrid-electric and fully-electric power systems, Arctic ship operations, and nonlinear and adaptive control of marine vessels in general.

Martin Ludvigsen has been a Professor at the Department of Marine Technology, NTNU since 2014. His research interests are within the field of underwater vehicles and its applications. This includes perception and interpretation of cameras and sonar data together with autonomy. Underwater vehicles for marine observation systems, adaptive mission planning for one or more vehicles for ocean column mapping has also been a focus point for his research group. He is co-founder and manager for the Applied Underwater Laboratory (AUR-Lab) at NTNU, Trondheim, Norway. AUR-Lab (https://www.ntnu.edu/web/aur-lab/aur-lab) is an asset for multidisciplinary marine research at NTNU, facilitating research within both engineering disciplines and marine science by providing ROV, AUV and USV operations. He is also the co-founder of Blueye Robotics, a portable ROV start-up. Ludvigsen has extensive at-sea experience in Arctic waters as well as in benthic environments associated with the Norwegian mid-ocean ridge. He has been involved in research projects both on deep sea research, in the upper water column and in the arctic deploying robotic underwater vehicles.