2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Uncertainty-based Online Mapping and Motion Planning for Marine Robotics Guidance

Èric Pairet[1], Juan David Hernández[2], Morteza Lahijanian[3], and Marc Carreras[4]

Abstract— **In real-world robotics, motion planning remains to be an open challenge. Not only robotic systems are required to move through unexplored environments, but also their manoeuvrability is constrained by their dynamics and often suffer from uncertainty. One approach to overcome this problem is to incrementally map the surroundings while, simultaneously, planning a safe and feasible path to a desired goal. This is especially critical in underwater environments, where autonomous vehicles must deal with both motion and environment uncertainties. In order to cope with these constraints, this work proposes an uncertainty-based framework for mapping and planning feasible motions online with probabilistic safety-guarantees. The proposed approach deals with the motion, probabilistic safety, and online computation constraints by (i) incrementally representing the environment as a collection of local maps, and (ii) iteratively (re)planning kinodynamically-feasible and probabilistically-safe paths to goal. The proposed framework is evaluated on the Sparus II, a nonholonomic torpedo-shaped AUV, by conducting simulated and real-world trials, thus proving the efficacy of the method and its suitability even for systems with limited on-board computational power.**

I. INTRODUCTION

The usage of autonomous underwater vehicles (AUVs) has considerably increased in the last decades for scientific, military, and commercial purposes [1]. Some of the most outstanding applications are in-water ship hull inspections [2], underwater archaeology [3], seabed exploration [4], and exploration of inaccessible areas to vessel-based instruments [5]. A common issue in all these applications is the lack of prior information about the environment in which AUVs have to operate. On top of that, underwater vehicles suffer from motion uncertainty and, in most cases, limited manoeuvrability. Thus, addressing the motion constraints, and the uncertainties on the vehicle's motion and the environment awareness jointly in a planner is essential to guarantee the vehicle's safety. This is a challenging task because such a planner must be able to continuously modify the vehicle's motion plan towards a desired goal according to the incremental environment awareness, while maintaining safety guarantees in face of the uncertainties.

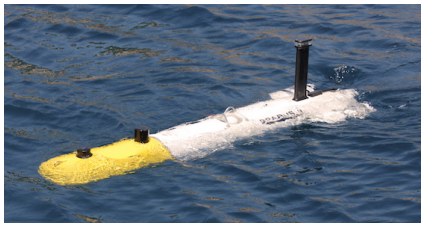


Fig. 1: Sparus II AUV, a nonholonomic vehicle.

In order to deal with kinodynamic motion constraints, sampling-based motion planners offer great capabilities (e.g., [6]–[8]). At the same time, these planners also bring the chance to deal with motion uncertainty, for which there are generally two approaches. One approach that is popular among existing planners is based on discrete Markov modelling of the evolution of the system in the environment and generating a policy over the approximated Markov states. Examples of such motion planners include stochastic motion roadmap (SMR) [9] and incremental Markov decision process (iMDP) [10]. These methods have shown to be effective and provide optimality guarantees in terms of probability of reaching a desired goal; however, they assume perfect knowledge about the environment. Works such as [11] have extended these techniques to partially unknown environments. Nonetheless, their large computational times remain to be the main hurdle in applications with fully-unknown environments or online planning requirements.

Another approach to deal with motion uncertainty in planning is based on chance-constraint method, which focuses on finding a motion plan that satisfies a given (minimum) safety probability bound. The challenge lies in the computation of the safety probability over plans. In [12], linear chance constraints are combined with disjunctive linear programming to perform probabilistic convex obstacle avoidance. This concept was extended and integrated into a sampling-based planner, leading to the chance constrained RRT (CC-RRT) [13] and the CC-RRT* [14]. The advantage of these methods is that satisfying plans can be computed quickly, making them desirable for online applications. They are, however, built on strong assumptions and rely on the prior knowledge of a convex environment with convex obstacles.

This research project has been conducted in the Computer Vision and Robotics Institute (VICOROB) at University of Girona (Spain) with the support of the EXCELLABUST and ARCHROV projects under the Grant agreements H2020-TWINN-2015, CSA, ID: 691980 and DPI2014-57746-C3-3-R, respectively. Additionally, this work has been partially supported by ORCA Hub EPSRC (EP/R026173/1) and consortium partners.

[1]È. Pairet is with VICOROB at University of Girona (Spain) and is currently with the Edinburgh Centre for Robotics at University of Edinburgh and Heriot-Watt University (UK). eric.pairet@ed.ac.uk

[2]J. D. Hernández is with VICOROB at University of Girona (Spain) and is currently with the Department of Computer Science at Rice University, Houston (TX, USA). His work while at Rice was supported in part by NSF IIS 1317849. juandhv@rice.edu

[3]M. Lahijanian was with the Department of Computer Science at University of Oxford (UK) and is currently with the Aerospace Engineering Sciences Department at University of Colorado Boulder (CO, USA). His work while at Oxford was supported in part by EPSRC Mobile Autonomy Grant (EP/M019918/1). morteza.lahijanian@colorado.edu

[4]M. Carreras is with the VICOROB at University of Girona (Spain). marc.carreras@udg.edu

In the field of underwater robotics, online computation techniques have been studied to address the problem of planning in unknown environments, e.g., [15]–[18]. Petillot et al. [15] tracked nearby underwater objects by using a multi-beam forward-looking sonar to then perform online obstacle avoidance. Along in this line, Maki et al. [16] proposed a method to conduct online planning by guiding an AUV with landmarks. These methods do not provide any theoretical performance or safety guarantees, and their evaluations are conducted either in simulation or in a highly controlled environment. Hence, their performance in realistic scenarios is yet to be tested. More recently, Hernández et al. [17], [18] presented an online framework to plan paths under motion constraints for AUVs. Notwithstanding their framework succeeded in solving start-to-goal queries in unexplored real-world environments, their planner uses ad-hoc heuristics to estimate the risk associated with the solution path, and approximates the system's dynamics with the Dubins curves. Consequently, that framework also lacks theoretical analysis and does not provide a measure of robustness or quantified safety guarantees.

This paper presents a new framework that seeks to overcome some of the limitations of the preceding works, namely, the lack of exploitation of the whole system's dynamic range and the use of ad-hoc heuristics to ensure the robot's safety [17], [18], and the conservativeness and the need of having a prior convex representation of the environment [12], [13]. Aiming to guarantee the AUV's safety when solving start-to-goal queries in unexplored environments, the proposed framework incrementally maps and plans paths online, while considering the vehicle's kinodynamic constraints, uncertainties in its motion and mapping, and a user-defined minimum probability of safety. The framework is twofold: (i) incrementally mapping the vehicle's surroundings to build an uncertain environment representation, and (ii) planning feasible paths (according to the AUV's motion constraints) with probabilistic safety guarantees (according to the uncertainty of the system's motion and environment awareness).

The main contribution of this work is an end-to-end framework that allows AUVs to navigate online in unknown environments with safety guarantees by means of online mapping, online motion planning, and efficient evaluation of uncertainties. The efficacy of this method has been demonstrated in both simulated and real-world scenarios on the Sparus II (Fig. 1), a nonholonomic torpedo-shaped AUV. The experimental results demonstrate the suitability of the method to address the aforementioned challenges.

II. SYSTEM DESCRIPTION AND PROBLEM FORMULATION

The aim of this paper is a framework that allows autonomous navigation of AUVs in unexplored environments with safety guarantees. Such a framework needs to be able to simultaneously map the environment and plan paths towards a desired goal, while accounting for the motion and mapping uncertainties. These requirements are jointly addressed by formulating a motion planning problem in the belief space. That is, given an instance of an uncertain map that is incrementally being constructed, find a set of feasible motions according to the system's dynamics, drive the system to the desired goal, and satisfy a minimum probability of safety in every point in time. Bearing the problem requirements in mind, this section firstly describes the nonholonomic robotic system and its associated uncertainty. It then explains the construction of the uncertain map before detailing the computation of the probability of collision between these two uncertain agents. Finally, this section provides a formal definition of the motion planning problem.

A. Motion Constraints and Uncertainty

In its most generic formulation, the motion model of a nonholonomic system is defined as:

$$\dot{\mathbf{x}} = f(\mathbf{x},\ \mathbf{u}), \tag{1}$$

where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ is the state, $\mathbf{u} \in \mathcal{U} \subseteq \mathbb{R}^{n_u}$ is the control input, and f is a smooth differentiable function. Although in some cases the uncertainty associated with the motion is not considered, more elaborated formulations that include the uncertainty allow to better control the system. In the particular case of a nonholonomic AUV, such an uncertainty can be estimated as additive Gaussian noise in the linearized, discrete-time form of (1). That is, by assuming that each choice of control $\mathbf{u}$ is applied for a small or multiple durations of Δt, the uncertain motion can be represented by:

$$\mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k + G\mathbf{w}_k, \tag{2}$$

where subscript k represents time $t = k\Delta t$, $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $G \in \mathbb{R}^{n_x \times n_w}$, and $\mathbf{w} \in \mathbb{R}^{n_w}$ is a random variable with normal distribution defined by mean zero and covariance $P_{\mathbf{w}}$, i.e., $\mathbf{w} \sim \mathcal{N}(0,\ P_{\mathbf{w}})$.

For the particular case of a torpedo-shaped AUV that operates at a constant depth, i.e. in a two-dimensional (2D) workspace $\mathcal{W} = \mathbb{R}^2$, the vehicle's motion model can be approximated by a unicycle system:

$$\dot{x} = v\ \cos(\theta), \quad \dot{y} = v\ \sin(\theta), \quad \dot{\theta} = \omega, \tag{3}$$

where x and y correspond to the Cartesian coordinates of the system with respect to a predefined reference frame, θ is the system's orientation with respect to the x-axis, v is the vehicle's forward velocity, and ω is the vehicle's turning rate. Thus, the system's state is defined as $\mathbf{x} = (x,\ y,\ \theta)^T$, and the system's control input is defined as $\mathbf{u} = (v,\ \omega)^T$.

Although (3) approximates the AUVs' behaviour, it does not consider any source of uncertainty. In coping with this limitation, (3) is first linearised by using a dynamic feedback linearisation controller as presented in [19]. This technique (i) increases the order of the system and defines its state as $\mathbf{z} = (x,\ \dot{x},\ y,\ \dot{y})^T$, and (ii) applies a proportional derivative (PD) controller on the model to drive the system towards a desired state $\mathbf{r} = (x_r,\ \dot{x}_r,\ y_r,\ \dot{y}_r)^T$. Then, the differences between the real system and the discretised closed-loop model can be approximated by a Gaussian distribution, and the closed-loop system can be represented as in (2) with

states $\mathbf{z} \in \mathcal{X} = \mathbb{R}^{n_z}$, and controls $\mathbf{r} \in \mathcal{U} = \mathcal{X}$. Thus, state $\mathbf{z}_k$ is best described by its probability distribution:

$$\mathbf{z}_k \sim b_k = \mathcal{N}(\hat{\mathbf{z}}_k,\ P_{\mathbf{z}_k}), \tag{4}$$

where b is referred to as the *belief* of $\mathbf{z}$, and $\hat{\mathbf{z}} \in \mathbb{R}^{n_z}$ and $P_{\mathbf{z}} \in \mathbb{R}^{n_z \times n_z}$ are the mean and covariance of b. The set of all beliefs is called the belief space and denoted by $\mathcal{B}$. The evolution of belief is then given by the independent propagation of its mean and covariance as:

$$\begin{aligned} \hat{\mathbf{z}}_{k+1} &= A\hat{\mathbf{z}}_k + B\mathbf{r}_k, \\ P_{\mathbf{z}_{k+1}} &= AP_{\mathbf{z}_k}A^T + P_{\mathbf{w}}, \end{aligned} \tag{5}$$

where $A \in \mathbb{R}^{n_z \times n_z}$ and $B \in \mathbb{R}^{n_z \times n_z}$ define the closed-loop linearised equation of motion with the PD controller, and $P_{\mathbf{w}}$ is the covariance of noise $\mathbf{w}$ as in (2).

B. Environment Uncertainty

Some applications in robotics, such the ones in the underwater domain, lack a complete awareness of the environment, either because there is no information of the surroundings or because of the presence of dynamic elements in the workspace. This work scopes the mapping requirements to undiscovered static environments. In order to reveal the obstacles in the environment, the robot is equipped with a range-based perception sensor. This setup allows the AUV to incrementally explore the surroundings as it moves, i.e. observe and integrate into the map the obstacles when they are inside the sensor's detection range.

The range-based sensor uniquely detects points on the boundary of a nearby obstacle. Assuming no uncertainty on these observations, yet bearing in mind that the robot's location is uncertain with respect to the global frame, the observed point is represented in the global frame as:

$$\mathbf{x}_O \sim \mathcal{N}(\hat{\mathbf{z}}_k \oplus \mathbf{d}_{\mathbf{x}_O},\ P_{\mathbf{z}_k}), \tag{6}$$

where $\mathbf{d}_{\mathbf{x}_O}$ is the detected obstacle point in the robot's local frame which is projected to the global frame with the composition function $\oplus$. From these uncertain points $\mathbf{x}_O$, the robot constructs an uncertain map consisting of the obstacle space and the free space, i.e. $\mathcal{M} = \mathcal{X}_O \cup \mathcal{X}_F$. The obstacle density function for point $\mathbf{x} \in \mathcal{X}$ denoted by $F_O(\mathbf{x})$ is then the sum of the normally distributed densities for each detected obstacle, which itself is a normal distribution.

C. Probabilistic Safety Guarantee

The system's and environment's uncertainties are jointly considered to guarantee the vehicle's safety. More specifically, the probability of the system being in collision with an obstacle in the environment at time k is characterised as:

$$\begin{aligned} p_{\text{collision}}(b_k,\ \mathcal{M}) &\leq \int_{\mathcal{X}} b_k(\mathbf{x})\, F_O(\mathbf{x})\, d\mathbf{x} \\ &= \int_{\mathcal{X}} \mathcal{N}(\mathbf{x} \,|\, \hat{\mathbf{z}}_k,\ P_{\mathbf{z}_k})\, F_O(\mathbf{x})\, d\mathbf{x}. \end{aligned} \tag{7}$$

Given a minimum probability of safety p_{safe}, we require $1 - p_{\text{collision}}(b,\ \mathcal{M}) > p_{\text{safe}}$ for every belief b on the path in order to probabilistically guarantee the robot's safety.

D. Planning Problem

The planning problem considered in this work seeks a dynamically feasible path in the belief space $\mathcal{B}$ that is probabilistically safe. Formally, let $B_{\text{goal}} \subseteq \mathcal{B}$ denote the set of all belief states that correspond to the desired goal region in the environment. Then, the constrained planning problem is to compute a dynamically-feasible path $\xi : [0,\ T] \to \mathcal{B}$ for system (2) such that $\xi(0) = b_{\text{start}} \in \mathcal{B}$, $\xi(T) \in B_{\text{goal}}$, $\mathbf{u}(t) \in \mathcal{U}$, and $1 - p_{\text{collision}}(\xi(t),\ \mathcal{M}) > p_{\text{safe}}$ for all $t \in [0,\ T]$.

III. Framework for Multi-Constrained Online Path Planning

To efficiently plan paths that meet motion and probabilistic constraints, while a robot moves through an unexplored environment, this work presents the framework depicted in Fig. 2. Such a framework is threefold: (i) the mapping module that incrementally builds an uncertainty-aware map, (ii) the planning module that finds a safe and feasible path towards the goal, and (iii) the framework manager that coordinates the framework's execution while interacting with the vehicle's perception sensors and trajectory tracking control.

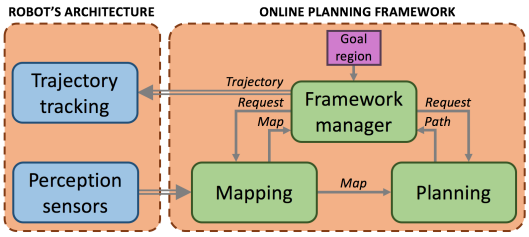


Fig. 2: Framework for incrementally mapping and planning under motion and uncertainty constraints.

A. Mapping Module

Incrementally exploring the surroundings with a location-uncertain system leads to an uncertain map. Under these conditions, obtaining a consistent representation of the entire environment $\mathcal{M}$ is a challenging task, which goes beyond the scope of this work. Instead, this work represents $\mathcal{M}$ as a sequence of independent local stochastic maps [20]:

$$\mathcal{M} = \{\mathcal{LM}_0,\ \ldots,\ \mathcal{LM}_n\}, \tag{8}$$

where $\mathcal{LM}_i$ is a local map, i.e. an area $m_i \in \mathcal{M}$ assumed to have a uniform and null uncertainty (due to zero-uncertainty assumption on observations) with respect to the local map's reference frame $\{\mathcal{LM}_i\}$. To make the uniformity assumption consistent within each $\mathcal{LM}_i$, a new local map is started when the relative uncertainty between the system and the current $\{\mathcal{LM}_n\}$ is over a predefined maximum value, where n corresponds to the total number of local maps describing $\mathcal{M}$. Because the system in (5) holds linear time-invariant (LTI) properties, the local maps management consists in periodically defining a new local map as the triplet:

$$\mathcal{LM}_i = \{\{\mathcal{LM}_i\},\ k_i,\ m_i\}, \tag{9}$$

where k_i is the time at which the $\mathcal{LM}_i$ is created.

While a local map $\mathcal{LM}_n$ is active, i.e. when the uncertainty is within the permitted limits, the environment information is directly represented with an Octomap [21]. Octomaps permit fusing range-based data into a probabilistic voxel representation, which generates a three-dimensional (3D) occupancy grid map with adjustable resolution. Thus, an area m_i is discretely represented as a set of voxels $\{\mathbf{v}_1, \ldots, \mathbf{v}_m\} \in V_i \sim m_i$. The Octomap directly permits to differentiate between free, occupied, and unknown voxels $\mathbf{v}_j \in \mathcal{LM}_n$. The local map $\mathcal{LM}_n$ expands the environment awareness with the occluded (hidden) regions, i.e. unseen voxels that are within the maximum sensor's range but located behind the obstacles (occupied space). This expanded representation establishes three different labels with their corresponding probability of occupancy: $p_{\text{occupied}}(\mathbf{v}_j)$: free voxels $\mathbf{v}_F$ are set to $p_{\text{occupied}}(\mathbf{v}_F) = 0$, occluded voxels $\mathbf{v}_H$ with $p_{\text{occluded}}(\mathbf{v}_H) = 0.5$ and occupied voxels $\mathbf{v}_O$ with $p_{\text{occupied}}(\mathbf{v}_O) = 1$. This interpretation of the environment awareness lets us consider those zones that are likely to be occupied. All the information related to a local map is stored in an octree data structure, which provides fast access time while, at the same time, optimising the use of memory.

B. Planning Module

The planning problem defined in Section II-D has three main requirements: to consider the vehicle's motion capabilities, to validate probabilistic constraints, and to meet online computation limitations. Sampling-based methods, particularly tree-based, provide a good alternative to meet these requirements. Tree-based algorithms are commonly composed by two procedures: *sample* and *extend* [7], [8]. The former one randomly samples states from the state space $\mathcal{X}$, towards which the latter one expands the tree. This twofold procedure can also be conducted in the belief space $\mathcal{B}$, by not only considering the system's motion capabilities, but also by incorporating the uncertainties associated with the system's motion and the perception of the surroundings (Fig. 3).

1) Planning Under Uncertain Motion Constraints: the system's motion capabilities are considered in the planner by expanding the tree with the system's closed-loop model defined in (5). The randomly sampled states are used as input references $\mathbf{r}$ for the closed-loop system, which allows us to guide the tree growth towards such a state. This expansion is done for a period of time T_{prop}. This strategy of using the closed-loop model to expand the tree has been adopted from the work presented by Kuwata et al. [22]. Furthermore, since the proposed formulation presented in (5) includes the system's uncertainty, each obtained belief (node) corresponds to a vehicle's state with its associated uncertainty.

Most of the tree-based algorithms usually do not optimize any metric or function. One alternative to cope with this issue is to use a asymptotic optimal RRT (RRT*) while approximating the dynamic range of the system to Dubins curves. This alternative has already been explored in [17], [18]. Instead, aiming to preserve the entire system's dynamic range, our planning module uses a shooting approach, which consists of expanding the tree from the node with the lowest cost within a neighbourhood of δ-radius. This strategy is also adopted in the stable sparse RRT (SST) planner to obtain asymptotically near-optimal solutions [23].

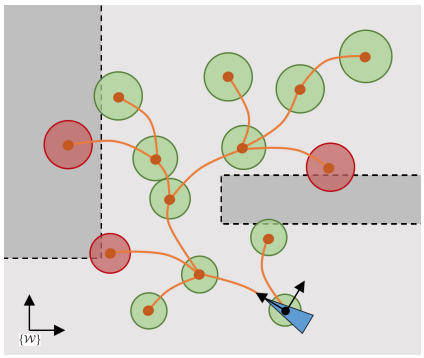


Fig. 3: Tree expansion under motion and probabilistic constraints. The states (nodes) of the tree are obtained by considering the motion capabilities. The spheres surrounding the states represent their uncertainty, where green corresponds to those states satisfying the probabilistic safety constraints, and red those that do not.

2) Planning Under Probabilistic Constraints: the previously explained tree expansion leads to a tree of motions (edges) and states (nodes) which are uncertain with respect to its root, i.e. the state where all the planning tree is rooted to. However, the uncertainty of all these elements can be referenced to the global frame by taking into account the uncertain location of the root into the world. Once all the states and their associated uncertainties are referenced to the global frame, then $p_{\text{collision}}$ can be bounded as presented in (7). Nonetheless, this computation results conservative because it does not consider that the relative uncertainty between the robot and its observation is zero. In fact, the relative uncertainty between the system and the observed obstacle will only start growing once the obstacle goes beyond the sensor detection range. Then, a less conservative formulation of (7) that already considers the discrete nature of the environment representation can be defined as:

$$p_{\text{collision}}(b_k,\ \mathcal{M}) \leq \sum_{\mathbf{x} \in \mathbf{v}_H, \mathbf{v}_O} p_{\text{occupied}}(\mathbf{x} \in \{\mathbf{v}_H | \mathbf{v}_O\})\ \mathcal{K}, \quad (10)$$

where $\mathcal{K}$ is a kernel containing the discretised version of $\mathcal{N}(\mathbf{x} \mid \hat{\mathbf{z}}_k,\ P_{\text{relative}})$, where P_{relative} is the relative uncertainty between the state $\mathbf{x}$ located in a hidden ($\mathbf{v}_H$) or occupied ($\mathbf{v}_O$) voxel in any of the local maps, and the state $\hat{\mathbf{z}}_k$ that wants to be checked for collision. Retrieving the voxel's timestamp k_i and the timestamp k_j of the state $\hat{\mathbf{z}}_k$, the relative uncertainty between these elements is defined as:

$$P_{\text{relative}} = \sum_{m=k_i}^{k_j - 1} A^{k_j - m - 1} P_{\mathbf{w}} (A^T)^{k_j - m - 1}, \quad (11)$$

where $P_{\mathbf{w}}$ is, as introduced in (2), the covariance matrix estimating the AUV's motion uncertainty.

The formulation in (10) is less computationally expensive than (7) since it does not integrate over the entire state space, but instead it uniquely considers those states in collision.

3) Online Planning: the planning module uses different strategies to meet online computation constraints. The first of them is anytime calculation, which allows the planner to provide the best available solution at any moment [24]. Another strategy seeks to decrease the computation time when checking the validity of the motions during the tree expansion. To do so, the planning module opportunistically evaluates (10), avoiding to check the validity of states that belong to the unknown space. Such a strategy was originally presented in [17], [18]. Furthermore, for those states that must be evaluated, the kernel $\mathcal{K}$ is verified by following a spiral pattern, i.e., moving from the kernel's centre outwards. This heuristic seeks to firstly check the voxels where the system has the highest changes. Finally, because the Gaussian estimation of the system's state extends infinitely over the whole environment, the kernel $\mathcal{K}$ is bounded at 3.03σ, thus being 99% certain of the state validation decision.

C. Framework manager

The framework manager is in charge of controlling the previously presented mapping and planning modules. These modules are coordinated following the pipeline presented in Algorithm 1 until reaching a predefined goal region B_{goal} (line 7). In this iterative process, the algorithm firstly requests an update of the environment awareness $\mathcal{M}$ (line 8). Then, if a previous solution old_path is available, it is probabilistically checked for collision according to the new map $\mathcal{M}$. If old_path is no longer valid, the framework manager sends to the controller the portion of old_path that is still safe, while the planner finds a new valid solution path (line 10 and 11). This approach prevents the vehicle stopping every time that a path gets partially invalidated while ensuring its safety.

The planning stage starts by placing the root of the new tree on a predicted position along the current valid path (line 12). This (i) guarantees the feasibility of reaching the root and (ii) prevents sudden changes in the vehicle's direction of motion when a replanning manoeuvre is required. Once the root is defined, the planner grows a new tree in $\mathcal{B}$ for a specific amount of time $planning\_time$ (line 13).

The planner tries to find a near-optimal path within the allocated $planning\_time$, and returns a new_path if one has been found (line 14). However, considering that the tree expands faster in undiscovered areas rather than in the presence of obstacles, the new_path is only executed under two circumstances (line 15 to 17): (i) if the new path has a lower cost than the path that is currently being followed, or (ii) if the new path does not drastically change the trajectory of the invalidated path. This last heuristic promotes the exploration of paths through challenging scenarios.

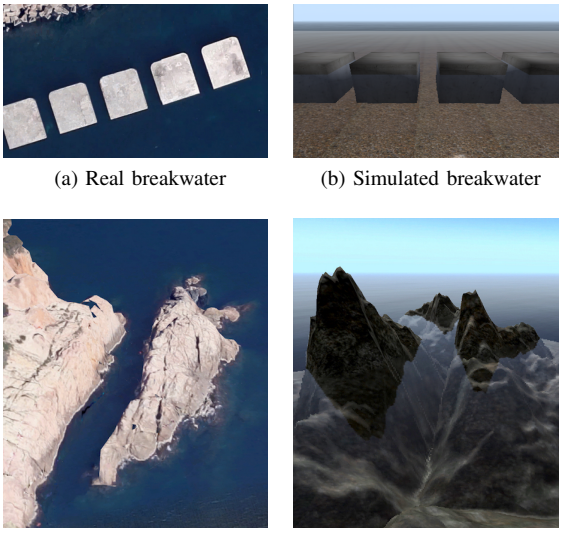
Algorithm 1: MANAGER(B_{goal}, p_{safe})

```
1  Input:
2  B_goal: Goal region.
3  p_safe: Desired level of probabilistic safety guarantees.
4  begin
5      {M, B, old_path} ← ∅
6      planner.loadProblem(B_goal, p_safe)
7      while not isGoalAchieved() do
8          M ← mapper.getMap()
9          planner.updateMap(M)
10         if not planner.isValid(old_path) then
11             sendPath(old_path)
12         planner.setNewRoot(old_path)
13         planner.solve(planning_time)
14         new_path ← planner.getSolution()
15         if bestPath() = new_path then
16             sendPath(new_path)
17             old_path ← new_path
```

IV. EXPERIMENTAL EVALUATION

The proposed framework has been implemented in robot operating system (ROS) and uses the facilities provided by the OMPL [25] to ease the motion planning requirements. To conduct the experiments reported in this section, the framework has been integrated with the component oriented layer-based architecture for autonomy (COLA2) [26], a control software architecture fully developed in ROS that controls the AUVs from the Underwater Vision and Robotics Research Centre (CIRS). One of such vehicles is the Sparus II, a nonholonomic AUV used in this work (Fig. 1). The Sparus II is a torpedo-shaped vehicle with hovering capabilities rated for depths up to $200m$ [27]. For the trials presented in this paper, the AUV was equipped with a mechanical scanned imaging sonar (MSIS) to perceive the surroundings.

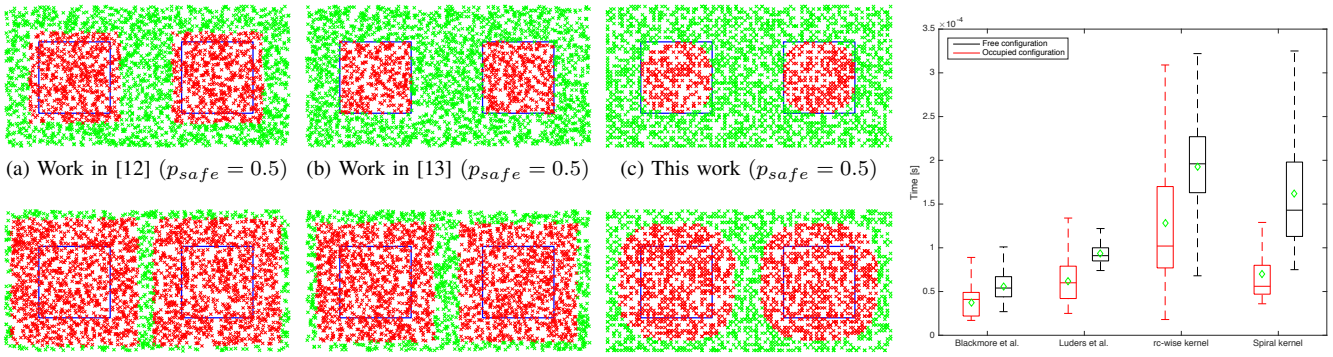
The test-bed to evaluate the proposed approach consists of two environments located in Sant Feliu de Guíxols (Spain): (a) breakwater structure that is composed of a series of concrete blocks ($14.5m$ long by $12m$ width), which are separated by four-metre gaps (Fig. 4a and Fig. 4b), and



(a) Real breakwater (b) Simulated breakwater

(c) Real canyon (d) Simulated canyon

Fig. 4: Evaluation scenarios.



(a) Work in [12] ($p_{safe} = 0.5$) (b) Work in [13] ($p_{safe} = 0.5$) (c) This work ($p_{safe} = 0.5$)

(d) Work in [12] ($p_{safe} = 0.95$) (e) Work in [13] ($p_{safe} = 0.95$) (f) This work ($p_{safe} = 0.95$)

(g) Computation time

Fig. 5: Comparison of the proposed probabilistic collision checking method with the approaches in [12] and [13]. (a) to (f) illustrate the $\mathcal{X}_F$ (green) and the $\mathcal{X}_O$ (red) in the breakwater structure scenario (blue lines) according to different methods and values of p_{safe}. (g) shows the average (green diamonds) and quartiles of the computational cost for each considered method; the proposed one has been evaluated with two kernel-tranversing approaches: row or column-wise and spiral-like.

(b) rocky formations that create an underwater canyon of $28m$ long (Fig. 4c and Fig. 4d). Using these environments, three experiments are reported: (i) comparison of the proposed probabilistic collision checking method with other approaches in the literature, (ii) evaluation of the overall performance of the framework in the underwater simulator (UWSim) [28] and (iii) validation of the framework in real in-water trials. Experiments (i) (ii), and (iii) are tested in the breakwater structure scenario, while experiment (ii) is also evaluated in a simulated underwater canyon.

A. Comparison of Probabilistic Collision Checking Methods

The proposed collision checking algorithm is compared with the most related methods in the state-of-the-art [12], [13]. Their performance is evaluated in the simulated breakwater structure in terms of exploration of $\mathcal{B}$ and computation time. For that, two different tests have been carried out: (i) evaluation of the map exploration according to the acceptance rate of 3,000 states uniformly sampled over $\mathcal{B}$ when fixing $p_{\text{safe}} = 0.5$ (Fig. 5a to Fig. 5c) and $p_{\text{safe}} = 0.95$ (Fig. 5d to Fig. 5f), and (ii) estimation of the computation time by executing each method 10 times to then, from each trial, take the computation time of 12,000 $\mathbf{x} \in \mathcal{X}_F$ and 12,000 $\mathbf{x} \in \mathcal{X}_O$. Fig. 5g depicts the average of the resulting 120,000 samples per state-category and per method. Our proposal is evaluated using two different kernel-traversing approaches: row-column-wise and spiral-like.

As depicted in Fig. 5g, our spiral-like traversing strategy is faster than the traditional row-column-wise approach. Thus, only the former approach is compared with [12], [13] in Table I. As a summary, the most exhaustively $\mathcal{B}$ is checked, the more time is required. This suggests that the trade-off between exploration and computation time depends on the type of environment; being conservative in narrow passages makes the free state space $\mathcal{X}_F$ smaller, limiting the chances to find a path to the goal. For instance, when considering $p_{\text{safe}} = 0.95$, only 608 $\mathbf{x} \in \mathcal{X}_F$ (see Fig. 5d) when using [12], in contrast to the 1,204 $\mathbf{x} \in \mathcal{X}_F$ (see Fig. 5f) when using our method. This implies that spending more time to exhaustively explore the $\mathcal{B}$ is beneficial when dealing with undiscovered environments. On top of that, and in contrast to the approaches in [12] and [13], the proposed kernel-based method deals with nonconvex representations of the environment, thus avoiding convexification routines, which usually involve a loss of the environment awareness and an increase of the computation time.

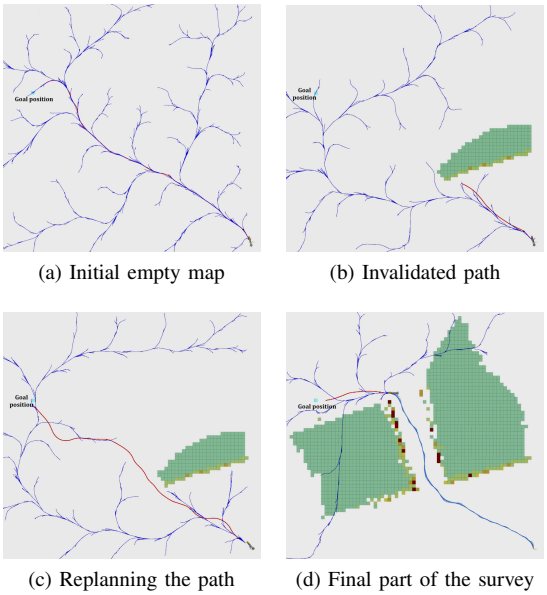
TABLE I: Trade-off between acceptance rate and computation time needed to accept and reject a state.

	Accepted samples		**Average time [ms]**	
	0.50	**0.95**	**Accept**	**Reject**
Blackmore et al. [12]	1,752	608	0.052	0.034
Luders et al. [13]	2,230	922	0.093	0.063
This work (spiral)	2,342	1,204	0.169	0.076

B. Start-to-goal Queries in Undiscovered Environments

The overall proposed framework has been tested in the previously introduced environments (Fig. 4) by attempting to solve different start-to-goal queries. In both simulated and in-water experiments, the robot was required to map the environment with a $0.5m$ map resolution, and to navigate at a constant depth of $1.5m$ with a maximum velocity of $0.35m/s$. The minimum probability of safeness for all experiments was set at $p_{\text{safe}} = 0.9$.

1) Simulated trials: before conducting in-water experiments, the framework was exhaustively tested in the simulated breakwater structure and canyon scenarios. In the former environment, 19 start-to-goal queries out of 20 attempts were successfully solved. Among those 19 successful experiments, the robot achieved the goal region B_{goal} by crossing through the first four-metre gap in 17 occasions, while in the remaining two trials, the planner found a less optimal path through the second four-metre passage. Fig. 6 depicts the mission execution in one of those trials. In the initial part of the mission, the environment is completely undiscovered, which makes the solution path to be a straight



(a) Initial empty map (b) Invalidated path

(c) Replanning the path (d) Final part of the survey

Fig. 6: Incrementally mapping and planning in the undiscovered breakwater structure scenario.

trajectory to the goal (Fig. 6a). As soon as the path gets invalidated (Fig. 6b), a new collision-free path is considered (Fig. 6c). After some mapping-planning iterations, the robot gets out of the four-metre gap between two blocks (Fig. 6d). In average, the calculated path towards the goal has a length of approximately $45.2m$ and is completed within $2'21''$.

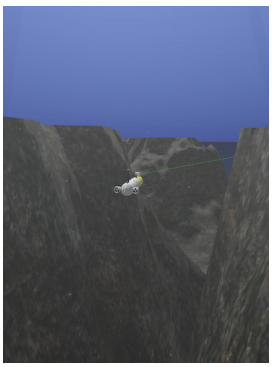
The start-to-goal query in the simulated canyon scenario has been successfully solved 20 times out of the performed 20 trials. The higher success rate with respect to the previous experiment is given by the nature of the environment; this scenario involves less abrupt manoeuvres and the passage is wider, more than twice larger though. Fig. 7 depicts the path calculated in one of those successful trials through the narrow passage in the middle of the canyon. In average, the calculated paths towards the goal have a length of approximately $58.4m$ and are completed within $2'59''$.

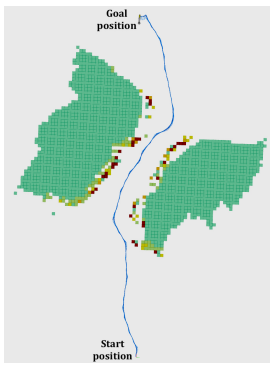
After those 40 trials conducted in two different scenarios, the framework has demonstrated to have a high success rate when solving start-to-goal queries in undiscovered environments. This performance is because the planner exploits the entire system's dynamic range, thus allowing the vehicle to execute more complex manoeuvres by adjusting the surge (forward) velocity and prioritising the turning rate.

2) Real-world trials: after the framework had demonstrated a satisfactory performance and high success rate in the simulated trials, it was deployed on the Sparus II AUV to prove its suitability for real-world robots with limited onboard computation power. The in-water experiments were conducted in the real breakwater structure located in Sant Feliu de Guíxols (Spain). The robot was required to solve a start-to-goal-query to reach a goal region B_{goal} located on the opposite side of the structure, which can only be achieved by navigating through any of the narrow four-metre gaps. During those autonomous missions, the vehicle is connected to a wireless access point buoy for monitoring purposes.

Prior experiments before running the entire framework revealed that the robot's behaviour when dealing with real conditions diverged from simulations in (i) the perception quality, being significantly noisier and (ii) the trajectory tracking performance, being slightly degraded because of the waves and currents. Noise on the observations was reduced by experimentally adjusting the range of the mechanical scanned imaging sonar (MSIS) at 10 metres.

Despite these challenging conditions, the framework successfully accomplished finding and driving the Sparus II AUV towards the desired goal region B_{goal} through one of the narrow gaps in the breakwater structure[1]. The path was found through the first corridor four out of these five times, while in the other trial the robot went through the second gap. Fig. 8 depicts Sparus II in one of those in-water trials and the path calculated towards the goal, which has a length of $57.9m$ and took $3'07''$.

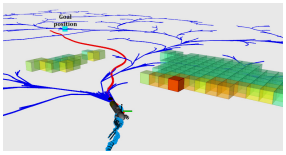




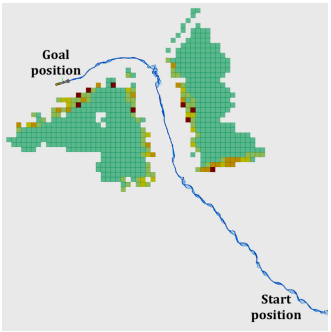
(a) Sparus II in the UWSim (b) Path calculated in the canyon

Fig. 7: Incrementally mapping and planning in the undiscovered canyon scenario.

(a) Sparus II during the survey



(b) Path towards the goal



(c) Path through the breakwater

Fig. 8: Sparus II AUV guided by the proposed uncertainty-based framework to solve a start-to-goal query in an undiscovered environment.

[1]A complete sea-trial through the real breakwater structure can be seen in: `https://youtu.be/dTejsNqNC00`.

V. FINAL REMARKS

This paper has presented a novel end-to-end framework which probabilistically guarantees the robot's safety when solving start-to-goal queries in unexplored environments. The proposed approach is twofold: (i) incrementally maps the vehicle's surroundings to build an uncertain representation of the environment, and (ii) plans feasible paths (according to the robot's kinodynamic constraints) with probabilistic safety guarantees (according to the uncertainties in the vehicle's motion and mapping). An efficient evaluation of the uncertainties allows the planner to exhaustively check for collisions, even in non-convexified environments. Moreover, anytime capabilities and opportunistic collision checking routines make the framework suitable for online applications with limited on-board computation power. Experimentation conducted in simulation proofs some of the theoretical qualities of this work, namely the lower rate of in-collision samples than the other uncertainty-aware planners in the literature. Additionally, simulated and real-world trials on the Sparus II AUV demonstrated the suitability of the framework to guarantee the robot's safety while navigating in unexplored environments and dealing with real-robots constraints.

The framework is not restricted to the presented experimental evaluation nor platform. Any other AUV, terrestrial or aerial systems with a linearisable equation of motion can benefit from this work. The presented work can also be extended to 3D scenarios as long as online computation constraints can still be met. Apart from that, the in-water trials pointed out the need of automatically adjusting the replanning period, exploring sampling strategies that promote obtaining a denser exploration in tight spaces, and considering the mapping uncertainty and its constraints into the framework. Next experimental efforts should focus on conducting experiments in the real canyon scenario.

REFERENCES

[1] R. B. Wynn, V. A. Huvenne, T. P. Le Bas, B. J. Murton, D. P. Connelly, B. J. Bett, H. A. Ruhl, K. J. Morris, J. Peakall, D. R. Parsons, *et al.*, "Autonomous Underwater Vehicles (AUVs): Their past, present and future contributions to the advancement of marine geoscience," *Marine Geology*, vol. 352, pp. 451–468, 2014.

[2] F. S. Hover, R. M. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. J. Leonard, "Advanced perception, navigation and planning for autonomous in-water ship hull inspection," *The International Journal of Robotics Research*, vol. 31, no. 12, pp. 1445–1464, 2012.

[3] B. Bingham, B. Foley, H. Singh, R. Camilli, K. Delaporta, R. Eustice, A. Mallios, D. Mindell, C. Roman, and D. Sakellariou, "Robotic tools for deep water archaeology: Surveying an ancient shipwreck with an autonomous underwater vehicle," *Journal of Field Robotics*, vol. 27, no. 6, pp. 702–717, 2010.

[4] S. Hert, S. Tiwari, and V. Lumelsky, "A terrain-covering algorithm for an auv," *Autonomous Robots*, vol. 3, no. 2, pp. 91–119, 1996.

[5] A. Mallios, P. Ridao, D. Ribas, M. Carreras, and R. Camilli, "Toward autonomous exploration in confined underwater environments," *Journal of Field Robotics*, vol. 33, no. 7, pp. 994–1012, 2016.

[6] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[7] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[8] D. Hsu, J.-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," in *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, vol. 3, pp. 2719–2726, IEEE, 1997.

[9] R. Alterovitz, T. Siméon, and K. Y. Goldberg, "The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty.," in *Robotics: Science and systems*, pp. 233–241, 2007.

[10] V. A. Huynh, S. Karaman, and E. Frazzoli, "An incremental sampling-based algorithm for stochastic optimal control," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 2865–2872, IEEE, 2012.

[11] R. Luna, M. Lahijanian, M. Moll, and L. E. Kavraki, "Optimal and efficient stochastic motion planning in partially-known environments," in *The Twenty-Eighth AAAI Conference on Artificial Intelligence*, (Quebec City, Canada), pp. 2549–2555, 27/07/2014 2014.

[12] L. Blackmore, H. Li, and B. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *American Control Conference, 2006*, pp. 7–pp, IEEE, 2006.

[13] B. Luders, M. Kothari, and J. How, "Chance constrained RRT for probabilistic robustness to environmental uncertainty," in *AIAA guidance, navigation, and control conference*, p. 8160, 2010.

[14] B. D. Luders, S. Karaman, and J. P. How, "Robust sampling-based motion planning with asymptotic optimality guarantees," in *AIAA Guidance, Navigation, and Control (GNC) Conference*, p. 5097, 2013.

[15] Y. Petillot, I. T. Ruiz, and D. M. Lane, "Underwater vehicle obstacle avoidance and path planning using a multi-beam forward looking sonar," *IEEE Journal of Oceanic Engineering*, vol. 26, no. 2, pp. 240–251, 2001.

[16] T. Maki, H. Mizushima, H. Kondo, T. Ura, T. Sakamaki, and M. Yanagisawa, "Real time path-planning of an AUV based on characteristics of passive acoustic landmarks for visual mapping of shallow vent fields," in *OCEANS 2007*, pp. 1–8, IEEE, 2007.

[17] J. D. Hernández, M. Moll, E. Vidal, M. Carreras, and L. E. Kavraki, "Planning feasible and safe paths online for autonomous underwater vehicles in unknown environments," in *Intelligent Robots and Systems (IROS)*, pp. 1313–1320, IEEE, 2016.

[18] J. D. Hernández, E. Vidal, M. Moll, N. Palomeras, M. Carreras, and L. E. Kavraki, "Online motion planning for unexplored environments using auvs," *Journal of Field Robotics*, p. to appear, 2018.

[19] A. De Luca, G. Oriolo, and M. Vendittelli, "Stabilization of the unicycle via dynamic feedback linearization," in *6th IFAC Symp. on Robot Control*, pp. 397–402, 2000.

[20] P. Piniés and J. D. Tardós, "Scalable SLAM building conditionally independent local maps," in *Intelligent Robots and Systems, 2007. IROS 2007.*, pp. 3466–3471, IEEE, 2007.

[21] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, 2013. Software available at `http://octomap.github.com`.

[22] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, 2009.

[23] Y. Li, Z. Littlefield, and K. E. Bekris, "Asymptotically optimal sampling-based kinodynamic planning," *The International Journal of Robotics Research*, vol. 35, no. 5, pp. 528–564, 2016.

[24] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1478–1483, IEEE, 2011.

[25] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, pp. 72–82, December 2012. `http://ompl.kavrakilab.org`.

[26] N. Palomeras Rovira, A. El-Fakdi Sencianes, M. Carreras Pérez, and P. Ridao Rodríguez, "COLA2: A control architecture for AUVs," © *IEEE Journal of Oceanic Engineering, 2012, vol. 37, p. 695-716*, 2012.

[27] M. Carreras, C. Candela, D. Ribas, N. Palomeras, L. Magí, A. Mallios, E. Vidal, È. Vidal, and P. Ridao, "Testing Sparus II AUV, an open platform for industrial, scientific and academic applications," *Instrumentation viewpoint*, no. 18, pp. 54–55, 2015.

[28] M. Prats, J. Pérez, J. J. Fernández, and P. J. Sanz, "An open source tool for simulation and supervision of underwater intervention missions," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 2577–2582, IEEE, 2012.