# Online Mapping and Motion Planning Under Uncertainty for Safe Navigation in Unknown Environments

Èric Pairet , *Associate Member, IEEE*, Juan David Hernández , *Senior Member, IEEE*, Marc Carreras , *Member, IEEE*, Yvan Petillot , *Member, IEEE*, and Morteza Lahijanian , *Member, IEEE*

*Abstract*—Safe autonomous navigation is an essential and challenging problem for robots operating in highly unstructured or completely unknown environments. Under these conditions, not only robotic systems must deal with limited localization information but also their maneuverability is constrained by their dynamics and often suffers from uncertainty. In order to cope with these constraints, this article 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: 1) incrementally mapping the surroundings to build an uncertainty-aware representation of the environment and 2) iteratively (re)planning trajectories to goal that is kinodynamically feasible and probabilistically safe through a multilayered sampling-based planner in the belief space. In-depth empirical analyses illustrate some important properties of this approach, namely: 1) the multilayered planning strategy enables rapid exploration of the high-dimensional belief space while preserving asymptotic optimality and completeness guarantees and 2) the proposed routine for probabilistic collision check-ing results in tighter probability bounds in comparison to other uncertainty-aware planners in the literature. Furthermore, real-world in-water experimental evaluation on a nonholonomic torpedo-shaped autonomous underwater vehicle and simulated trials in an urban environment on an unmanned aerial vehicle demonstrate the efficacy of the method and its suitability for systems with limited onboard computational power.

*Note to Practitioners*—Emergent robotic applications require operating in previously unmapped scenarios. This article presents a unified mapping–planning strategy that enables robots to navigate autonomously and safely in harsh environments.

*Index Terms*—Field robotics, online mapping, online motion planning under uncertainty, safe autonomous navigation in unknown environments, sampling-based motion planning.

## I. INTRODUCTION

AUTONOMOUS robots have been increasingly employed to assist humans notably in hazardous or inaccessible environments in recent years. Examples include rescue missions in disaster response scenarios [7], in-water ship hull [31] and wind turbine inspections [51], and deep underwater and space exploration [4], [74], among many others. A fundamental requirement for a robot engaged in any of these applications is to be adept at navigating autonomously through highly unstructured and hostile environments. However, this is not a trivial task due to a limited or complete lack of prior knowledge about the environment in which the robot has to operate. This implies that the robot has to base its decision-making on onboard sensors despite their limited accuracy. In addition, the robot itself might suffer from poor localization, as well as restricted and uncertain maneuverability. Therefore, even though challenging, it is essential to jointly consider all these motion and sensory constraints, as well as their associated uncertainties, when planning for navigation actions. This problem becomes particularly more challenging in safety–critical missions where the robot's safety must be ensured at all times.

Although there exist alternative methodologies addressing each of the abovementioned issues individually, limited attention has been devoted to the autonomous navigation problem in unknown environments as a whole [44]. The classical algorithms known as simultaneous localization and mapping (SLAM) enable a mobile robot to concurrently build and use a map to estimate its location [17]. These algorithms rely on identifying distinctive landmarks, which can bound the

uncertainty of both the environment representation and the robot localization. Nonetheless, even for scenarios rich in features, there are always some residual uncertainties. More recently, online motion planning frameworks have been developed to empower a mobile robot to compute navigation actions in unexplored environments while accounting for the system's motion capabilities, e.g., [11], [21], [27]–[29], [68], [76], [80]. These approaches, however, do not cope with any source of uncertainty and employ *ad hoc* heuristics that lack quantified safety guarantees. The few attempts to ensure safety through probabilistic methods, such as [12], [35], [71], are generally computationally expensive, built on strong assumptions, and commonly suppose a complete prior environment knowledge. Therefore, they are unsuitable for applications requiring online computations to deal with unknown environments.

In this context, our previous framework guaranteed (in compliance with a user-defined minimum probability of safety) the robot's safety when navigating through unexplored environments [55]. The underlying strategy consisted of an iterative mapping–planning scheme capable of continuously modifying the vehicle's motion plan toward the desired goal according to the incremental environmental awareness. At any time, the resulting motion plan was guaranteed to be feasible and safe in face of localization, mapping, and motion uncertainties. Despite the promising results achieved with this iterative mapping–planning scheme, its underlying formalization had some limitations. Namely, the framework was exclusively tailored to cope with a low-dimensional robot (three *degrees of freedom* (DoFs)) navigating in an unknown, symmetrically structured, 2-D workspace. The initially proposed mapping–planning scheme and its constituent components would scale poorly when dealing with systems and scenarios of higher complexity. More demanding problems exacerbate the curses of dimensionality and computational load to guarantee probabilistic safeness in face of uncertainties. Such a challenge motivated the development of this follow-up work to extend the framework's capabilities to suit the requirements of a larger group of robotic systems and environments.

Building on our previous mapping–planning scheme [55], the main contribution of this article is threefold.

1) Multilayered planning strategy capable of rapid search in high-dimensional belief spaces, with asymptotical optimality and probabilistic completeness guarantees.
2) Probabilistic map fusion that efficiently retrieves environmental uncertainties in form of a cumulative map, while dealing with overlapping local submaps.
3) Probabilistic collision checking routine, which rapidly evaluates the validity of a state subject to uncertainties by trading the tightness of the safety bound for computational efficiency, while accounting for the tail events.

Our new contributions in the framework's key constituent components are supported with rigorous theoretical development and thorough experimental evaluations. These novel advancements allow for faster online motion planning and more efficient evaluation of uncertainties. Consequently, the improved framework is now capable to compute navigation actions online for high-dimensional systems and more challenging unknown environments while providing safety guarantees. To the best of our knowledge, this is the first generic architecture capable of jointly dealing with kinodynamic and probabilistic constraints in unknown environments online. Both the precedent and new framework are analyzed and compared in multiple scenarios with different interesting real-world[1] and simulated[2] physical systems. The experimental results demonstrate the suitability of the proposed method to address the challenge of probabilistically safe autonomous navigation in unknown environments while being suitable for systems with limited onboard computational power.

The remainder of this article is organized as follows. Section II provides a comprehensive review of the literature and the corresponding contribution of this article. Then, Section III formally defines the considered problem. In Section IV, an overview of the framework is presented, and then, the mapping and planning components are detailed in Sections V and VI, respectively. The description of the framework is followed by a thorough analysis of its key constituent features and its performance and capabilities as a whole in Section VII. Finally, this article concludes with a discussion in Section VIII.

## II. RELATED WORK

This section gives a brief overview of prior work on planning under kinodynamic constraints and planning under uncertainty, as well as frameworks for online mapping–planning. Finally, this section discusses all contributions of this work with respect to the latest related literature.

### A. Planning Under Kinodynamic Constraints

Planning under kinodynamic constraints deals with the challenge of computing trajectories that are feasible according to the vehicle's motion capabilities. This problem is commonly formulated as finding a trajectory between two points through the system's state space. The robotics literature offers various approaches to tackle this problem.

One strategy is to represent the continuous state space as a lattice space, i.e., a graph where edges correspond to a reduced set of precomputed motion primitives. Then, the motion planning problem can be efficiently solved using graph search algorithms. For the particular case of a car-like system, the motion primitives can be defined as a set of lines and arcs to build a geometric state lattice [16], [67]. These approaches can find the shortest path, but the transition between segments presents abrupt changes in angular velocity, which could only be achieved by a system capable of infinite angular acceleration. More complex lattice space definitions allow the consideration of more restrictive concatenation rules and richer sets of primitive motions, e.g., [20], [62], at the cost of more memory usage and more computationally expensive queries. Even though planning in lattice spaces has proven to be suitable for many applications, it requires the crafting of a set of motions such that the resulting lattice offers,

---

[1] A mission through a real breakwater structure with an autonomous underwater vehicle (AUV) can be seen in: https://youtu.be/dTejsNqNC00

[2] A mission in the DARPA Subterranean Challenge 2019 scenario with an unmanned aerial vehicle (UAV) can be seen in https://youtu.be/I5X_QFKDpeI

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

PAIRET *et al.*: ONLINE MAPPING AND MOTION PLANNING UNDER UNCERTAINTY FOR SAFE NAVIGATION 3

at least, one suitable solution to the planning problem. Some works in the learning community have addressed this difficult and time-consuming task with data-driven techniques [14]. However, the resulting set of motions still represents a very limited range of the real dynamic capabilities of the robot. This is undesirable in applications where the environment is not known in advance, and where having the entire dynamic range of motions available for planning can be critical to finding a suitable solution. All in all, lattice-based methods struggle with planning in high-dimensional state spaces.

To deal with kinodynamic constraints, sampling-based motion planners offer great opportunities, e.g., [32], [36], [38]. Most sampling-based planners, however, lose their asymptotic optimality guarantees when a steering function does not exist in the system's kinodynamically constrained state space. To cope with this limitation, there are different assumptions and heuristics that can be applied at the expense of longer computational times. For example, Webb and van den Berg [78] proposed a version of the asymptotic optimal RRT (RRT*) that can deal with kinodynamic constraints of systems with linearisable dynamics [78]. If the system's dynamics are not linearisable, asymptotic optimality can be obtained in any planner by augmenting the dimensionality of the state space to account for the search cost [25]. However, this strategy implies solving the planning problem repeatedly to improve the cost of the solution at each iteration, consequently being unsuitable for applications with online requirements. Finally, the stable sparse RRT (SST) planner offers asymptotically near-optimal guarantees by means of a shooting approach, which consists of expanding the tree from the node with the lowest cost within a neighborhood of predefined $\delta$-radius [41].

Planning in high-dimensional spaces with multiple constraints poses a challenge for classical planners and, typically, results in long computation times if a solution can be found at all. In such problems, a common approach to boost performance is via a multilayered planning scheme. The key idea is to leverage from a lead to guide (warm-start) the search. In this regard, an interesting approach is the incremental trajectory optimization for motion planning (ITOMP) algorithm, which interleaves planning and optimization; the planner is given a fixed time budget to find a solution, which is then used as a warm-start for the optimizer [58]. Work in [63] and [64] introduced a synergistic three-layered planner: the high-level planner uses discrete search to initially determine those candidate regions (from a decomposed representation of the environment), which might contain part of the final solution; a low-level planner employs a sampling-based motion planner to find a solution; and a middle layer updates the candidate regions according to the considered constraints. However, the proposed combination of planners does not guarantee asymptotic optimality, and the discrete planner becomes slow for high-dimensional problems. Palmieri *et al.* [57] presented the Theta*-rapidly exploring random tree (RRT) scheme, which first uses the Theta* path planner to compute a lead path, which is then employed to bias the search of the RRT planner [57]. This approach, however, lacks asymptotic optimality guarantees, given that the second planner is an RRT.

More recently, a multilayered approach based on the RRT* as a lead planner and the SST as the final planner has been proposed in [76]. The final planner's search space is strictly constrained around the lead path, raising concerns about the completeness guarantees of the overall architecture.

### B. Planning Under Uncertainty

An essential capability for any autonomous robot is to operate in the presence of uncertainty [13]. Sources of uncertainty relevant to autonomous systems fall into four types [39].

1) *Uncertainty in Localization:* The robot's location is uncertain with respect to the environment. This issue is particularly critical in robots operating in GPS-denied environments or for systems suffering from low-accuracy state estimation.
2) *Uncertainty in Motion (Dynamics):* The future robot state cannot be predicted accurately, either because of discrepancies between the considered and the real system's dynamic behavior or due to limited precision in the system's command tracking.
3) *Uncertainty in the Environmental Awareness:* The robot has inexact or incomplete information about its surroundings (e.g., obstacle location). This issue can arise from inaccuracies in the *a priori* map, or imperfect and noisy exteroceptive sensory capabilities.
4) *Disturbances in the Operational Environment:* The robot is subject to external factors, such as wind, atmospheric turbulences, or water currents, which makes the robot deviate from the planned trajectory, thus compromising the reliability of deterministic path planning techniques.

This section scrutinizes relevant planning strategies dealing with any of the three first sources of uncertainty. Given the scope of our work, terrain traversability analysis methods (e.g., [19], [24], [54]) are excluded from this review.

One approach that is popular among existing planners is based on discrete Markov processes. This strategy models the evolution of the system in the environment and generates a policy over the approximated Markov states. Examples of such motion planners include stochastic motion roadmap (SMR) [5] and incremental Markov decision process (iMDP) [33]. These methods have shown to be effective and provide optimality guarantees in terms of the probability of reaching the desired goal; however, they assume perfect knowledge about the environment. Works such as [47] have extended these techniques to partially unknown environments. Nonetheless, their large computational times remain the main hurdle in applications with fully unknown environments or requiring online planning.

Another approach to deal with uncertainties in planning is by means of feedback controllers and sampling-based planners. Van Den Berg *et al.* [75] proposed the linear quadratic Gaussian (LQG) motion planning method, which finds the best path simulating the performance of LQG on all extensions of an RRT [75]. This idea was later applied in roadmaps to propose the feedback-based information roadmap (FIRM) [1]. This method, though, relies on full *a priori* awareness of the environment to explore the belief space offline and then to quickly perform queries online. Consequently, this strategy

is not suitable for planning applications where the *a priori* information about the environment, if available, is not fully informative. A similar strategy is used in [2] for simultaneous localization and planning. Alternatively, Sun *et al.* [73] presented the high-frequency replanning (HFR) architecture, a strategy that leverages from an LQG and a multithread RRT, allowing to continuously replan in the face of alterations in the robot or environment space, while accounting for uncertainties. However, the asymptotic optimality guarantees of such a method can only be assured in multithreaded implementations.

An alternative approach to dealing with uncertainty is the chance-constraint strategy. In these methods, instead of maximizing the probability of success, the objective is to find a path that satisfies a minimum safety probability constraint. The challenge in incorporating this method in planners lies in the computation of the safety probability over plans. In [9], 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) [45] and the CC-RRT* [46]. These approaches evolve the system's dynamics in an open-loop fashion, hence growing the uncertainty unboundedly forward in time. To improve accuracy, linear chance constraints were applied after propagation of the system's state conditioned on the precedent states being collision-free [59]. Such a strategy is commonly referred to as truncating the distribution estimating the system's state, and its usage in planning led to the CC-RRT*-D planner [43]. The advantage of chance-constraint-based methods is that satisfying plans can be computed quickly, making them desirable for online applications. They are, however, built on strong assumptions that result in overly conservative calculations and rely on the prior knowledge of a convex environment. Nonetheless, chance-constraint methods are still widely used in the planning community to deal with localization, motion, and environmental uncertainties, e.g., [12], [71].

In recent years, planners based on various discretization methods have been developed to deal with limited computational power or online planning requirements in face of uncertainty. Majumdar and Tedrake [48] proposed a precomputed library of funnels to efficiently estimate the system's kinodynamic and uncertainty propagation in 3-D environments [48]. However, library-based approaches consider a reduced set of the real system's capabilities that can endanger the efficacy of the planner. Another approach in favor of performance consists of approximating the computation of the probability of collision to a discrete support [55], [71]. This strategy truncates the infinite expansion of the belief in a bounded patch considered to contain a large portion of the belief's probability mass. In our previous work [55], all uncertainties were projected onto discrete support, referred to as kernel, whose resolution resembled the optimal one for online mapping applications. Although considering discrete support for the computation of the probability of collision allows for quick calculations, none of the works using such technique actually normalizes the calculations for the probability mass laying outside the patch, i.e., tail events. Therefore, they cannot offer guarantees on the compliance of the probabilistic safety constraints.

## C. Frameworks for Online Mapping–Planning

Limited attention has been devoted to the navigation problem as a whole, especially in the face of uncertainties. Note that the navigation requirements differ from those of coverage path planning, for which there is a perpendicular literature thread, e.g., [23], [37]. Current navigation frameworks in the robotics literature are built on strong assumptions, which could endanger (or completely neglect) some of the essential requirements for safe navigation in undiscovered environments. Some of the prerequisites are the ability to create an uncertainty-aware representation of the environment such that uncertainties about the environment can be considered at the planning stage. It is also crucial to ensure completeness guarantees, i.e., the ability of finding a solution if one exists, and among many others, being capable of guaranteeing the vehicle's safety at any time during the mission. Ideally, an online mapping–planning framework should be able to find paths quickly while offering asymptotic optimality guarantees.

A common strategy for online navigation is to continuously replan in the face of changes in the robot's pose or the environment awareness. Scherer *et al.* [69] endowed an UAV with the capability to map online with an occupancy probabilistic grid and then to guide itself toward the goal with a combination of global and local potential field-based planners [69]. Along this line, navigation in 3-D environments by mapping from stereo vision and planning with the RRT was considered in [6]. The resulting paths of these approaches do not account for kinodynamic constraints or safety guarantees. Alternatively, in [42], the local planner of the RRT approximated an UAV capabilities by 3-D Dubins paths. Nevertheless, none of these approaches considers any of the multiple sources of uncertainty in the mapping nor the planning stage, thus not providing any theoretical performance or safety guarantees.

More recently, Ho *et al.* [29] proposed an online framework to build an uncertainty-aware map and plan over it using the RRT. However, the resulting paths do not meet kinodynamic nor safety constraints. Instead, proposals in [27] and [28] presented an online framework to plan paths under motion constraints for AUVs, but their approach assumes zero uncertainty. While their framework succeeded in solving start-to-goal queries in unexplored real-world environments, their planner used *ad hoc* heuristics to estimate the solution's associated risk and approximated the system's dynamics with Dubins curves. Frameworks can employ multilayered schemes to scope the complexity of online constrained planning in a subregion of the entire planning space, e.g., [18], [76]. Youakim *et al.* [80] presented a multirepresentation, multiheuristic A* planner capable of jointly dealing with mobile-base and manipulation planning in unknown environments while accounting for localization uncertainty via heuristics. Despite all methods have been tested in real-world environments, the underlying frameworks lack of theoretical analysis and do not provide a measure of robustness or quantified safety guarantees.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

PAIRET *et al.*: ONLINE MAPPING AND MOTION PLANNING UNDER UNCERTAINTY FOR SAFE NAVIGATION 5

## D. Closely Related Contributions

An early version of the work presented in this manuscript has appeared before [55]. This consisted of a simpler framework that proved to be suitable for real-world motion planning problems, but its applicability was strictly limited to underwater robots operating at constant depth, i.e., 2-D workspaces. This motivated the development of this follow-up work to extend the framework's capabilities to suit the requirements of a larger group of robotic systems and environments. Overall, given the precedent efforts by the authors, this article provides the following contributions.

1) An online mapping–planning framework that probabilistically guarantees the robot safety during navigation tasks in unknown environments (see Section IV).
2) A mapping strategy using local submaps that builds an uncertainty-aware map (Section V). These calculations now, in contrast to [55], include efficient retrieval of environmental uncertainties and consider probabilistic map fusion to deal with overlapping local submaps.
3) A multilayered planner (MLP) that guides the search in the high-dimensional belief space (Section VI), in contrast to the uniform search of the single-layered planner in [55]. Our planner satisfies kinodynamic constraints and probabilistic safety guarantees while providing probabilistic completeness and asymptotic optimality guarantees.
4) A rapid probabilistic collision checking routine (Section VI-C). In contrast to [55], the calculations now include a controllable confidence level $\alpha$ that allows to trade the tightness of the safety bound for computational efficiency while correcting for the tail events (i.e., the probability mass excluded by the confidence level).
5) A thorough evaluation of the whole framework and its key constituent components (see Section VII). Besides robot deployments on challenging real-world environments, this assessment, in contrast to [55], now includes rigorous analysis on different scenarios and dynamical systems.

Our contributed advancements allow for faster online motion planning and more efficient evaluation of uncertainties. Consequently, the framework now can compute probabilistically safe navigation actions online for high-dimensional systems and more challenging unknown environments.

## III. PROBLEM FORMULATION

In this work, the focus is on the challenging problem of safe autonomous navigation in unexplored environments. To start with, the robotic system must be capable of perceiving and creating a consistent representation of the surroundings despite its potentially uncertain localization. The perceived surroundings must be encoded efficiently such that the robot can exploit them online for planning purposes. Besides the mapping requirements, the process of planning navigation actions toward the desired goal is challenging by itself. The robot must not only account for its limited and uncertain maneuverability but also for the evolving awareness and uncertainty of the

TABLE I
SUMMARY OF THE NOMENCLATURE IN THIS ARTICLE

| Symbol | Description |
|---|---|
| **Generic definitions** | |
| $\mathcal{W}$ | workspace |
| $\mathcal{X}$ | state space |
| $\mathcal{B}$ | belief space |
| $\mathcal{U}$ | control space |
| $\mathrm{b}_k = \mathcal{N}(\hat{\mathbf{x}}_k, \ \Sigma_{\mathbf{x}_k})$ | state belief at time k |
| $\mathrm{b}_A^B = \mathcal{N}(\hat{\mathbf{x}}_A^B, \ \Sigma_A^B)$ | state belief of A as seen from B |
| **Framework** | |
| $W$ | global frame |
| $R$ | robot base frame |
| $R'$ | planning frame |
| **Mapping** | |
| $\mathcal{M}$ | probabilistic map awareness |
| $\mathcal{LM}$ | local submap |
| $F_O(\mathbf{x})$ | occupancy probability at $\mathbf{x}$ |
| $F_{\mathcal{X}}$ | cumulative map over $\mathcal{X}$ |
| **Planning** | |
| $b_{\mathrm{start}}$ | estimated system state at start |
| $\mathcal{X}_{\mathrm{goal}}$ | goal region in state space |
| $\mathcal{B}_{\mathrm{goal}}$ | goal region in belief space |
| $p_{\mathrm{goal}}$ | minimum probability of goal |
| $p_{\mathrm{safe}}$ | minimum probability of safety |
| $\Delta T_{MP}$ | overall planning budget time |
| $\Delta T_L$ | lead planner budget time |
| $\Delta T_C$ | constrained planner budget time |
| $\xi'$ | lead geometric path |
| $\mathcal{X}'$ | lead region in state space |
| $\xi$ | feasible and safe trajectory |

surroundings as the robot moves. This section provides formal definitions for these uncertainties and the problem of safe autonomous navigation in unexplored environments. Table I summarizes the nomenclature used through this article.

## A. Motion Uncertainty and Constraints

Consider a mobile robot that operates in a workspace $\mathcal{W} \subset \mathbb{R}^{n_w}$, where $n_w \in \{2, 3\}$, under motion uncertainty. The uncertainty in the robot's motion can be due to many reasons, e.g., unmodelled dynamics or noise in actuation, and can be described in several ways. In this work, inspired by [8], [26], [40], and [52], the evolution of the uncertain robotic system is assumed to follow a Gaussian process. That is, the robot state $\mathbf{x}_k$ at every time step $k$ is defined by a Gaussian distribution

$$\mathbf{x}_k \sim \mathrm{b}_k = \mathcal{N}(\hat{\mathbf{x}}_k, \ \Sigma_{\mathbf{x}_k}) \qquad (1)$$

where $\mathrm{b}_k$ is referred to as the *belief* of $\mathbf{x}_k$ and is fully defined by mean $\hat{\mathbf{x}}_k$ and covariance $\Sigma_{\mathbf{x}_k}$. The set of all beliefs is called the belief space and denoted by $\mathcal{B}$. Intuitively, $\mathcal{B}$ is an uncertain representation of the state space $\mathcal{X}$. Mean $\hat{\mathbf{x}} \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ is the nominal state of the robot and evolves according to

$$\hat{\mathbf{x}}_{k+1} = f(\hat{\mathbf{x}}_k, \ \mathbf{u}_k) \qquad (2)$$

where $f : \mathcal{X} \times \mathcal{U} \rightarrow \mathcal{X}$ captures the nominal (known) dynamics of the robot, and $\mathbf{u}_k \in \mathcal{U} \subset \mathbb{R}^{n_u}$ is the system's control input. Covariance $\Sigma_{\mathbf{x}_k} \in \mathbb{R}_{>0}^{n_x \times n_x}$ describes the uncertainty

around the nominal robot state and evolves according to

$$\Sigma_{\mathbf{x}_{k+1}} = g\left(\Sigma_{\mathbf{x}_k}, \, \mathbf{u}_k\right) \tag{3}$$

where $g : \mathbb{R}^{n_x \times n_x} \times \mathcal{U} \to \mathbb{R}^{n_x \times n_x}_{>0}$ is the covariance function. Examples of Gaussian processes for robots with unicycle and fixed-wing dynamics are provided in Appendix A. Methods for modeling robots with (partially) unknown dynamics as Gaussian processes are discussed in [34] and [52].

### B. Environment Uncertainty

Some applications in robotics lack complete awareness of the environment, either because there is no information about the surroundings or due to 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 exteroceptive sensors such that it can autonomously explore the surroundings as it moves, i.e., to integrate into the map the obstacles when they are inside the sensor's detection range. Importantly, most sensors uniquely detect points on the boundary of a nearby obstacle.

This work assumes no uncertainty in the robot local observations denoted by $\mathbf{h}_k$. To transform this local observation from the robot frame to the global frame, let $\mathbf{h}_k \sim \mathcal{N}(\hat{\mathbf{h}}_k, \, 0)$. Bearing in mind that the robot's location might be uncertain with respect to the global frame $b_k \sim \mathcal{N}(\hat{\mathbf{x}}_k, \, \Sigma_{\mathbf{x}_k})$, the observed point is represented in the global frame as

$$b_O = b_k \oplus \mathbf{h}_k \tag{4}$$
$$= \mathcal{N}\left(\hat{\mathbf{x}}_k \oplus \hat{\mathbf{h}}_k, \, \mathbf{J}_{1\oplus}\Sigma_{\mathbf{x}_k}\mathbf{J}_{1\oplus}^T\right) \tag{5}$$

where $b_O \sim \mathcal{N}(\hat{\mathbf{x}}_O, \, \Sigma_{\mathbf{x}_O})$ is the result of the Gaussian relationships via a compounding operator $\oplus$ explained in Appendix B. From these uncertain points $\mathbf{x}_O$, the robot constructs a probabilistic map $\mathcal{M}$. Then, the obstacle occupancy probability for point $\mathbf{x} \in \mathcal{X}$ denoted by $F_{\mathcal{X}}(\mathbf{x})$ is the sum of the normally distributed densities in $\mathcal{M}$. The cumulative sum over all space $\mathcal{X}$ is called cumulative map and denoted by $F_{\mathcal{X}}$.

### C. Probabilistic Safety Guarantees

The system's and the environment's uncertainty 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 characterized by

$$p_{\text{collision}}(b_k, \, \mathcal{M}) = \int_{\mathcal{X}} b_k(\mathbf{x}) \, F_{\mathcal{X}}(\mathbf{x}) \, d\mathbf{x}$$
$$= \int_{\mathcal{X}} \mathcal{N}\left(\mathbf{x} \,|\, \hat{\mathbf{x}}_k, \, \Sigma_{\mathbf{x}_k}\right) F_{\mathcal{X}}(\mathbf{x}) \, d\mathbf{x} \tag{6}$$

where $F_{\mathcal{X}}(\mathbf{x})$ is the cumulative obstacle occupancy probability, as introduced in Section III-B. Then, given a minimum probability of safety $p_{\text{safe}}$, we require $1 - p_{\text{collision}}(b, \, \mathcal{M}) \geq p_{\text{safe}}$ for every belief $b$ on the trajectory in order to probabilistically guarantee the robot's safety.
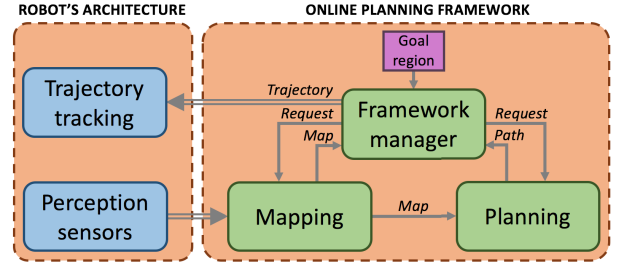


Fig. 1.  Framework for online mapping and motion planning under kinematic and uncertainty constraints.

### D. Planning Problem

Therefore, the planning problem considered in this work seeks a dynamically feasible trajectory in the belief space $\mathcal{B}$, which is probabilistically safe. Formally, let $\mathcal{B}_{\text{goal}} \subset \mathcal{B}$ denote the set of all belief states that correspond to the desired goal region $\mathcal{X}_{\text{goal}}$ in the environment as

$$\mathcal{B}_{\text{goal}} = \left\{ b \in \mathcal{B} \,\middle|\, \int_{\mathcal{X}_{\text{goal}}} b(\mathbf{x}) \, d\mathbf{x} \geq p_{\text{goal}} \right\} \tag{7}$$

where $p_{\text{goal}}$ is the minimum probability that a belief must satisfy for being considered to be in the goal region. Then, the constrained planning problem is to compute a sequence of controls $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{T-1} \in \mathcal{U}$ that result in a dynamically feasible trajectory $\xi : [0, \, T] \to \mathcal{B}$ for the robotic system described by (1), (2), and (3) such that $\xi(0) = b_{\text{start}} \in \mathcal{B}$, i.e., the system state at the beginning of the mission, $\xi(T) \in \mathcal{B}_{\text{goal}}$, and $1 - p_{\text{collision}}(\xi(t), \, \mathcal{M}) \geq p_{\text{safe}}$ for all $t \in [0, \, T]$.

## IV. FRAMEWORK FOR ONLINE NAVIGATION

This article presents a framework that endows a robotic system with the capability of safely navigating through unknown environments. This is achieved by means of online mapping and online motion planning of trajectories that meet motion and probabilistic constraints. The framework, depicted in Fig. 1, is threefold: 1) a mapping module that incrementally builds an uncertainty-aware map; 2) a planning module that continuously computes a safe and feasible trajectory toward the goal; and 3) a framework manager that coordinates the overall framework's execution. The remainder of this section describes the manager's strategy to control the interaction between the two core modules of the framework, i.e., the mapping (see Section V) and the planning (see Section VI). Note that, although the framework's description focuses on the online navigation challenge, the proposed online scheduling intrinsically solves the off-line motion planning problem.

The framework manager coordinates the mapping (MAPPER) and planning (PLANNER) modules according to the pipeline presented in Algorithm 1. This is, given the desired goal region $\mathcal{B}_{\text{goal}}$ and the required probabilistic safety guarantees $p_{\text{safe}}$, the manager conducts an iterative process until the system reaches the predefined goal region (line 7). An iteration consists of solving an updated version of the underlying motion planning problem that accounts for any alteration to the system's state and environment awareness.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

PAIRET *et al.*: ONLINE MAPPING AND MOTION PLANNING UNDER UNCERTAINTY FOR SAFE NAVIGATION

7

Each iteration starts by predicting a planning frame $R'$ that corresponds to the robot state at the time the current iteration solution will be available (line 8). Calculating a suitable planning frame is essential to guarantee the continuity and feasibility between consecutive plans. As the framework iterates each $\Delta T_{\mathrm{MP}}$ and it has full knowledge of the plan in execution *ongoing_traj*, calculating a planning frame that is suitable after $\Delta T_{\mathrm{MP}}$ is formulated as a state prediction problem. This is, given the current robot state $\mathbf{x}_R^W$ and the set of subsequent controls $\mathbf{u}$ involved in the execution of *ongoing_traj*, $\mathbf{x}_{R'}^W$ is computed by integrating (2) and (3) for the time-horizon $\Delta T_{\mathrm{MP}}$. Then, the manager retrieves, from the MAPPER, the current environment awareness as a cumulative map $F_{\mathcal{X}}^{R'}$ relative to $R'$ (line 9). Both the predicted planning frame $R'$ and the updated cumulative map $F_{\mathcal{X}}^{R'}$ are provided to the PLANNER (lines 10 and 11).

Before proceeding to solve the updated planning problem, the current plan in execution *ongoing_traj*, if any, is probabilistically checked for collision according to the current uncertainty-aware map $F_{\mathcal{X}}^{R'}$. In the event of *ongoing_traj* not being any longer valid, the framework manager dispatches to the robot the segment $\overline{ongoing\_traj}$ of *ongoing_traj* that is still safe (lines 12 and 13). This approach prevents the vehicle from stopping every time that a trajectory gets partially invalidated while ensuring its safety.

Finally, the PLANNER attempts to solve the planning problem by growing a new tree in $\mathcal{B}$ for a specific amount of time $\Delta T_{\mathrm{MP}}$ (line 14). The PLANNER tries to find a near-optimal trajectory that meets kinematic and probabilistic constraints within the allocated time budget $\Delta T_{\mathrm{MP}}$ and returns a *new_traj* if one is found (line 15). The newly found *new_traj* is uniquely dispatched to the robot when it fulfills the selection criteria defined in `satisfiesCriteria()` (sline 16–18). This work bases the selection criteria `satisfiesCriteria()` on the length of the trajectory; *new_traj* is dispatched if `length(new_traj) ≤ length(ongoing_traj)`, where `length(ongoing_traj)` $= \infty$ if *ongoing_traj* is partially invalidated, i.e., it does not reach the goal region $\mathcal{B}_{\mathrm{goal}}$.

Note that the computations in lines 8 and 9 are low demanding, and they can be scheduled in parallel to the main execution of the framework's pipeline. Therefore, the overall iteration rate of the framework is $1/\Delta T_{\mathrm{MP}}$, as solving the planning problem (line 14) is the unique process of the framework that requires a nonnegligible amount of time.

Given the nature of the problem of navigation in unknown environments, it may be possible that a feasible and probabilistically safe trajectory toward the goal region does not exist. Therefore, the framework is endowed with a contingency plan that attempts to return the vehicle nearby the deployment location $b_{\mathrm{start}}$. This contingency plan gets activated when the planner has not been able to find a solution in the last $n_{\mathrm{cp}}$ consecutive iterations, where $n_{\mathrm{cp}}$ is a user-defined safety value. In the event of the contingency plan getting activated, the MANAGER is reinitialized with the new planning problem. Note that, if the environmental awareness is highly uncertain, there might not exist a trajectory toward the new goal region. In this situation, not considering the previous map information for planning would allow the vehicle to move safely toward

---

**Algorithm 1** MANAGER($\mathcal{B}_{\mathrm{goal}}$, $p_{\mathrm{safe}}$)

1 **Input:**
2   $\mathcal{B}_{\mathrm{goal}}$: Goal region
3   $p_{\mathrm{safe}}$: Required probabilistic safety guarantees
4 **begin**
5     *ongoing_traj* $\leftarrow \varnothing$
6     PLANNER.loadProblem($\mathcal{B}_{\mathrm{goal}}$, $p_{\mathrm{safe}}$)
7     **while not** isGoalAchieved() **do**
        /* Predict planning frame     */
8         $R' \leftarrow$ pedictFrame(*ongoing_traj*)

        /* Retrieve cumulative map     */
9         $F_{\mathcal{X}}^{R'} \leftarrow$ MAPPER.getMap($R'$)

        /* Update planning problem     */
10        PLANNER.setNewFrame($R'$)
11        PLANNER.updateMap($F_{\mathcal{X}}^{R'}$)

        /* Check ongoing plan     */
12        **if not** PLANNER.isValid(*ongoing_traj*) **then**
13           dispatchPath($\overline{ongoing\_traj}$)

        /* Solve planning problem     */
14        PLANNER.solve($\Delta T_{\mathrm{MP}}$)

        /* Dispatch best valid plan     */
15        *new_traj* $\leftarrow$ planner.getSolution()
16        **if** satisfiesCriteria(*new_traj*) **then**
17           *ongoing_traj* $\leftarrow$ *new_traj*
18           dispatchPath(*ongoing_traj*)

---

the deployment location. In case no feasible motion plan is found to return to the deployment location, an emergency maneuver should be performed, e.g., coming to a complete stop for ground vehicles, going to the water surface for AUVs, and immediate landing for UAVs. Alternatively, one might consider ensuring the existence of a contingency plan at any time as in [22], but doing so under uncertainties is not trivial.

## V. INCREMENTALLY MAPPING UNKNOWN ENVIRONMENTS VIA LOCAL MAPS

Incrementally exploring the environment with a location-uncertain system leads to an uncertain representation of the surroundings. Under these conditions, obtaining a consistent and reliable representation of the entire environment is a challenging task commonly addressed with probabilistic inference approaches. These algorithms rely on gathering data from which distinctive features (landmarks) can be extracted and used to bound the uncertainty of the environment representation and system localization. Nonetheless, even for scenarios rich in features, there are always some residual uncertainties. Moreover, onboard perception sensors usually suffer from noises, which compromises the accuracy of the environment representation. All these issues motivate the need for an environment representation that jointly explains captures the

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

8                                                                                                                    IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING
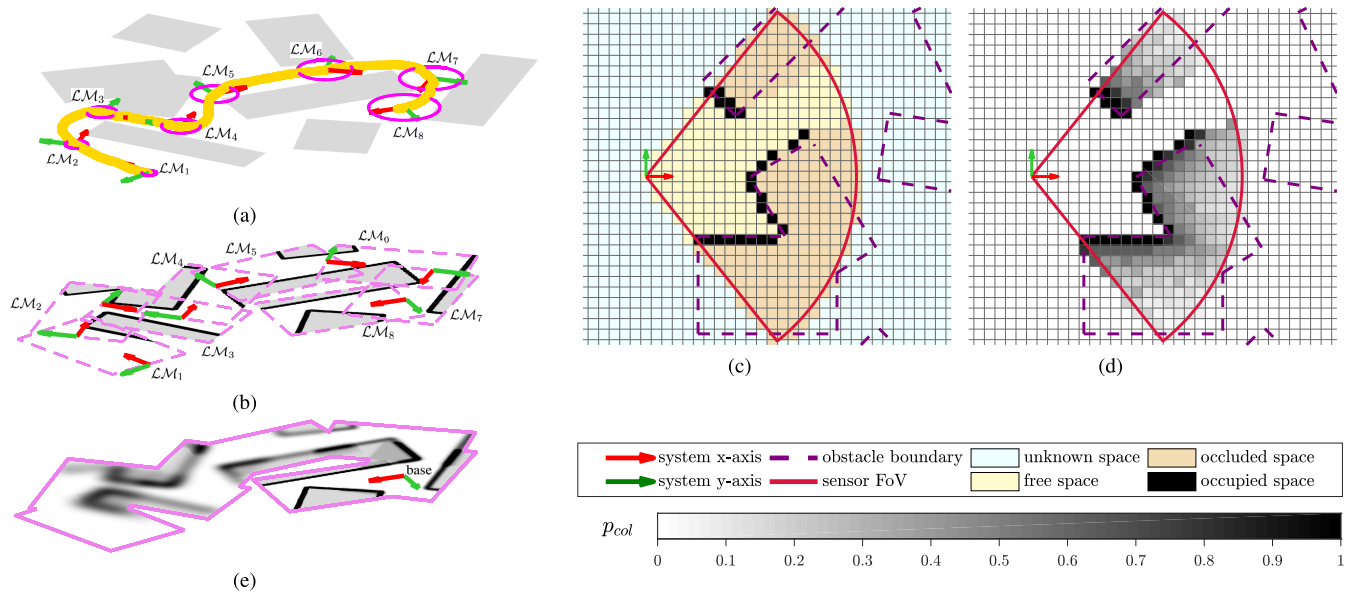


Fig. 2. Online mapping of the environment suitable for motion planning under uncertainty. (a) As the robot navigates through an unknown environment, (b) it builds a probabilistic map $\mathcal{M}$, which represents the surroundings as a set of local maps. Each local map is uncertain [magenta ellipses in (a)] with respect to the global frame. (c) and (d) Each local map is encoded as an occupancy grid map, which is built from a set of sequential sensor scans over a finite horizon time. (e) All local maps are fused into a cumulative map representation $F_{\mathcal{X}}^{R'}$ taking into account the relative uncertainty of each local map with respect to the predicted planning frame $R'$. (a) Environment. (b) Probabilistic map $\mathcal{M}$. (c) Semantic scene understanding. (d) Probabilistic scene understanding. (e) Cumulative map $F_{\mathcal{X}}^{R'}$. Note that our mapping suits 3-D scenes too and illustrations in 2-D for visualization purposes only.

uncertainty on the true obstacle's localization and the detection confidence according to the sensor model while being suitable for motion planning. In this work, such a representation is referred to as a probabilistic map.

This section details the undertaken mapping approach, which builds a set of local occupancy submaps whose base poses are uncertain with respect to a global frame (see Section V-A). Each submap is an occupancy grid map, which provides an efficient strategy to encode the incremental environment awareness (see Section V-B) and retrieve information about the environment occupancy (see Section V-C and Section V-D). This overall mapping strategy has proven to be suitable for real-time robotic mapping and planning applications in our previous work [55] and, despite being out of the scope of this manuscript, has also shown to be effective for online mapping and localization applications [29].

### A. Global Map as a Set of Local Submaps

There are different alternatives to represent the incremental knowledge of an environment, e.g., [3], [53], [66], [77]. The framework presented in this manuscript encodes the environment $\mathcal{M}$ via a set of $n$ local stochastic submaps [60], [61] due to its demonstrated efficiency on dealing with applications requiring real-time robotic localization, mapping, and planning [29], [55]. Formally, the local submaps method is defined as

$$\mathcal{M} = \{\mathcal{LM}_1, \ldots, \mathcal{LM}_n\} \tag{8}$$

$$\mathcal{LM}_i = \left\{ \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}, \ \hat{\mathbf{x}}_{\mathcal{LM}_i}^W, \ \Sigma_{\mathcal{LM}_i}^W \right\} \tag{9}$$

where each local submap $\mathcal{LM}_i$ contains a set of sequential sensor scans over a finite horizon time $\Delta T_{\mathcal{LM}}$. Within this

time period, all point coordinates $\mathbf{v}$ of the sensed environment are registered into the active submap $\mathcal{LM}_n$. The coordinate frame of $\mathcal{LM}_n$ is defined in a global frame $W$ by its estimated state $\mathbf{x}_{\mathcal{LM}_n}^W \sim \mathcal{N}(\hat{\mathbf{x}}_{\mathcal{LM}_n}^W, \Sigma_{\mathcal{LM}_n}^W)$. Importantly, such local registration assumes null uncertainty on observations, i.e., $\Sigma_{\mathbf{v}}^{\mathcal{LM}_n} = \mathbf{0} \ \forall \ \mathbf{v} \in \mathcal{LM}_n$. A new local submap $\mathcal{LM}_{n+1}$ is initiated every $\Delta T_{\mathcal{LM}}$ such that the accumulated localization error within the active local submap $\mathcal{LM}_n$ is low. In other words, the local mapping time horizon $\Delta T_{\mathcal{LM}}$ must be defined such that it always maintains the robot pose uncertainty $\Sigma_R^{\mathcal{LM}_n}$ within the active local map $\mathcal{LM}_n$ negligible.

The coordinate system of a new local submap $\mathcal{LM}_{n+1}$ is defined at the robot state estimate when $\mathcal{LM}_{n+1}$ is initiated, i.e., $\mathbf{x}_{\mathcal{LM}_{n+1}}^W = \mathbf{x}_R^W$. It is assumed that the robot starts building $\mathcal{LM}_{n+1}$ as soon as it finishes the $\mathcal{LM}_n$. Therefore, the robot state at the end of $\mathcal{LM}_n$ (defined as the last global robot state when building $\mathcal{LM}_n$) is the same as the global robot start state of $\mathcal{LM}_{n+1}$. For simplicity, the origin of the global map $W$ is chosen to be the same as the coordinate frame of the first local submap $\mathcal{LM}_1$, i.e., the robot's initial state.

Fig. 2 illustrates the concept of using local submaps to map the incremental knowledge about the environment. Particularly, the figure depicts a robot that has been navigating in an unknown environment, while, in the meantime, it has been encoding the perceived surrounding environment in a total of eight local submaps. Noteworthy, the example assumes open-loop navigation, i.e., without localization updates. Therefore, the first defined submaps are less uncertain with respect to the global frame $W$ than those built at a later stage. This fact corresponds to unbounded growth of the uncertainty on the system localization estimate.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

PAIRET *et al.*: ONLINE MAPPING AND MOTION PLANNING UNDER UNCERTAINTY FOR SAFE NAVIGATION

9

## B. Local Submap as Occupancy Grid Map

The assumption of null uncertainty on the robot pose within each local submap, also referred to as known robot poses, enables the representation of each local submap as an occupancy grid map. The chosen alternative to efficiently encode an occupancy grip map is via Octomaps [30]. Octomaps permits fusing range-based data into a probabilistic voxel representation, which generates an occupancy grid map with adjustable resolution. Octomaps store the information in an octree data structure, which provides fast access time while, at the same time, optimizing memory usage. All these desirable features make the undertaken mapping strategy ideal for online mapping and planning.

The probabilistic sensor fusion within an occupancy grid map is performed as an Octomap [30], [50]. This is, the probability $P(\mathbf{v}|\mathbf{h}_{1:k})$ of a cell $\mathbf{v}$ to be occupied given a set of sensor measurements $\mathbf{h}_{1:k}$ is estimated as

$$P(\mathbf{v}|\mathbf{h}_{1:k}) = \left[1 + \frac{1-P(\mathbf{v}|\mathbf{h}_k)}{P(\mathbf{v}|\mathbf{h}_k)}\frac{1-P(\mathbf{v}|\mathbf{h}_{1:k-1})}{P(\mathbf{v}|\mathbf{h}_{1:k-1})}\frac{1-P(\mathbf{v})}{P(\mathbf{v})}\right]^{-1} \tag{10}$$

where $P(\mathbf{v}|\mathbf{h}_k)$ is the inverse sensor model characterizing the sensor used for mapping and $P(\mathbf{v}|\mathbf{h}_{1:k-1})$ is the preceding estimate given all historical measurements. Using log-odds notation

$$L(\cdot) = \log\left[\frac{P(\cdot)}{1-P(\cdot)}\right] \tag{11}$$

and under the common assumption of a uniform (noninformative) prior, i.e., $P(\mathbf{v}) = 0.5$, (10) is simplified to

$$L(\mathbf{v}|\mathbf{h}_{1:k}) = L(\mathbf{v}|\mathbf{h}_{1:k-1}) + L(\mathbf{v}|\mathbf{h}_k). \tag{12}$$

To change the state of a node $\mathbf{v}$, (12) requires as many observations as the ones used to define its current state. This overconfidence in the map is addressed as in [79] by using a clamping policy to ensure that the confidence in the map remains bounded

$$\begin{aligned}L(\mathbf{v}|\mathbf{h}_{1:k}) &= [L(\mathbf{v}|\mathbf{h}_{1:k})]_{l_{\min}}^{l_{\max}}\\ &= \max(\min(L(\mathbf{v}|\mathbf{h}_{1:k}), l_{\max}), l_{\min})\end{aligned} \tag{13}$$

where $l_{\min}$ and $l_{\max}$ denote lower and upper bounds on log-odds values. As a consequence, the model of the environment remains updatable [30].

The measurement update rules in (12) and (13) can be used with any kind of distance sensor, as long as the inverse sensor model is available. Our framework employs the extended beam-based inverse sensor model depicted in Fig. 2(c). This model assumes that: 1) the line of sight between the sensor origin and the endpoint of measurement does not contain any obstacle (free space); 2) endpoints correspond to obstacle surfaces (occupied space); and 3) the line continuing beyond the endpoint until the maximum sensor range is likely to be occupied by the observed obstacle (occluded space). Then, the extended ray-casting operation to update each voxel $\mathbf{v}$ from the sensor origin to the maximum sensor range is performed

using the following log-odds inverse sensor model:

$$L(\mathbf{v}|\mathbf{h}_t) = \begin{cases} l_{\text{free}}, & \text{if } \mathbf{v} \text{ is traversed by the beam}\\ l_{\text{occ}}, & \text{if } \mathbf{v} \text{ is hit by the beam}\\ l_{\text{ocl}}, & \text{if } \mathbf{v} \text{ is between the hit and sensor range} \end{cases} \tag{14}$$

where $l_{\text{free}}$ and $l_{\text{occ}}$ are constants determined according to the sensor model, and $l_{\text{ocl}}$ penalizes occluded zones according to the decaying function

$$l_{\text{ocl}} = \gamma^d \, l_{\text{occ}} \tag{15}$$

where, for a decay rate $\gamma \in [0, 1]$, $l_{\text{ocl}}$ decreases $\gamma$ times for each unit of $d$, which is the distance from the measurement endpoint. This corresponds to $l_{\text{ocl}} = l_{\text{occ}}$ for $d = 0$, i.e., in the hit point, and to $l_{\text{ocl}} \to 0$, i.e., to a noninformative $P(\mathbf{v}) = 0.5$, as $d \to \infty$. The maximum expand of the occluded region is as far as the sensor range.

## C. Map Fusion and Single Point Query

An occupancy query to the current probabilistic map $\mathcal{M}$ is done by converting the given query into multiple local queries. The occupancy probability values at each local submap can be fused together by means of the log-odds update rule in (12) with the corresponding clamping operation in (13). These operations apply because combining measurements from multiple local submaps is a similar operation as combining multiple measurement updates in a single global map [29].

Without loss of generality, assume that an occupancy query at position $\hat{\mathbf{x}}^Y$ is performed from an uncertain coordinate frame $Y$ with known pose estimate $\mathbf{x}_Y^W \sim \mathcal{N}(\hat{\mathbf{x}}_Y^W, \Sigma_Y^W)$. This global query corresponds to the multiple local log-odds occupancy queries

$$L(\hat{\mathbf{x}}^Y) = \sum_{i=1}^{n}\left[L_{1:i-1}(\hat{\mathbf{x}}^Y) + L_i(\mathbf{x}^{\mathcal{LM}_i})\right]_{l_{\min}}^{l_{\max}} \tag{16}$$

where $L_{1:i-1}(\hat{\mathbf{x}}^Y)$ is the accumulative log-odd estimate from the precedent $i-1$ local submaps with $L_{1:i-1}(\hat{\mathbf{x}}^Y) = 0$ for $i = 1$, $L_i(\cdot)$ implies that the log-odds lookup is done in the local submap $\mathcal{LM}_i$, and $\mathbf{x}^{\mathcal{LM}_i} \sim \mathcal{N}(\hat{\mathbf{x}}^{\mathcal{LM}_i}, \Sigma_Y^{\mathcal{LM}_i})$ corresponds to $\hat{\mathbf{x}}^Y$ in local coordinates. $\mathbf{x}^{\mathcal{LM}_i}$ is calculated via the linear estimation of known spatial relationships

$$\mathbf{x}^{\mathcal{LM}_i} = \ominus\mathbf{x}_{\mathcal{LM}_i}^W \oplus \left(\mathbf{x}_Y^W \oplus \mathbf{x}^Y\right) \tag{17}$$

where $\oplus$ denotes the compounding operation and $\ominus$ corresponds to its inverse relation, as commonly used to simplify notation when calculating spatial transformations (see Appendix B for a brief introduction and [70] for a full review).

Given that $\hat{\mathbf{x}}^Y$ in local coordinates follows a probabilistic distribution, the local occupancy query $L_i(\mathbf{x}^{\mathcal{LM}_i})$ is

$$P_i(\mathbf{x}^{\mathcal{LM}_i}) = \sum_{\mathbf{v}\in\mathcal{LM}_i} P(\mathbf{v})\,\mathcal{N}\left(\mathbf{v}\mid\hat{\mathbf{x}}^{\mathcal{LM}_i}, \Sigma_Y^{\mathcal{LM}_i}\right) \tag{18}$$

where $\mathbf{v}$ represents the set of voxels in submap $\mathcal{LM}_i$ and $P_i(\mathbf{x}^{\mathcal{LM}_i})$ can be described in log-odds $L_i(\mathbf{x}^{\mathcal{LM}_i})$ notation via the log-odds transform.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

10                                                                                          IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING

## D. Computation of the Cumulative Map $F_{\mathcal{X}}^{R'}$

Section V-C provides a strategy to query the occupancy probability $P(\mathbf{x})$ of a single point coordinate $\mathbf{x} \in \mathcal{X}$. Our previous work [55] demonstrated that this approach is suitable for the requirements of an online planner under probabilistic constraints. However, bearing in mind that each planning cycle requires numerous queries of $P(\mathbf{x})$ involving different $\mathbf{x}$, the overall planner performance can be enhanced by computing the map fusion before the planning time budget starts.

The probabilistic map fusion over all state space $\mathcal{X}$ is described by the cumulative distribution $F_{\mathcal{X}}$ over the local density distributions of the sensed environment.[3] In particular, for the online planning problem, it is of interest to fuse the map information with respect to the predicted planning frame $R'$ such that the cumulative map $F_{\mathcal{X}}^{R'}$ reflects the relative uncertainty between the current environmental awareness and the planning frame $R'$. Fig. 2(e) illustrates the extraction of $F_{\mathcal{X}}^{R'}$ from a set of local maps. Computing $F_{\mathcal{X}}^{R'}$ implies that the computational requirements of retrieving $P(\mathbf{x}^{R'})$ during planning time are reduced to those of a lookup table in the cumulative map $F_{\mathcal{X}}^{R'}$.

Subject to the log-odds transformation, $F_{\mathcal{X}}^{R'}$ is computed by rewriting (16) and (18) as

$$L(\hat{\mathcal{X}}^{R'}) = \sum_{i=1}^{n} \left[ L_{1:i-1}(\hat{\mathcal{X}}^{R'}) + L_i(\mathcal{X}^{\mathcal{LM}_i}) \right]_{l_{\min}}^{l_{\max}} \quad (19)$$

where $L_{1:i-1}(\hat{\mathcal{X}}^{R'})$ is the accumulative log-odd estimate from the precedent $i-1$ local submaps with $L_{1:i-1}(\hat{\mathcal{X}}^{R'}) = 0$ for $i = 1$, $L_i(\cdot)$ is the log-odds lookup done in the local occupancy submap $\mathcal{LM}_i$, and $\mathcal{X}^{\mathcal{LM}_i} \sim \mathcal{N}(\hat{\mathcal{X}}^{\mathcal{LM}_i}, \Sigma_{R'}^{\mathcal{LM}_i})$ corresponds to the state space $\hat{\mathcal{X}}^{R'}$ in local coordinates defined as

$$\mathcal{X}^{\mathcal{LM}_i} = \ominus \mathcal{X}_{\mathcal{LM}_i}^{W} \oplus \left( \mathcal{X}_{R'}^{W} \oplus \mathcal{X}^{R'} \right). \quad (20)$$

Then, the occupancy probability $L_i(\mathcal{X}^{\mathcal{LM}_i})$ at $\mathcal{LM}_i$ for all $\mathbf{x} \in X^{\mathcal{LM}_i}$ is computed as

$$P_i(\mathcal{X}^{\mathcal{LM}_i}) = \sum_{\mathbf{v} \in \mathcal{LM}_i} P(\mathbf{v}) \, \mathcal{N}\left( \mathbf{v} \mid \hat{\mathbf{x}}, \Sigma_{R'}^{\mathcal{LM}_i} \right) \quad \forall \, \mathbf{x} \in \mathcal{X}^{\mathcal{LM}_i}$$

$$= \mathcal{LM}_i \otimes \mathcal{K}_{\alpha}\left( \Sigma_{R'}^{\mathcal{LM}_i} \right) \quad (21)$$

where $\mathbf{v}$ represents the set of voxels in submap $\mathcal{LM}_i$ and $\mathcal{K}_{\alpha}(\cdot)$ with confidence level $\alpha = 1$ is a kernel representing the discrete version of a Gaussian distribution over the entire span of the local submap $\mathcal{LM}_i$ (see Appendix C). $\otimes$ is the correlation operator, i.e., a sliding inner product, and $P_i(\mathcal{X}^{\mathcal{LM}_i})$ can be described in log-odds $L_i(\mathcal{X}^{\mathcal{LM}_i})$ via the log-odds transform.

Interestingly, the underlying computation of $F_{\mathcal{X}}^{R'}$ is the correlation operator $\otimes$, a common technique for which there exist efficient implementations. On top of that, the independence between local submaps allows parallelizing (21) for each $\mathcal{LM}_i$ in different threads. Ideally, this process could be scheduled

[3]Only those voxels describing the known environment, i.e., free, occluded, and occupied space, are considered in the computation of $F_{\mathcal{X}}$. Considering the unknown space with its $P(\mathbf{v}) = 0.5$ in the computations would lead to a cumulative map with misleadingly overestimating occupancy probabilities.

such that $F_{\mathcal{X}}^{R'}$ is ready before the planning time budget starts.

## VI. MULTILAYERED MOTION PLANNING UNDER ENVIRONMENT AND MOTION UNCERTAINTY

The planning problem defined in Section III-D has three main requirements: 1) to consider the vehicle's motion constraints; 2) to validate probabilistic constraints in face of uncertainties; and 3) to meet online computation limitations. Our previous approach successfully addressed all these requirements formulating a single-layered sampling-based planning strategy in the belief space [55]. The planner in question: 1) samples feasible states in the system's state space and 2) extends and validates the tree of motions in the belief space. This approach proved to be suitable for solving online motion planning problems in challenging real-world scenarios, but its applicability was limited to low-dimensional planning problems given the huge search space and the computational burden of all considered constraints.

As discussed in Section II, multilayered planning strategies enable online planning in high-dimensional spaces. This motivates the use of such an idea to extend our framework's capabilities to suit the planning requirements of a larger group of robotic systems and environments. Principally, the extended planning strategy employs a multilayered planning scheme (see Section VI-A) to overcome the aforementioned scalability issues. Such a strategy allows deferring the computation of kinematic constraints (see Section VI-B) and probabilistic constraints (see Section VI-C) after identifying some subregions of the system's state space that potentially contain a solution to the planning problem.

### A. Multilayered Motion Planning

The capabilities of our previous planner (hereinafter referred to as the constrained planner) are extended to deal with problems of higher dimensionality by means of a multilayered planning strategy. As schematized in Fig. 3, the proposed strategy adopts a sequential two-layered planning scheme consisting of a lead planner and the constrained planner. The lead planner seeks to determine a subregion $\mathcal{X}' \subset \mathcal{X}$ of the entire state space that eases and, consequently, speeds up the search of the final trajectory $\xi$, which accounts for all considered constraints (see Section III-D). To this aim, the multilayered scheme is designed as follows.

1) *Lead Planner:* It employs the RRT* to rapidly find a path in the workspace $\mathcal{W}$. The computed lead path is a nearly optimal geometric solution $\xi' \in \mathcal{W}$ used to determine $\mathcal{X}'$ via the lifting operator $\mathrm{lift} : \mathcal{W} \to \mathcal{X}$ detailed in the following.
2) *Constrained Planner:* It leverages the delimited search space $\mathcal{X}'$ and the SST in [55] to rapidly compute the final solution $\xi$, which meets kinodynamic (see Section VI-B) and probabilistic safety constraints (see Section VI-C).

Although the planners within the multilayered planning scheme could be different, the selection above suits the online requirements of our framework. This is, the framework's overall planning time $\Delta T_{\mathrm{MP}}$ is divided as $\Delta T_{\mathrm{MP}} = \Delta T_L + \Delta T_C$,
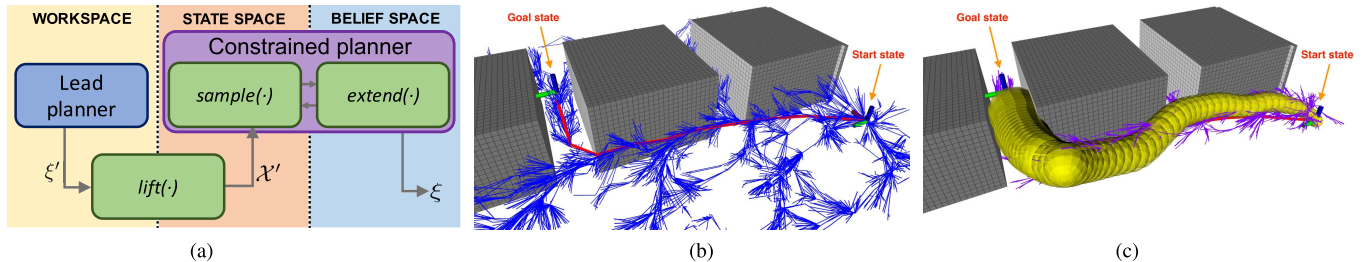
Fig. 3. (a) Multilayered motion planning framework: (b) lead planner—RRT* shown in blue computes a geometric lead $\xi'$ (red path) to guide the search space $\mathcal{X}'$ of (c) constrained planner—SST shown in magenta. The resulting trajectory $\xi$ with its uncertainty (yellow funnel) satisfies kinodynamic and probabilistic safety constraints.

where $\Delta T_L$ and $\Delta T_C$ are the time budgets allocated to the lead and constrained planners, respectively. Then, given our selection of planners, the assignment of time budgets allows $\Delta T_L \ll \Delta T_C$ as the lead planner is adept at providing quickly a suitable lead path such that the constrained planner has at its disposal most of the time budget $\Delta T_L \approx \Delta T_{\mathrm{MP}}$ to refine the final trajectory, which accounts for all the considered constraints.

Given our selection of planners and their operational space, the designed multilayered planning scheme requires the lifting lift : $\mathcal{W} \to \mathcal{X}$. A common lift($\cdot$) strategy is to define $\mathcal{X}'$ as a tube around $\xi'$ with radius $d$ for the geometric components of the state space, whereas the nongeometric components are left unbounded, e.g., [56], [57], [76]. The performance of this approach, however, is susceptible to the parametrization of $d$; tight search spaces, i.e., small radius $d$, promote final solutions with lower cost than those obtained with bigger radius $d$. On top of that, relying on a fixed $d$ requires hand-tuning such parameter to ensure that the final solution lies within $\mathcal{X}'$; if $\mathcal{X}'$ does not contain the final solution, the planner will lack probabilistic completeness. Adjusting $d$ to ensure probabilistic completeness would prove to be a cumbersome task since the type of environment and planning constraints, among many other factors, should be taken into account.

Different from other multilayered planning schemes in the literature, ours uses a method of information interchange between planners that maintains the completeness and asymptotic optimality properties of the constrained planner when used in a standalone fashion [55]. This work builds on the idea of sampling around a lead path to present alternative definitions of $\mathcal{X}'$ via the lift($\cdot$) operator. In particular, the designed MLP exploits a mixture of samplers to trade off the low-cost trajectories found when sampling around a lead path and the probabilistic completeness of uniform sampling. This article proposes two mixtures of sampling techniques.

1) *Bias to Rigid $\mathcal{X}'$:* Given a fixed radius $d$, the planner samples uniformly in $\mathcal{X}'$ with probability $p$ and uniformly over the space with probability $1 - p$.

2) *Adaptive $\mathcal{X}'$:* The planner adjusts $d$ within the range of a strictly guided sampling to a uniform search. Adjusting $d$ can be conducted via some heuristics or as an optimization problem subject to a cost function.

The performance of both approaches in comparison to a rigid $\mathcal{X}'$ strategy is discussed in Section VII-B. Noteworthy,
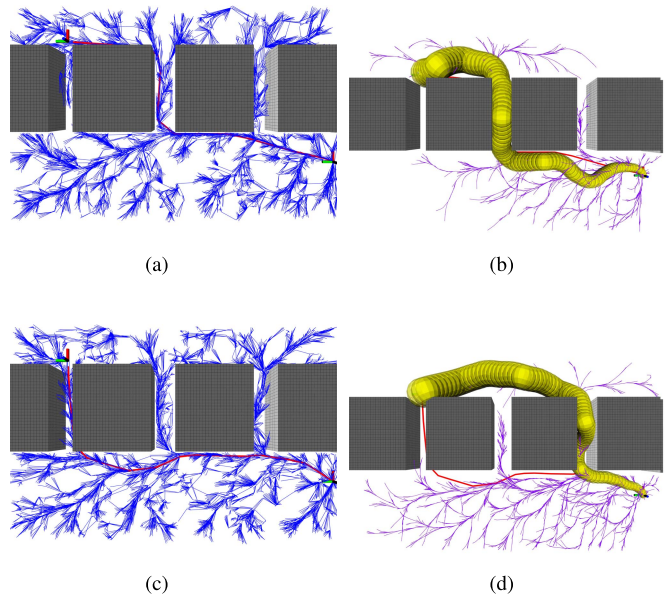


Fig. 4. Probabilistic completeness of the proposed multilayered planning scheme with adaptive lead, which (a) and (b) promotes finding the final solution in the neighborhood of the asymptotically optimal lead $\xi'$ (red), while it (c) and (d) preserves completeness guarantees even when the lead $\xi'$ transverses a corridor, which does not offer a probabilistic safe passage. (a) Lead planner—RRT*. (b) Constrained planner–SST. (c) Lead planner—RRT*. (d) Constrained planner–SST.

any of the two presented mixtures of sampling strategies ensures probabilistic completeness of the overall multilayered scheme. As an extreme example, let us consider the scenario depicted in Fig. 4(c) and (d), where the lead planner finds an asymptotically optimal solution through the farthest (most left) corridor. However, according to the probabilistic safety constraints defined in Section VI-C, such a corridor does not offer any safe passage. Despite the initial bias toward this unsuitable $\mathcal{X}'$, a mixture of sampling strategies, as the ones introduced in this section, permits finding a solution if one exists provided enough time, thus ensuring probabilistic completeness guarantees.

## B. Planning Under Motion Constraints

The system's motion capabilities are considered in the constrained planner by expanding a tree with the system's motion model (2) and (3). In particular, the constrained planner employs the SST algorithm [41] to build a tree in the

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

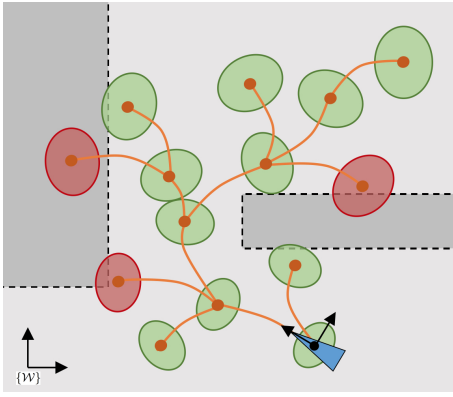12 IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING



Fig. 5. Tree expansion under motion and probabilistic constraints. The state beliefs (nodes) of the tree are obtained by considering the motion capabilities. The ellipses surrounding the states represent their uncertainty, where green corresponds to those states satisfying the probabilistic safety constraints and red corresponds to those that do not.

belief space with state beliefs $x \sim b = \mathcal{N}(\hat{x}, \Sigma_x)$ as nodes. The tree expansion is based on two procedures: *sample*($\cdot$) and *extend*($\cdot$), which are conducted in the state and belief space, respectively [see Fig. 3(a)]. That is, *sample*($\cdot$) draws a random state $x_{\text{rand}} \in \mathcal{X}'$, where $\mathcal{X}'$ is a subregion of $\mathcal{X}$ as defined in Section VI-A. The planner then selects a node from the tree to attempt connecting to the randomly sampled state $x_{\text{rand}}$. Such a selection is conducted via nearest neighbor in the state space using the Euclidean metric. The selected node $x_{\text{near}}$ has a probabilistic representation in the belief space, i.e., $x_{\text{near}}$ is better described as $x_{\text{near}} \sim b_{\text{near}} = \mathcal{N}(\hat{x}_{\text{near}}, \Sigma_{x_{\text{near}}})$. Then, from this belief, the *extend*($\cdot$) procedure expands the tree in the belief space by evolving the system's motion model (2) and (3) with a randomly sampled control input $\mathbf{u} \in \mathcal{U}$. This expansion is done for a random period of time $T_{\text{prop}}$. Since the considered motion model includes the system's uncertainty, each obtained belief (tree node) corresponds to a vehicle's state with its associated uncertainty (see Fig. 5).

### C. Planning Under Probabilistic Constraints

In our approach, the environmental awareness relative to the current robot state is represented as a cumulative distribution on discrete support $F_{\mathcal{X}}^{R'}$ (see Section V). As discussed previously, this representation of the environment favors efficiency for online mapping and planning applications. In fact, we leverage such encoding to guarantee $1 - p_{\text{collision}}(b, \mathcal{M}) \geq p_{\text{safe}}$ for each belief $b$ of the tree as

$$1 - \left( p_{\text{collision},\alpha}\left(b, F_{\mathcal{X}}^{R'}\right) + (1 - \alpha) \right) \geq p_{\text{safe}} \quad (22)$$

$$\alpha - p_{\text{collision},\alpha}\left(b, F_{\mathcal{X}}^{R'}\right) \geq p_{\text{safe}} \quad (23)$$

where $\alpha$ is the confidence level on the computation of $p_{\text{collision},\alpha}(\cdot) \in [0, \alpha]$. In other words, $p_{\text{collision},\alpha}(\cdot)$ does not cover a $(1 - \alpha)$ span of the belief $b$ over the state space. Therefore, it is assumed that the remaining $(1 - \alpha)$ is in collision to ensure probabilistic guarantees on the collision checking decision. All in all, this method can be exploited to trade a constant conservatism $\alpha$ in favor of performance.

Then, the probability of collision of a robot centered belief $b^{R'} \sim \mathcal{N}(\hat{b}^{R'}, \Sigma_b^{R'})$ with the environment is calculated as

$$p_{\text{collision},\alpha}\left(b^{R'}, F_{\mathcal{X}}^{R'}\right) = \left\langle \mathcal{K}_\alpha\left(\Sigma_b^{R'}\right), F_{\mathcal{X}}^{R'} \right\rangle_F$$
$$= \text{vec}\left(\mathcal{K}_\alpha\left(\Sigma_b^{R'}\right)\right)^T \text{vec}\left(F_{\mathcal{X}}^{R'}\right) \quad (24)$$

where $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product of the overlapping region between the $\hat{b}^{R'}$-centered discrete state belief $\mathcal{K}_\alpha(\Sigma_{b_k}^{R'})$ (see Appendix C) and the cumulative environment awareness $F_{\mathcal{X}}^{R'}$. The Frobenius inner product is an efficient operation via matrix vectorisation.

The overall proposed MLP leads to the exploration tree depicted in Fig. 5, whose edges account for the vehicle's kinodynamic capabilities and whose nodes are probabilistically safe subject to the system's localization, motion, and environment uncertainties. In addition, the expansion of the tree is also subject to states not leading to an inevitable collision, i.e., a state must allow for the vehicle to make a full stop before colliding.

## VII. EXPERIMENTAL EVALUATION

The proposed framework has been implemented in ROS and uses the facilities provided by Octomap [30] and the OMPL [72] as the core building block of the proposed mapping and planning strategies, respectively. This implementation is used to evaluate thoroughly the different proposed features and the framework as a whole. This section reports the results of such analysis in an incremental fashion. First, Section VII-A presents a discussion on the capabilities of our framework's precedent version in simulated and real-world scenarios. Then, Section VII-B and Section VII-C report the performance of the key components of the newly proposed framework, i.e., the multilayered scheme and the probabilistic collision checking. The potential of these components is individually evaluated against closely related state-of-the-art approaches. Finally, in Section VII-D, the capabilities of the new framework are demonstrated in a challenging scenario.

### A. Navigation in Unknown 2-D Environments

An early version of the work presented in this article has appeared before [55]. This consisted of a simpler framework that evaluated all uncertainties on the fly (in contrast to the proposed cumulative map encoding) while exploring the belief space via a single-layered planner (in contrast to the proposed multilayered guided exploration). As reported next, our precedent work proved to be suitable for safe robot navigation, but its computational requirements limited its applicability to low-dimensional spaces.

The precedent framework has been deployed on the Sparus II autonomous underwater vehicle (AUV) (see Fig. 6), a nonholonomic torpedo-shaped vehicle with hovering capabilities [10]. To meet the limitations of our precedent work, the AUV is limited to operate at a constant depth, i.e., in SE(2). Under these conditions, the motion model of the Sparus II can be approximated by a unicycle system, as detailed in Appendix A-A. The AUV is equipped with a mechanical scanned imaging sonar (MSIS) to perceive the

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

PAIRET *et al.*: ONLINE MAPPING AND MOTION PLANNING UNDER UNCERTAINTY FOR SAFE NAVIGATION                                                                                  13
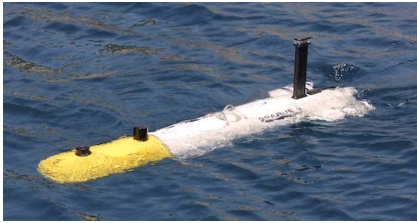


Fig. 6.    Sparus II AUV, a nonholonomic vehicle.

surroundings and incrementally map the environment. We use the default parameters in [30] of $l_{\text{free}} = -0.4$ $(P(\mathbf{v}) = 0.4)$, $l_{\text{occ}} = 0.85$ $(P(\mathbf{v}) = 0.7)$, $l_{\text{min}} = -2$ $(P(\mathbf{v}) = 0.12)$, and $l_{\text{max}} = 3.5$ $(P(\mathbf{v}) = 0.97)$. The decay rate in (15) is set to $\gamma = 0.8$. The framework's planning time is set to $\Delta T_{\text{MP}} = 1.5$ s.

The test bed to evaluate the precedent framework consists of two environments located in Sant Feliu de Guíxols (Spain): 1) breakwater structure that is composed of a series of concrete blocks (14.5-m long × 12-m width), which are separated by 4-m gaps [see Fig. 7(a) and (b)] and 2) rocky formations that create an underwater canyon of 28-m long [see Fig. 8(a) and (b)]. Using these environments, two experiments are reported: 1) evaluation of the overall performance of the framework in the underwater simulator (UWSim) [65] and 2) validation of the framework in real in-water trials. Experiment 1) is conducted in both environments, while experiment 2) is uniquely tested in the real breakwater structure scenario.

*1) Simulated Trials:* The framework is exhaustively tested in the simulated breakwater structure and canyon scenarios with 20 attempts per scenario (total of 40 trials).

In the breakwater structure, 19 start-to-goal queries are successfully solved. Among those successful experiments, the robot achieved the goal region $\mathcal{B}_{\text{goal}}$ by crossing through the first 1-m gap in 17 occasions, while, in the remaining two trials, the planner found a less optimal trajectory through the second 4-m passage. Fig. 7 depicts the mission execution in one of those trials. In the initial part of the mission, as the environment is completely undiscovered, the computed trajectory goes straight to the goal [see Fig. 7(c)]. As soon as the trajectory gets invalidated, a new collision-free trajectory is computed [see Fig. 7(d)]. After some mapping–planning iterations, the robot gets out of the 4-m gap between two blocks [see Fig. 7(e)]. On average, the computed trajectories toward the goal have a length of approximately 45.2 m and are completed in 2′21″.

All 20 start-to-goal queries in the simulated canyon scenario are successfully solved. The higher success rate with respect to the previous experiment is given by the nature of the environment; this scenario involves less abrupt maneuvers, and the passage is wider, more than twice larger though. Fig. 8(d) depicts one of the trajectories calculated through the narrow passage of the canyon. On average, the calculated trajectories toward the goal have a length of approximately 58.4 m and are completed in 2′59″.

*2) Real-World Trials:* In-water experiments are conducted in the real breakwater structure located in Sant Feliu de Guíxols (Spain). Similar to the simulated trials, the robot is required to solve a start-to-goal-query to reach a goal region $\mathcal{B}_{\text{goal}}$ located on the opposite side of the structure, which can only be achieved by navigating through any of the narrow 4-m gaps. A total of five start-to-goal queries are attempted. During those autonomous missions, the vehicle is connected to a wireless access point buoy for monitoring purposes; all components of the framework run on the robot to prove the framework's suitability for real-world robots with limited onboard computation power.

In all five trials, the framework was successful in finding and driving the Sparus II AUV toward the desired goal region $\mathcal{B}_{\text{goal}}$ through one of the narrow gaps in the breakwater structure.[4] In four trials, the trajectory was found through the first corridor, while, in the other trial, the robot went through the second gap. Fig. 9 depicts Sparus II in one of those in-water trials and the trajectory calculated toward the goal, which has a length of 57.9 m and took 3′07″.

### B. Multilayered Planning Scheme

The multilayered planning scheme presented in Section VI-A is one of the key features allowing us to overcome the scalability issues of our previous single-layered planner [55]. Nonetheless, different from current multilayered approaches that rely on rigid definitions of the search space $\mathcal{X}'$ (rigid-$\mathcal{X}'$), this article explores two alternative definitions of $\mathcal{X}'$ (biased-$\mathcal{X}'$ and adaptive-$\mathcal{X}'$) based on a mixture of sampling experts. This section reports the performance of these four strategies in the following belief space planning problem: reaching the state between the blocks in Fig. 10 while satisfying kinodynamic and probabilistic safety constraints subject to a $p_{\text{safe}} = 0.99$ minimum safety probability bound. In this evaluation, the entire 3-D environment is considered to be known in advance, and the system dynamics are approximated as described in Appendix A-B.

The four methods [single-layered planner (SLP) and MLP with rigid-$\mathcal{X}'$, biased-$\mathcal{X}'$, and adaptive-$\mathcal{X}'$] are evaluated for their ability to quickly find a solution and for the cost of the resulting trajectory. The given total planning time budget is set at $\Delta T_{\text{MP}} = 1.5$ s to emulate online planning requirements, which is distributed as $\Delta T_L = 0.3$ s and $\Delta T_C = 1.2$ s for the three multilayered schemes. With this setup, each planner attempts to solve the defined planning problem a total of 2000 times.

Fig. 11 depicts the number of successfully solved trials and the resulting trajectory length when considering a rigid-$\mathcal{X}'$ lead with radius parameterisations $d \in [0, 40]$ m. While $d = 0$ m strictly limits the search space to those states on the lead path, $d = 40$ m spans the search over all state space of the defined planning problem, therefore resembling uniform sampling. As it can be observed, small search spaces (small $d$) endanger the planner's ability to find a solution with limited time. However, when a trajectory is found, the resulting cost is lower than those solutions found with wider $\mathcal{X}'$ leads. Instead, these wide search spaces (big $d$) make the planner struggle at solving

---

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

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

14

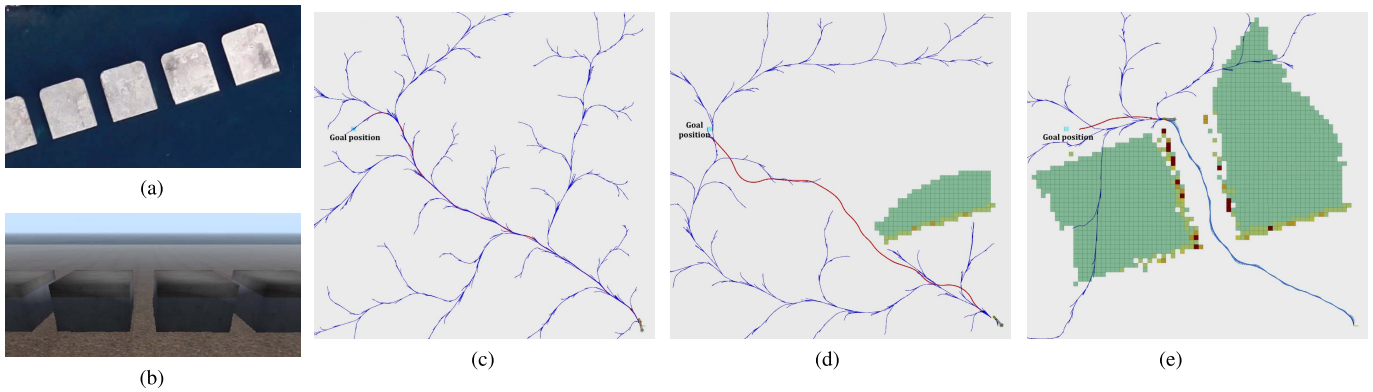IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING



Fig. 7. Incrementally mapping and planning in the undiscovered breakwater structure scenario. (c) Initial state of the Sparus II AUV in the unknown environment with the initially found trajectory (red). (d) Anytime the trajectory is invalidated, a new collision-free trajectory is computed. (e) After some iterations, the robot gets out of the 4-m gap between blocks. (a) Real breakwater. (b) Simulated breakwater. (c) Initial empty map. (d) Replanned trajectory. (e) Final part of the survey.
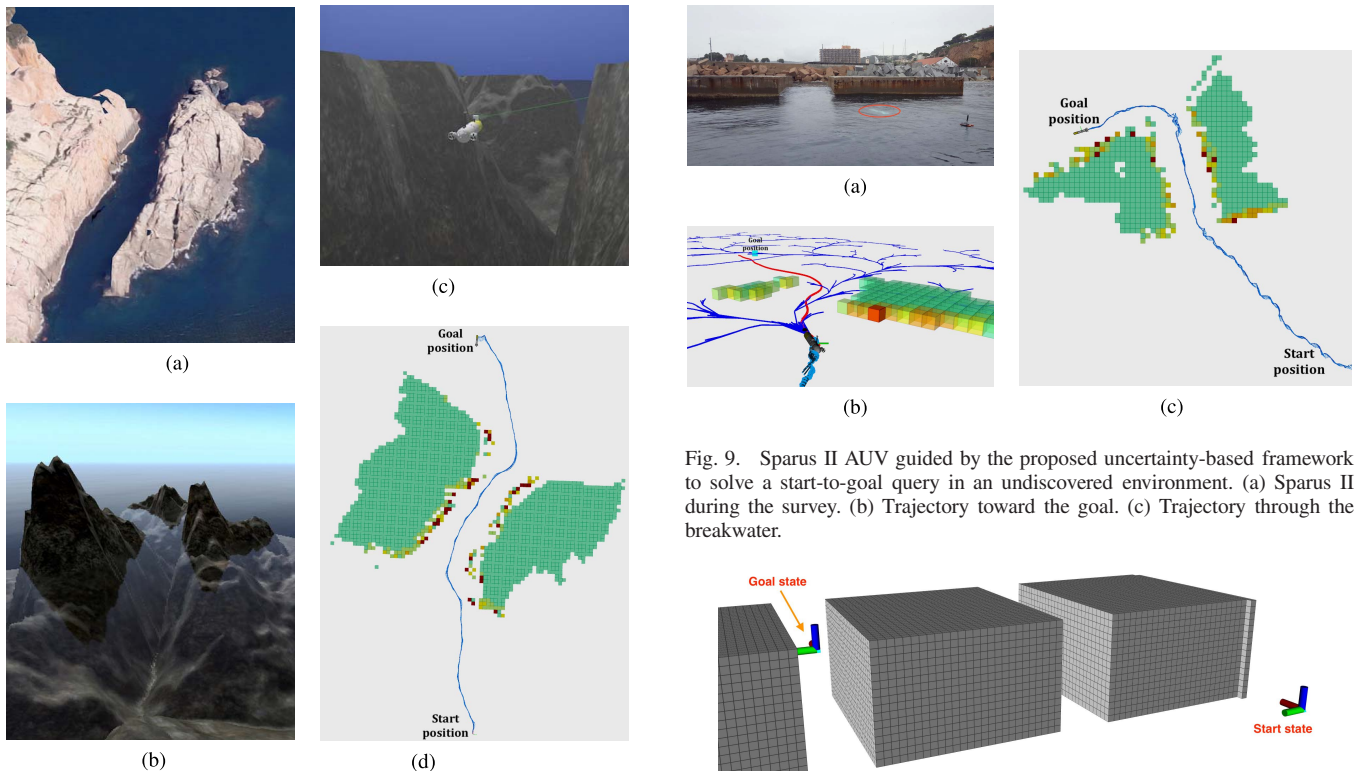


Fig. 8. Incrementally mapping and planning in the undiscovered canyon scenario. (a) Real canyon. (b) Simulated canyon. (c) Sparus II in the UWSim. (d) Trajectory through the canyon.
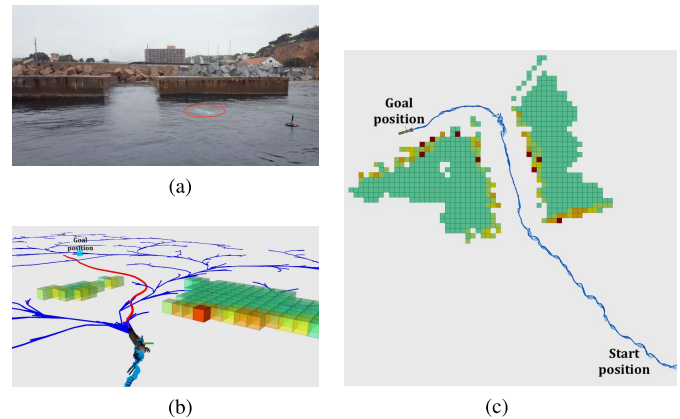


Fig. 9. Sparus II AUV guided by the proposed uncertainty-based framework to solve a start-to-goal query in an undiscovered environment. (a) Sparus II during the survey. (b) Trajectory toward the goal. (c) Trajectory through the breakwater.
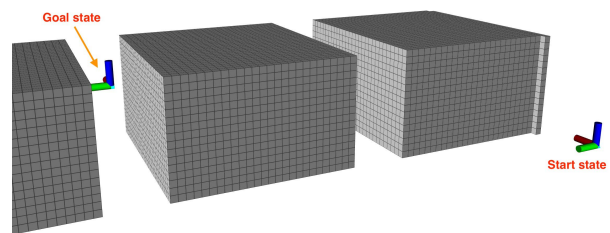


Fig. 10. Planning problem to assess the proposed multilayered scheme with adaptive $\mathcal{X}'$ in comparison to other state-of-the-art approaches. The problem is defined in the belief space for a SE(3) system operating in a 3-D workspace. The minimum safety bound is set to $p_{\text{safe}} = 0.99$.

most of the planning problems due to the search space extent. In between these two extremes, a suitable parameterization with $d = 12$ m (dashed lines) enables solving most of the trials to the planning problem while providing a trajectory with low length cost. Nevertheless, there are no efficient means of defining the optimal $d$ in advance since it is dependant on the planning problem and environment characteristics. Therefore, a rigid-$\mathcal{X}'$ strategy is not suitable for applications that lack a fully prior informative representation of the environment. Moreover, too restrictive guided searches can endanger the completeness guarantees of the planner.

The performance of those approaches that guarantee completeness, i.e., the SLP (green) and MLP with biased-$\mathcal{X}'$ (magenta) and adaptive-$\mathcal{X}'$ (orange) strategies is depicted in Fig. 12. In particular, biased-$\mathcal{X}'$ is parametrized with radius $d = 12$ m (best lead definition according to experimentation in Fig. 11) and analyzed for different $p \in [0, 1]$, whereas adaptive-$\mathcal{X}'$ is defined, as shown in Fig. 13, i.e., with an initial radius $d = 3$ m, which increases at a rate of 20 m/s. Intuitively, adaptive-$\mathcal{X}'$ adjusts $d$ from a strictly guided search to uniform sampling, i.e., as $t \to \infty$, $d \to \infty$, i.e., $\mathcal{X}' \to \mathcal{X}$.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

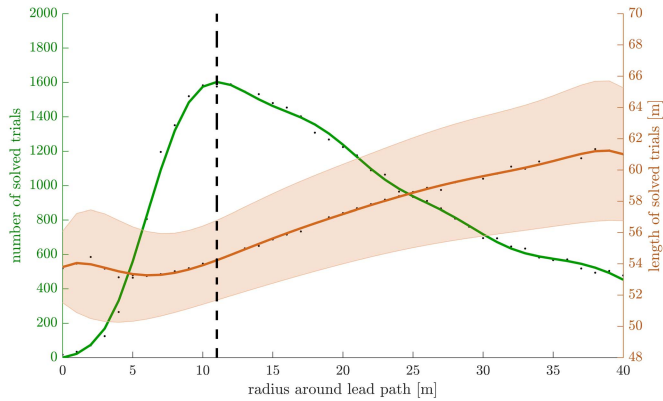PAIRET *et al.*: ONLINE MAPPING AND MOTION PLANNING UNDER UNCERTAINTY FOR SAFE NAVIGATION

15

Fig. 11. Performance of the multilayer planning scheme with a rigid-$\mathcal{X}'$, i.e., fixed radius around the geometric lead path.
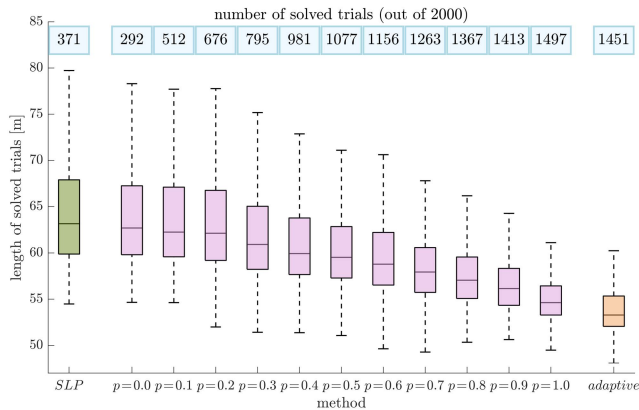


Fig. 12. Performance of 1) our precedent SLP scheme (green) and the newly proposed multilayered scheme when considering 2) a fixed lead with different bias $p$ (magenta, with best radius as found in Fig. 11), or 3) an adaptive lead, as defined in Fig. 13 (orange).
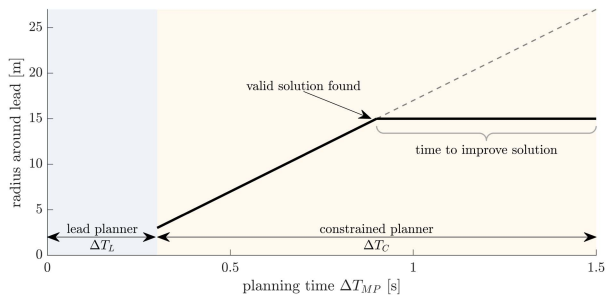


Fig. 13. Two-layered planning scheme proposed in this work. After computing a lead path, the constrained planner leverages an adaptive $\mathcal{X}'$ strategy to initially promote solutions with low cost (small $d$) before ensuring probabilistic completeness by sampling the entire space ($d \to \infty$). Once a solution is found, $\mathcal{X}'$ is fixed to let the constrained planner refine the found solution until the completion of the planning time $\Delta T_{\mathrm{MP}}$.

As it can be observed in Fig. 12, our precedent single-layered planning scheme struggles at finding a solution on most of the trials; sampling uniformly the entire high-dimensional belief space requires more time to find a

solution than the affordable time budget in online applications. Slightly worse performance is obtained when using a multilayered scheme with biased-$\mathcal{X}'$ and $p = 0$ because it still uses uniform sampling but with a portion of the total planning time budget. However, as $p \to 1$, i.e., the planner is more guided to the lead $\mathcal{X}'$ (whose optimal radius has been determined empirically in Fig. 11), the performance of the planner increases, in both the number of solved trials and the length of the final solution. Interestingly, the proposed adaptive sampling method endows the framework with a competitive success rate and solution length to that obtained when hand-defining the optimal radius.

### C. Probabilistic Collision Checking

Sampling-based planners must be able to analyze the validity of a certain state accurately and efficiently. While accuracy is relevant to avoid discarding regions of the state space, which, in fact, are collision-free, efficiency allows for more space exploration given a limited time budget. However, accurate calculations jeopardize the ability to validate a state rapidly, especially when accounting for uncertainty. In this regard, chance constraints formulations [9], [46] offer an interesting accuracy–efficiency tradeoff that has proven to be suitable for many motion planning problems in the last decade (see Section II). In fact, chance constraints formulations are still the most widely used probabilistic collision checking method among those state-of-the-art motion planning applications that account for uncertainty (e.g. [12], [71]). This motivates the use of chance constraints as the baseline reference to assess the proposed probabilistic collision checking algorithm.

The performance analysis comprises two chance constraints formulations [9], [46] and our method with four different parametrizations $\alpha = \{0.90, 0.95, 0.99, 0.999\}$. Each method is assessed by its accuracy and efficiency. Accuracy is computed as the ability to correctly detect that a state is valid

$$\frac{TP}{TP + FN} \in [0, 1] \qquad (25)$$

where a true positive (TP) indicates that a method's outcome matches the standard of truth,[5] while a false negative (FN) reflects that the method has mistakenly computed a state as invalid. TP + FN is the total number of valid states according to the standard of truth. Therefore, the higher the value of the metric in (25), the more accurate the method is. For the method's efficiency, the analysis considers the average computation time to process the state validity.[6] These two metrics are analyzed subject to three variables relevant to motion planning problems under uncertainty: 1) number of obstacles $n_o$ in the environment; 2) state uncertainty $\Sigma_{\mathbf{x}}$; and 3) minimum safety probability bound $p_{\mathrm{safe}}$. With this setup, a problem instance is parametrized by the triplet $\langle n_o, \sigma_x, p_{\mathrm{safe}} \rangle$. In total, 847 instances are retrieved according to the parametrization span and discretization defined in Table II.

Each problem instance is set up as follows. An environment $\mathcal{M}$ is defined in $\mathbb{R}^3$ with a total of $n_o$ cubical obstacles.

---

[5]The standard of truth is approximated by numerical integration of (6).

[6]All experiments are performed with an Intel Core i7-7820X CPU at 3.60 GHz $\times$ 16 with optimized C++ implementation for all methods.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

16                                                                                                          IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING
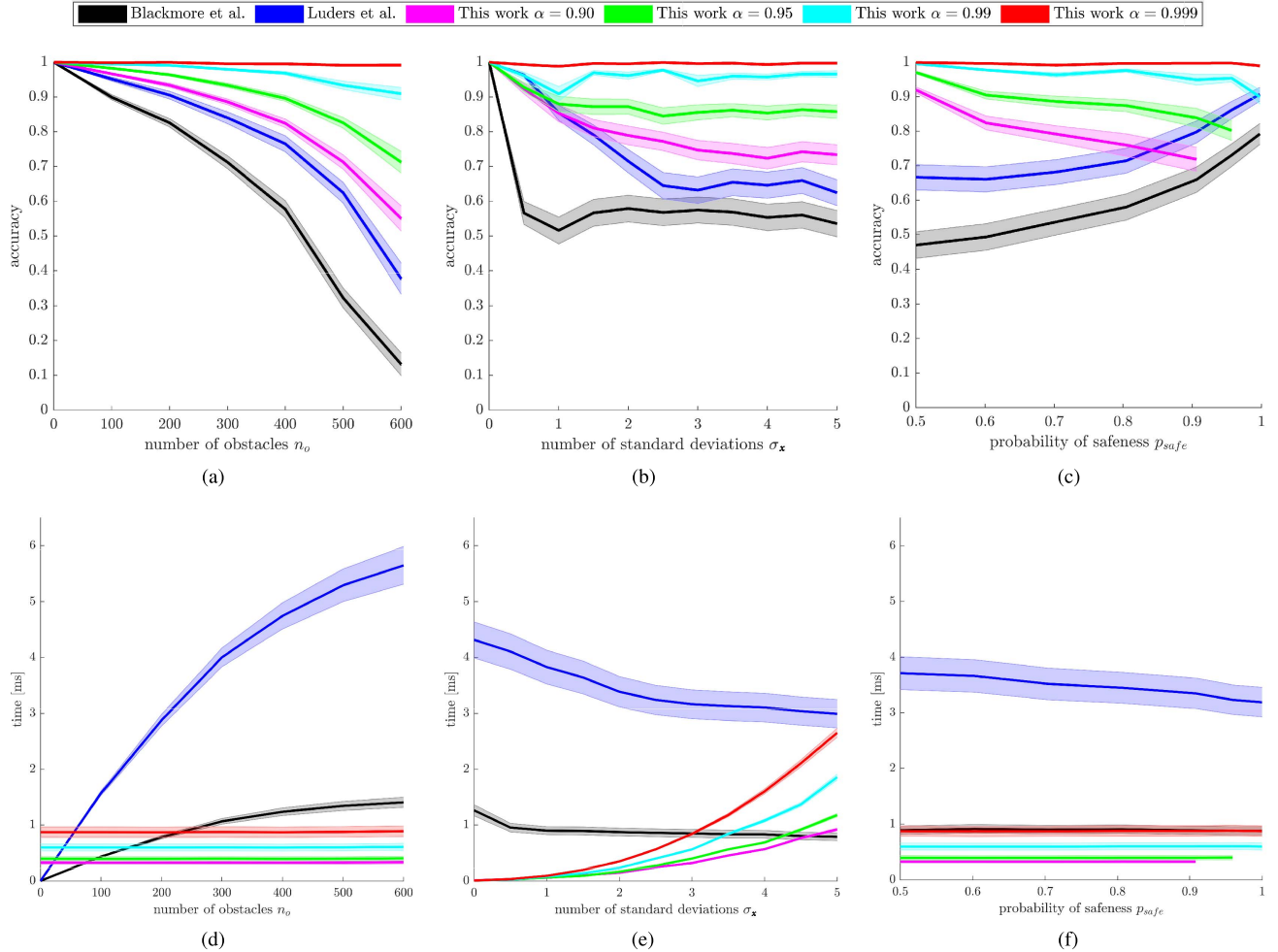


Fig. 14.  Evaluation of the accuracy (first row) and performance (second row) of the chance constraints formulations [9], [46] and the proposed probabilistic collision checking method with $\alpha = \{0.90, 0.95, 0.99, 0.999\}$. The accuracy and performance metrics are represented subject to the number of obstacles $n_o$ in the environment (first column), the state uncertainty $\Sigma_{\mathbf{x}} = \sigma_x^2 \mathbb{I}_{3 \times 3}$ (second column), and minimum safety probability bound $p_{\text{safe}}$ (third column). The shadowed area corresponds to the variance of the metrics. In the interest of clarity, only one tenth of the variance is displayed.

TABLE II

PARAMETRIZATION FOR THE COMPARISON OF PROBABILISTIC
COLLISION CHECKING METHODS

|  | Minimum value | Maximum value | Discretisation step |
|---|---|---|---|
| $n_o$ | 0 | 600 | 100 |
| $p_{\text{safe}}$ | 0.5 | 1.0 | 0.05 |
| $\sigma_x$ | 0 | 5.0 | 0.5 |

In order to have a computational representation of the scene suitable for each method, the environment is encoded as: 1) a set of linear constraints, where each cubical obstacle is characterized by six constraints and 2) a global occupancy grid map with 0.5-m resolution. Then, given the known environment $\mathcal{M}$, each probabilistic collision checking method is required to validate, subject to $p_{\text{safe}}$, 10 000 beliefs $b \sim \mathcal{N}(\hat{\mathbf{x}}, \Sigma_{\mathbf{x}})$. The state estimate $\hat{\mathbf{x}} \in \mathbb{R}^3$ is uniformly sampled over $\mathcal{X}$, and the covariance $\Sigma_{\mathbf{x}} \in \mathbb{R}^{3 \times 3}$ is set diagonal, i.e., $\Sigma_{\mathbf{x}} = \sigma_x^2 \mathbb{I}_{3 \times 3}$.

The data from the 847 problem instances are depicted in Fig. 14. In the interest of clarity, the corresponding discussion is divided into three parts: accuracy, efficiency, and suitability.

*1) Accuracy Discussion:* The accuracy analysis (first row in Fig. 14) depicts that the number of obstacles in the environment is the variable penalizing the methods' accuracy the most. This behavior is due to the methods' conservatism, whose relevance increases with the hardness of the motion planning problem. In other words, the more conservative a method is, the more negatively affected it is. On top of that, the conservatism of chance constraints formulations [9], [46] increases with the number of obstacles, whereas our approach accounts for a constant conservatism $\alpha$. This tighter bound allows our method to outperform both chance constraints formulations, even when choosing the most conservative parametrization $\alpha = 0.9$. Higher values of $\alpha$ favor accuracy at the cost of more computational expenses (see discussion below). Importantly, the confidence level $\alpha$ of our method should
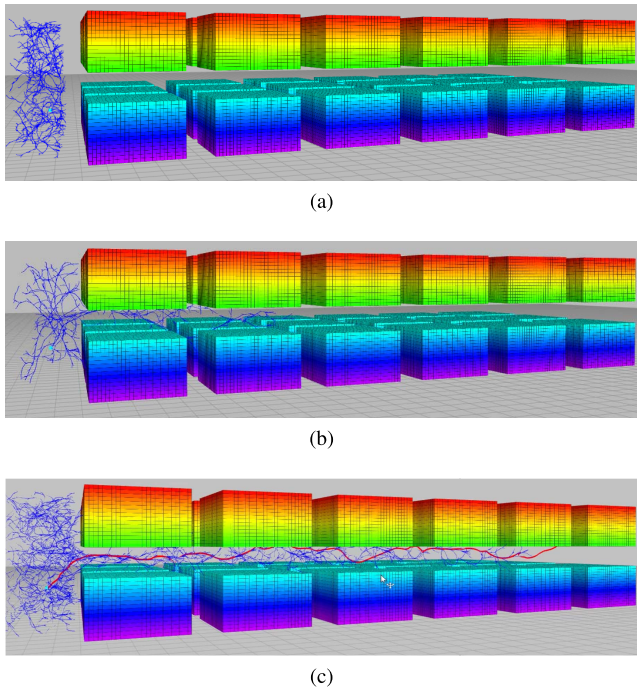
(a)

(b)

(c)

Fig. 15. Effect of conservatism in the workspace. The over conservatism of chance constraints formulations [9], [46] impedes finding a solution. Our probabilistic collision checking with a fixed conservatism $\alpha$ succeeds on finding a trajectory that transverses the environment with a total of 36 obstacles. (a) [9]. (b) [46]. (c) This work.



(a)



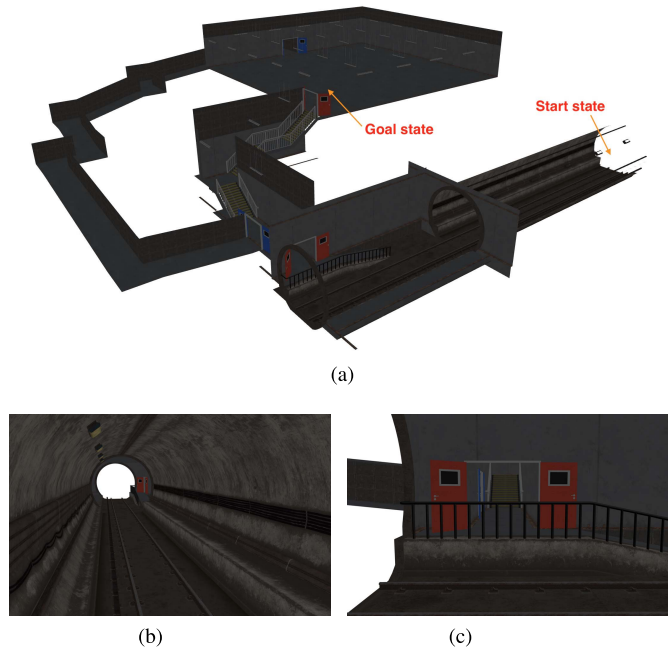(b)                                                    (c)

Fig. 16. Urban Stairwell scenario of the DARPA Subterranean Challenge 2019. (a) Start-to-goal query that requires traversing (perspective view). (b) 40-m-long tunnel and (c) narrow 25-m-long stairwell (entrance to narrow stairwell). Planning through the stairwell is particularly challenging due to the accumulated localization uncertainty.

always be set such that $\alpha \geq p_{\text{safe}}$; otherwise, the constraint in (23) will never be satisfied since the analyzed part of the space is not sufficient to ensure probabilistic safety. This fact is visible in Fig. 14(c), where, for $\alpha < p_{\text{safe}}$, our method with parametrization $\alpha$ is not used.

*2) Efficiency Discussion:* The efficiency analysis (second row in Fig. 14) reflects the expected computational complexity according to the theoretical grounds of each algorithm. That is, chance constraints strategies are fast for scenarios with few numbers of constraints, but their computational expenses grow linearly as the number of constraints increases. This linear correlation is influenced by the iterative nature of chance constraints, which allows invalidating a state as soon as $1 - p_{\text{collision}}(b, \mathcal{M}) < p_{\text{safe}}$, i.e., without the need to check all constraints. In other words, invalid states involve less time than those which are valid. Consequently, harder planning problems, i.e., those involving more obstacles, higher uncertainties, or more restrictive safety guarantees, show a mild deviation toward lower computational time due to the presence of a high number of invalid states. In contrast, the computational requirements of our method are uniquely influenced by the state uncertainty $\Sigma_z$, which determines the number of voxels to include in the calculations (see Section VI-C). This might restrict the suitability of our approach to systems whose state uncertainty is bounded over time (see discussion below).

*3) Suitability Discussion:* Robotic systems operating in uncrowded environments, i.e., very few obstacles sparsely distributed in the space, might find chance constraints to be a suitable alternative. However, the accuracy and efficiency of such approaches scale poorly as the complexity of the motion planning problem increases, i.e., more crowded

environments or higher uncertainties. As it can be observed in Fig. 15(a) and (b), this behavior endangers the ability of a planner to find a trajectory through tight apertures or narrow passages, even if one exists. If an alternative route toward the goal exists, the resulting solution will be larger than those trajectories found with less conservative approaches. Moreover, chance constraints require the representation of the environment to be a set of linear constraints, which can be prohibitively expensive to compute online, especially in applications where the environment is incrementally discovered.

In contrast, our approach trades a constant conservatism $\alpha$ in favor of accuracy and performance. This allows dealing with crowded environments efficiently while providing higher accuracy than chance constraints methods. Therefore, as depicted in Fig. 15(c), our probabilistic collision checking method enables a planner to find a solution through the tight corridors where chance constraints methods are over conservatist. However, our method involves higher computation times for highly uncertain states. This limitation might be relevant for systems with unbounded uncertainty, but most robotic systems are endowed with state estimation algorithms that keep the state uncertainty bounded over time. Alternatively, the parameter $\alpha$ can be adjusted to reduce the computation time while still guaranteeing safeness.

On the whole, the presented probabilistic collision checking approach proves to be a suitable strategy for a wide range of motion planning problems under uncertainty, even for those where chance constraints struggle at finding a solution. Moreover, our method is suitable for applications building a representation of the environment online, given that

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

18

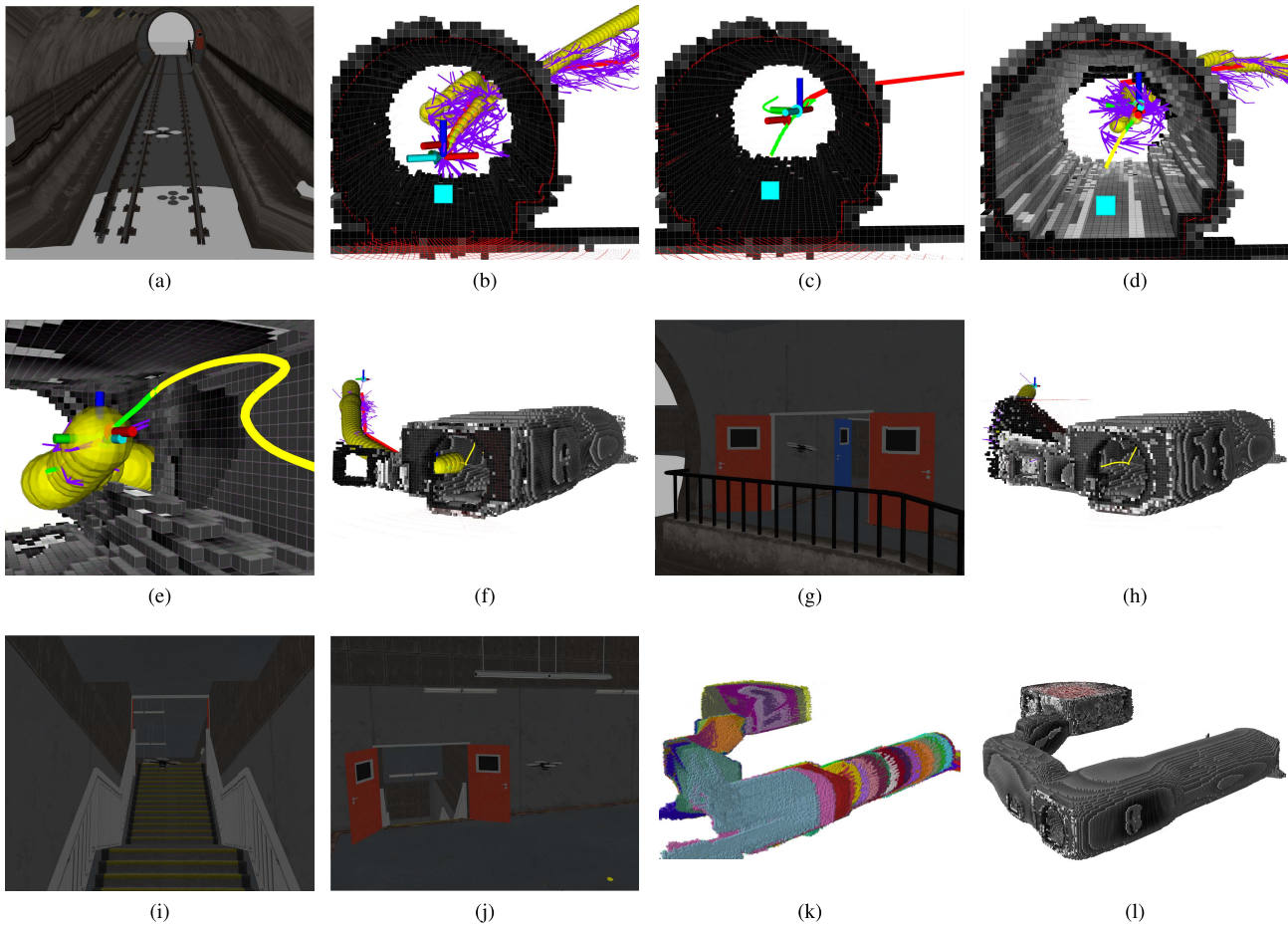IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING



Fig. 17. Online mapping and planning through the Urban Stairwell scenario of the DARPA Subterranean Challenge 2019. (a) Initial state of the quadrotor and (b) first mapping and planning iteration: geometric path (red), kinodynamic tree satisfying the probabilistic safety guarantees $p_{\text{safe}} = 0.95$ (magenta), and resulting trajectory (green) with the associated uncertainty propagation (yellow). (c) and (d) When the previous trajectory is partially invalidated due to the incremental knowledge of the surroundings, the framework finds a new trajectory toward the goal. Note that the previously observed patches of the environment become more uncertain (grayish areas) as the robot moves. (e) and (f) Entrance to the narrow stairwell is fully mapped and the framework successfully plans through it despite the considerable accumulated uncertainty. (g), (h), and (i) As the robot moves into the stairwell, the framework continues iterating over the mapping–planning process to ensure safe navigation until (j) reaching the goal region. (k) Incremental set of local maps composing the discovered environment during the mission, and (l) corresponding cumulative map $F_{\mathcal{X}}^{R'}$ (only showing those voxels $P(\mathbf{v}) > 0.4$ for visualization purposes).

those usually exploit the efficient encoding of occupancy grid maps.

### D. Navigation in Unknown 3-D Environments

The proposed framework as a whole has been deployed on a simulated quadrotor unmanned aerial vehicle (UAV) [49] equipped with a 3-D Light Detection and Ranging (LIDAR). The considered environment is the Urban Stairwell scenario of the DARPA Subterranean Challenge 2019. This scenario is challenging due to its extensive workspace of approximately $40 \times 50 \times 15$ m and all narrow passages that must be traversed to accomplish the requested start-to-goal motion planning query. Fig. 16 illustrates the Urban Stairwell scenario altogether with the defined start-to-goal query. In these experiments, the quadrotor's dynamics are approximated to those of a fixed-wing plane, as described in Appendix A-B, and the surroundings are mapped online from the sensor's data at a resolution of 0.2 m. The remaining parameters of the mapping module are as those in the experiments reported in Section VII-A. During the mission, no localization updates

are considered to test the planner in the most adversarial conditions, i.e., large environmental and localization uncertainties. The required probabilistic safety guarantees are $p_{\text{safe}} = 0.95$, and the planning time is $\Delta T_{\text{MP}} = 1.5$ s, distributed as $\Delta T_L = 0.3$ s and $\Delta T_C = 1.2$ s.

Fig. 17 depicts some snapshots[7] of the online mapping and planning procedure in the Urban Stairwell scenario of the DARPA Subterranean Challenge 2019. Noteworthy is that the mesh of the Urban Stairwell scenario is composed of a total of 108 512 faces. Although these faces could be potentially approximated online from the sensor's data and used as linear constraints in [9] and [46], it would imply checking for collision against 30 times more linear constraints than those considered in Section VII-C, for which chance constraints methods already showed poor performance due to their over conservatism. Instead, our framework efficiently deals with such complex scenarios online. All in all, the proposed framework demonstrates its suitability for probabilistically

[7]A complete trial through the Urban Stairwell scenario of the DARPA Subterranean Challenge 2019 can be seen in https://youtu.be/I5X_QFKDpeI

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

PAIRET *et al.*: ONLINE MAPPING AND MOTION PLANNING UNDER UNCERTAINTY FOR SAFE NAVIGATION
19

safe autonomous navigation in hostile and unknown environments.

## VIII. Conclusion

This article has presented a novel end-to-end framework that probabilistically guarantees the robot's safety when navigating in unexplored environments. The proposed approach is twofold: 1) incrementally maps the vehicle's surroundings to build an uncertain representation of the environment and 2) plans feasible trajectories (according to the robot's kinodynamic constraints) with probabilistic safety guarantees (according to the uncertainties in the vehicle's localization, motion, and mapping). Our proposed approach includes a multilayered planning strategy that enables for faster exploration of the high-dimensional belief space while preserving asymptotically optimal and completeness guarantees, and an efficient evaluation and tighter bound on the computation of the probability of collision than other uncertainty-aware planners in the literature. Overall, the framework is capable to deal with high-dimensional problems online while being suitable for systems with limited onboard computation power. Experimentation conducted in simulation shows some of the theoretical qualities of this work. In addition, simulated and real-world trials on an AUV and a quadrotor UAV demonstrated the suitability of the framework to guarantee the robot's safety while navigating in unexplored environments and dealing with real-robot constraints.
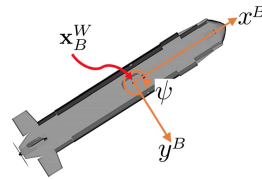
The framework is not restricted to the presented experimental evaluation or a specific platform. Any other mobile robot, either terrestrial, maritime, or aerial system, can benefit from this work. The modularity of the proposed framework allows for multiple extensions and variations. Foremost, although the experimental evaluation of the proposed framework has been conducted considering the worst case scenario of open-loop navigation without uncertainty update, the framework can bear with periodic navigation updates. An interesting possible feature that could be added to the framework is the use of the truncation trick, i.e., to uniquely propagate the posterior of the estimation, which is in no collision. However, truncating the system's belief involves approximating the posterior to a Gaussian distribution. Another possible extension is leveraging the multiresolution encoding of octomaps to check the compliance of the safety guarantee at different resolutions. Formulating this process as a multiresolution kernel checking could speed up computations even further. Finally, the conducted experimentation pointed out that automatically adjusting the replanning period might be beneficial, as well as studying more intelligent methods to leverage from the lead path or even prior solutions.

## APPENDIX A
## KINEMATIC MODELS

### A. Unicycle System

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



$$\dot{x} = v \, \cos(\psi)$$
$$\dot{y} = v \, \sin(\psi)$$
$$\dot{\psi} = \omega$$
$$\dot{v} = a$$

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

The model above approximates the AUV's behavior, but, in an underwater environment, it is subject to uncertain external forces, e.g., current. To capture this uncertainty in the dynamics, the vehicle motion is modeled as a Gaussian process. The system's motion model is first linearized by using a dynamic feedback linearization controller as presented in [15]. This technique: 1) transforms the state of the closed-loop system to $\mathbf{z} = (x, \, y, \, \dot{x}, \, \dot{y})^T$ and 2) applies a proportional derivative (PD) controller on the model to drive the system toward a desired state $\mathbf{r}$. Then, the differences between the real system and the linearized closed-loop model can be approximated by a Gaussian distribution, and the closed-loop system can be represented as a Gaussian process as in (2) and (3) with states $\mathbf{z} \in \mathcal{X} = \mathbb{R}^4$, and controls $\mathbf{r} \in \mathcal{U} = \mathcal{X}$. Thus, the system state $\mathbf{z}_k$ is best described by its probability distribution in the belief space $\mathcal{B}$, i.e., $\mathbf{b}_k = \mathcal{N}(\hat{\mathbf{z}}_k, \, \Sigma_{\mathbf{z}_k})$. The evolution of the belief is then given by the independent propagation of its mean and covariance as
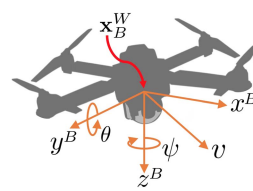
$$\hat{\mathbf{z}}_{k+1} = A\hat{\mathbf{z}}_k + B\mathbf{r}_k \qquad (26)$$
$$\Sigma_{\mathbf{z}_{k+1}} = A \Sigma_{\mathbf{z}_k} A^T + \Sigma_{\mathbf{w}} \qquad (27)$$

where $A \in \mathbb{R}^{4\times4}$ and $B \in \mathbb{R}^{4\times4}$ define the closed-loop linearized equations of motion with the PD controller as in [15], and $\Sigma_{\mathbf{w}}$ is the covariance of the noise modeling the discrepancies between (26) and the real system behavior.

### B. Fixed-Wing System

Although more complex models could be used to represent the motion capabilities of an AUV or a quadrotor unmanned aerial vehicle (UAV) operating in a 3-D workspace $\mathcal{W} = \mathbb{R}^3$ with configuration space SE(3), both vehicle's motion model can be approximated by a fixed-wing system



$$\dot{x} = v \, \cos(\psi) \, \cos(\theta)$$
$$\dot{y} = v \, \sin(\psi) \, \cos(\theta)$$
$$\dot{z} = v \, \sin(\theta)$$
$$\dot{\psi} = \omega$$
$$\dot{\theta} = q$$

where $x$, $y$, and $z$ correspond to the Cartesian coordinates of the system with respect to a predefined reference frame, $\psi$ and $\theta$, respectively, define the system's orientation around

the $z$-axis and the $y$-axis, $v$ is the vehicle's forward velocity, and $\omega$ and $q$ are the vehicle's turning rate. Thus, the system's state is defined as $\mathbf{x} = (x, \ y, \ z, \ \psi, \ \theta)^T$, and the system's control input is defined as $\mathbf{u} = (v, \ \omega, \ q)^T$.

Similar to [26], the Gaussian process describing the UAV's motion model is learned from the simulated data. The training data are extracted from the UAV's simulator producing a varied set of stationary excitations via the control input $\mathbf{u}$. Relevant control inputs are selected with the above system's model to maximize information on the output system's state $\mathbf{x}$.

Further discussion on methods for modeling robots with (partially) unknown dynamics as Gaussian processes is available in [34] and [52].

## APPENDIX B
## GAUSSIAN RELATIONSHIPS

This appendix summarizes the calculation of spatial relationships via the inverse and compound operators. These elemental transformations can be composed to calculate more complex spatial relationships. The interested reader may wish to consult [70] for a more thorough explanation about Gaussian relationships than the brief introduction that follows.

### A. Inverse Relationship

The inverse relationship $\ominus$ represents the Gaussian relationship $\mathbf{x}_i^j$ as a function of $\mathbf{x}_j^i$ as

$$\mathbf{x}_i^j := \ominus \mathbf{x}_j^i. \qquad (28)$$

The first-order estimates of the mean and the covariance of the compounding operation are

$$\hat{\mathbf{x}}_i^j \approx \ominus \hat{\mathbf{x}}_j^i \qquad (29)$$

$$\Sigma_{\mathbf{x}_i^j} \approx \mathbf{J}_\ominus \Sigma_{\mathbf{x}_j^i} \mathbf{J}_\ominus^T \qquad (30)$$

where

$$\mathbf{J}_\ominus := \frac{\partial \mathbf{x}_i^j}{\partial \mathbf{x}_j^i}. \qquad (31)$$

### B. Compound Relationship

The compounding operation $\oplus$ computes the Gaussian relationship $\mathbf{x}_k^i$ from two spatial relationships $\mathbf{x}_j^i$ and $\mathbf{x}_k^j$ that are arranged head-to-tail as

$$\mathbf{x}_k^i := \mathbf{x}_j^i \oplus \mathbf{x}_k^j. \qquad (32)$$

The first-order estimates of the mean and the covariance of the compounding operation are

$$\hat{\mathbf{x}}_k^i \approx \hat{\mathbf{x}}_j^i \oplus \hat{\mathbf{x}}_k^j \qquad (33)$$

$$\Sigma_{\mathbf{x}_k^i} \approx \mathbf{J}_\oplus \begin{bmatrix} \Sigma_{\mathbf{x}_j^i} & \Sigma_{\left(\mathbf{x}_j^i, \mathbf{x}_k^j\right)} \\ \Sigma_{\left(\mathbf{x}_k^j, \mathbf{x}_j^i\right)} & \Sigma_{\mathbf{x}_k^j} \end{bmatrix} \mathbf{J}_\oplus^T \qquad (34)$$

where $\mathbf{J}$ denotes the Jacobian, i.e., the matrix of partial derivatives

$$\mathbf{J}_\oplus := \frac{\partial \mathbf{x}_j^i \oplus \mathbf{x}_k^j}{\partial \left(\mathbf{x}_j^i, \mathbf{x}_k^j\right)} = \frac{\partial \mathbf{x}_k^i}{\partial \left(\mathbf{x}_j^i, \mathbf{x}_k^j\right)} \qquad (35)$$

$$= \begin{bmatrix} \mathbf{J}_{1\oplus} & \mathbf{J}_{2\oplus} \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{x}_k^i}{\partial \mathbf{x}_j^i} & \frac{\partial \mathbf{x}_k^i}{\partial \mathbf{x}_k^j} \end{bmatrix}. \qquad (36)$$

TABLE III

CRITICAL VALUES $t_\alpha$ COMPUTED WITH (39) FOR DIFFERENT CONFIDENCE LEVELS $\alpha$ AND GAUSSIAN DISTRIBUTION DIMENSIONALITIES

| n-D | Confidence level $\alpha$ [%] | | | | |
|---|---|---|---|---|---|
| | 85.0 | 90.0 | 95.0 | 99.0 | 99.9 |
| 1 | 1.4395 | 1.6449 | 1.9600 | 2.5758 | 3.2905 |
| 2 | 1.9479 | 2.1460 | 2.4477 | 3.0349 | 3.7169 |
| 3 | 2.3059 | 2.5003 | 2.7955 | 3.3682 | 4.0331 |

If the relationships $\mathbf{x}_j^i$ and $\mathbf{x}_k^j$ are independent, i.e., $\Sigma_{(\mathbf{x}_j^i, \mathbf{x}_k^j)} = 0$, (34) can be rewritten as

$$\Sigma_{\mathbf{x}_k^i} \approx \mathbf{J}_{1\oplus} \Sigma_{\mathbf{x}_j^i} \mathbf{J}_{1\oplus}^T + \mathbf{J}_{2\oplus} \Sigma_{\mathbf{x}_k^j} \mathbf{J}_{2\oplus}^T. \qquad (37)$$

## APPENDIX C
## $\alpha$-KERNEL CONSTRUCTION

A Gaussian distribution $\mathcal{N}(\hat{\mathbf{x}}, \Sigma_\mathbf{x})$ describing a state's belief $b$ is continuous and extends over the entire belief space. For the required computations, $b$ is represented on a discrete support $\mathcal{K}_\alpha(\Sigma_\mathbf{x})$, referred to as $\alpha$-kernel, with resolution $h$ and size $m_d$ at each dimension $d$ as defined by

$$m_d = 2 \operatorname{ceil}\left(\frac{t_\alpha \sigma_d}{h}\right) + 1 \qquad (38)$$

where the kernel size $m_d$ is always odd, $\sigma_d$ is the standard deviation of $\Sigma_\mathbf{x}$ along dimension $d$, and the critical value $t_\alpha$ is computed from the desired confidence level $\alpha \in [0, 1]$ as

$$t_\alpha = -\phi^{-1}\left(\frac{1}{2}(1 - \alpha)\right) \qquad (39)$$

where $\phi^{-1}(\cdot)$ denotes the quantile function of the d-dimensional Gaussian normal distribution describing the system's belief $b$. Table III shows some critical values $t_\alpha$ according to commonly desirable confidence levels $\alpha$ for 1-D, 2-D, and 3-D Gaussian distributions.

Noteworthy is that the confidence level $\alpha$ involves a tradeoff between computational performance and accuracy. On one hand, $\alpha$ bounds the extend of the resulting $\mathcal{K}_\alpha(\cdot)$ over the belief space, thus determining the total number of voxels in the kernel and, consequently, having an impact on the computational load of the probabilistic collision checking formulated in (23). On the other hand, (23) introduces a constant conservatism $\alpha$, implying that $\alpha$ must be selected such that $\alpha > p_{\text{safe}}$. Otherwise, the method will not find any valid state. This requirement is implicit in the probabilistic collision checking formulated in (23).

The value of each cell $n \in \mathcal{K}_\alpha(\Sigma_\mathbf{x})$ can be drawn from the corresponding Gaussian distribution as $h^D \mathcal{N}(\mathbf{x} \mid \hat{\mathbf{x}}, \Sigma_\mathbf{x})$, where $h^D$ is a normalizing constant according to the kernel resolution $h$ and $\mathbf{x}$ is the point coordinate of $n$ referenced at $\hat{\mathbf{x}}$. For the particular case of a multivariate Gaussian $\mathcal{N}(\hat{\mathbf{x}}, \Sigma_\mathbf{x})$ with diagonal covariance matrix $\Sigma_\mathbf{x}$, its elements can be written as $\Sigma_{ij} = \sigma_i^2 \mathbb{I}_{ij}$, where $\mathbb{I}_{ij}$ are the matrix elements of the identity matrix (so $\mathbb{I}_{ij} = 0$ if $i \neq j$ and $\mathbb{I}_{ij} = 1$).

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

PAIRET *et al.*: ONLINE MAPPING AND MOTION PLANNING UNDER UNCERTAINTY FOR SAFE NAVIGATION 21

Then, the multivariate Gaussian with diagonal $\Sigma_{ij} = \sigma_i^2 \mathbb{I}_{ij}$ factorizes into a product of univariate Gaussians as

$$\mathcal{N}(\mathbf{x} \mid \hat{\mathbf{x}}, \ \Sigma_{\mathbf{x}}) = h^D \prod_{i=1}^{D} \mathcal{N}\big(\mathbf{x}_i \mid \hat{\mathbf{x}}, \ \sigma_{\mathbf{x}_i}^2\big) \tag{40}$$

where, for any arbitrary positive definite covariance matrix $\Sigma_{\mathbf{x}}$, the resulting distribution is normalized. The property in (40) provides a computationally efficient strategy to build any d-dimensional kernel $\mathcal{K}(\cdot)$ from 1-D Gaussian signals.

It is worth mentioning that the kernel computation can be conducted and stored offline for different kernel sizes. At the running time, the planner would uniquely need to retrieve in a lookup table fashion the required kernel. Although this is an option to speed up the performance of the presented probabilistic collision checking, the implementation in this work computes the kernels online.

## Acknowledgment

## References

[1] A.-A. Agha-Mohammadi, S. Chakravorty, and N. M. Amato, "FIRM: Sampling-based feedback motion-planning under motion uncertainty and imperfect measurements," *Int. J. Robot. Res.*, vol. 33, no. 2, pp. 268–304, 2014.

[2] A.-A. Agha-Mohammadi, S. Agarwal, S.-K. Kim, S. Chakravorty, and N. M. Amato, "SLAP: Simultaneous localization and planning under uncertainty via dynamic replanning in belief space," *IEEE Trans. Robot.*, vol. 34, no. 5, pp. 1195–1214, Oct. 2018.

[3] A.-A. Agha-Mohammadi, E. Heiden, K. Hausman, and G. Sukhatme, "Confidence-rich grid mapping," *Int. J. Robot. Res.*, vol. 38, nos. 12–13, pp. 1352–1374, Oct. 2019.

[4] A. Agha *et al.*, "NeBula: Quest for robotic autonomy in challenging environments; TEAM CoSTAR at the DARPA subterranean challenge," 2021, *arXiv:2103.11470*. [Online]. Available: http://arxiv.org/abs/2103.11470

[5] R. Alterovitz, T. Simeon, and K. Goldberg, "The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty," in *Proc. Robot., Sci. Syst. III*, Jun. 2007, pp. 233–241.

[6] F. Andert, F. Adolf, L. Goormann, and J. Dittrich, "Mapping and path planning in complex environments: An obstacle avoidance approach for an unmanned helicopter," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 745–750.

[7] H. Balta *et al.*, "Integrated data management for a fleet of search-and-rescue robots," *J. Field Robot.*, vol. 34, no. 3, pp. 539–582, May 2017.

[8] T. Beckers, J. Umlauft, and S. Hirche, "Stable model-based control with Gaussian process regression for robot manipulators," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3877–3884, Jul. 2017.

[9] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Trans. Robot.*, vol. 27, no. 6, pp. 1080–1094, Dec. 2011.

[10] M. Carreras *et al.*, "Testing Sparus II AUV, an open platform for industrial, scientific and academic applications," *Instrum. Viewpoint*, vol. 18, pp. 54–55, Feb. 2021.

[11] M. Chen *et al.*, "FaSTrack: A modular framework for real-time motion planning and guaranteed safe tracking," *IEEE Trans. Autom. Control*, early access, Feb. 16, 2021, doi: 10.1109/TAC.2021.3059838.

[12] M. D. S. Arantes, C. F. M. Toledo, B. C. Williams, and M. Ono, "Collision-free encoding for chance-constrained nonconvex path planning," *IEEE Trans. Robot.*, vol. 35, no. 2, pp. 433–448, Apr. 2019.

[13] N. Dadkhah and B. Mettler, "Survey of motion planning literature in the presence of uncertainty: Considerations for UAV guidance," *J. Intell. Robot. Syst.*, vol. 65, no. 1, pp. 233–246, 2012.

[14] R. De Iaco, S. L. Smith, and K. Czarnecki, "Learning a lattice planner control set for autonomous vehicles," 2019, *arXiv:1903.02044*. [Online]. Available: http://arxiv.org/abs/1903.02044

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

[16] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *Amer. J. Math.*, vol. 79, no. 3, pp. 497–516, 1957.

[17] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 99–110, Jun. 2006.

[18] D. D. Fan, K. Otsu, Y. Kubo, A. Dixit, J. Burdick, and A.-A. Agha-Mohammadi, "STEP: Stochastic traversability evaluation and planning for safe off-road navigation," in *Robotics: Science and Systems*. RSS Found., 2021, pp. 1–21.

[19] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3019–3026, Oct. 2018.

[20] E. Frazzoli, M. A. Dahleh, and E. Feron, "Maneuver-based motion planning for nonlinear systems with symmetries," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1077–1091, Dec. 2005.

[21] D. Fridovich-Keil, S. L. Herbert, J. F. Fisac, S. Deglurkar, and C. J. Tomlin, "Planning, fast and slow: A framework for adaptive real-time safe trajectory planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 387–394.

[22] D. Fridovich-Keil, J. F. Fisac, and C. J. Tomlin, "Safely probabilistically complete real-time planning and exploration in unknown environments," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 7470–7476.

[23] E. Galceran, R. Campos, N. Palomeras, D. Ribas, M. Carreras, and P. Ridao, "Coverage path planning with real-time replanning and surface reconstruction for inspection of three-dimensional underwater structures using autonomous underwater vehicles," *J. Field Robot.*, vol. 32, no. 7, pp. 952–983, Oct. 2015.

[24] S. Ghosh, K. Otsu, and M. Ono, "Probabilistic kinematic state estimation for motion planning of planetary rovers," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 5148–5154.

[25] K. Hauser and Y. Zhou, "Asymptotically optimal planning by feasible kinodynamic planning in a state–cost space," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1431–1443, Dec. 2016.

[26] P. Hemakumara and S. Sukkarieh, "Learning UAV stability and control derivatives using Gaussian processes," *IEEE Trans. Robot.*, vol. 29, no. 4, pp. 813–824, Aug. 2013.

[27] J. D. Hernandez, M. Moll, E. Vidal, M. Carreras, and L. E. Kavraki, "Planning feasible and safe paths online for autonomous underwater vehicles in unknown environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 1313–1320.

[28] J. D. Hernández, E. Vidal, M. Moll, N. Palomeras, M. Carreras, and L. E. Kavraki, "Online motion planning for unexplored underwater environments using autonomous underwater vehicles," *J. Field Robot.*, vol. 36, no. 2, pp. 370–396, 2019.

[29] B.-J. Ho, P. Sodhi, P. Teixeira, M. Hsiao, T. Kusnur, and M. Kaess, "Virtual occupancy grid map for submap-based pose graph SLAM and planning in 3D environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 2175–2182.

[30] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auto. Robots*, vol. 34, no. 3, pp. 189–206, Apr. 2013.

[31] F. S. Hover *et al.*, "Advanced perception, navigation and planning for autonomous in-water ship hull inspection," *Int. J. Robot. Res.*, vol. 31, no. 12, pp. 1445–1464, Oct. 2012.

[32] D. Hsu, J.-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," in *Proc. Int. Conf. Robot. Automat.*, vol. 3, Apr. 1997, pp. 2719–2726.

[33] V. A. Huynh, S. Karaman, and E. Frazzoli, "An incremental sampling-based algorithm for stochastic optimal control," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 2865–2872.

[34] J. Jackson, L. Laurenti, E. Frew, and M. Lahijanian, "Safety verification of unknown dynamical systems via Gaussian process regression," 2020, *arXiv:2004.01821*. [Online]. Available: http://arxiv.org/abs/2004.01821

[35] L. Janson, E. Schmerling, and M. Pavone, "Monte Carlo motion planning for robot trajectory optimization under uncertainty," in *Robotics Research*. Cham, Switzerland: Springer, 2018, pp. 343–361.

[36] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[37] S.-K. Kim *et al.*, "PLGRIM: Hierarchical value learning for large-scale exploration in unknown environments," in *Proc. Int. Conf. Automated Planning Scheduling*, vol. 31, May 2021, pp. 652–662.

[38] S. M. LaValle and J. J. Kuffner, Jr., "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, 2001.

[39] S. M. LaValle and R. Sharma, "A framework for motion planning in stochastic environments: Modeling and analysis," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1995, pp. 3057–3062.

[40] J. Le Ny and G. J. Pappas, "On trajectory optimization for active sensing in Gaussian process models," in *Proc. 48h IEEE Conf. Decis. Control (CDC) Held Jointly With 28th Chin. Control Conf.*, Dec. 2009, pp. 6286–6292.

[41] Y. Li, Z. Littlefield, and K. E. Bekris, "Asymptotically optimal sampling-based kinodynamic planning," *Int. J. Robot. Res.*, vol. 35, no. 5, pp. 528–564, Feb. 2016.

[42] Y. Lin and S. Saripalli, "Path planning using 3D Dubins curve for unmanned aerial vehicles," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, May 2014, pp. 296–304.

[43] W. Liu and M. H. Ang, "Incremental sampling-based algorithm for risk-aware planning under motion uncertainty," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 2051–2058.

[44] I. Lluvia, E. Lazkano, and A. Ansuategi, "Active mapping and robot exploration: A survey," *Sensors*, vol. 21, no. 7, p. 2445, Apr. 2021.

[45] B. Luders, M. Kothari, and J. How, "Chance constrained RRT for probabilistic robustness to environmental uncertainty," in *Proc. AIAA Guid., Navigat., Control Conf.*, 2010, p. 8160.

[46] B. D. Luders, S. Karaman, and J. P. How, "Robust sampling-based motion planning with asymptotic optimality guarantees," in *Proc. AIAA Guid., Navigat., Control (GNC) Conf.*, Aug. 2013, p. 5097.

[47] R. Luna, M. Lahijanian, M. Moll, and L. E. Kavraki, "Optimal and efficient stochastic motion planning in partially-known environments," in *Proc. 28th AAAI Conf. Artif. Intell.*, Jul. 2014, pp. 2549–2555.

[48] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *Int. J. Robot. Res.*, vol. 36, no. 8, pp. 947–982, Jun. 25, 2017.

[49] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, and O. von Stryk, "Comprehensive simulation of quadrotor UAVs using ROS and gazebo," in *Proc. 3rd Int. Conf. Simulation, Modeling Program. Auto. Robots (SIMPAR)*, 2012, pp. 400–411.

[50] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2, Mar. 1985, pp. 116–121.

[51] G. Morgenthal and N. Hallermann, "Quality assessment of unmanned aerial vehicle (UAV) based visual inspection of structures," *Adv. Struct. Eng.*, vol. 17, no. 3, pp. 289–302, Mar. 2014.

[52] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Model learning with local Gaussian process regression," *Adv. Robot.*, vol. 23, no. 15, pp. 2015–2034, 2009.

[53] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean signed distance fields for on-board MAV planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 1366–1373.

[54] K. Otsu, G. Matheron, S. Ghosh, O. Toupet, and M. Ono, "Fast approximate clearance evaluation for rovers with articulated suspension systems," *J. Field Robot.*, vol. 37, no. 5, pp. 768–785, Aug. 2020.

[55] E. Pairet, J. D. Hernandez, M. Lahijanian, and M. Carreras, "Uncertainty-based online mapping and motion planning for marine robotics guidance," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 2367–2374.

[56] E. Pairet, C. Chamzas, Y. R. Petillot, and L. Kavraki, "Path planning for manipulation using experience-driven random trees," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3295–3302, Apr. 2021.

[57] L. Palmieri, S. Koenig, and K. O. Arras, "RRT-based nonholonomic motion planning using any-angle path biasing," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 2775–2781.

[58] C. Park, J. Pan, and D. Manocha, "ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments," in *Proc. 22nd Int. Conf. Automated Planning Scheduling*, 2012, pp. 207–215.

[59] S. Patil, J. van den Berg, and R. Alterovitz, "Estimating probability of collision for safe motion planning under Gaussian motion and sensing uncertainty," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 3238–3244.

[60] P. Pinies and J. D. Tardos, "Scalable SLAM building conditionally independent local maps," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2007, pp. 3466–3471.

[61] P. Pinies and J. D. Tardos, "Large-scale SLAM building conditionally independent local maps: Application to monocular vision," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 1094–1106, 2008.

[62] M. Pivtoraiko and A. Kelly, "Kinodynamic motion planning with state lattice motion primitives," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 2172–2179.

[63] E. Plaku, "Region-guided and sampling-based tree search for motion planning with dynamics," *IEEE Trans. Robot.*, vol. 31, no. 3, pp. 723–735, Jun. 2015.

[64] E. Plaku, L. E. Kavraki, and M. Y. Vardi, "Motion planning with dynamics by a synergistic combination of layers of planning," *IEEE Trans. Robot.*, vol. 26, no. 3, pp. 469–482, 2010.

[65] M. Prats, J. Perez, J. J. Fernandez, and P. J. Sanz, "An open source tool for simulation and supervision of underwater intervention missions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 2577–2582.

[66] F. Ramos and L. Ott, "Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent," *Int. J. Robot. Res.*, vol. 35, no. 14, pp. 1717–1730, Jan. 2016.

[67] J. A. Reeds and L. A. Shepp, "Optimal paths for a car that goes both forwards and backwards," *Pacific J. Math.*, vol. 145, no. 2, pp. 367–393, Oct. 1990.

[68] J. S. Willners, D. Gonzalez-Adell, J. D. Hernández, È. Pairet, and Y. Petillot, "Online 3-dimensional path planning with kinematic constraints in unknown environments using hybrid A* with tree pruning," *Sensors*, vol. 21, no. 4, p. 1152, Feb. 2021.

[69] S. Scherer, S. Singh, L. Chamberlain, and M. Elgersma, "Flying fast and low among obstacles: Methodology and experiments," *Int. J. Robot. Res.*, vol. 27, no. 5, pp. 549–574, May 2008.

[70] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous Robot Vehicles*. New York, NY, USA: Springer, 1990, pp. 167–193.

[71] D. Strawser and B. Williams, "Approximate branch and bound for fast, risk-bound stochastic path planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 7047–7054.

[72] I. A. Şucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robot. Autom. Mag.*, vol. 19, no. 4, pp. 72–82, Dec. 2012.

[73] W. Sun, S. Patil, and R. Alterovitz, "High-frequency replanning under uncertainty using parallel sampling-based motion planning," *IEEE Trans. Robot.*, vol. 31, no. 1, pp. 104–116, Feb. 2015.

[74] S. Suresh, P. Sodhi, J. G. Mangelson, D. Wettergreen, and M. Kaess, "Active SLAM using 3D submap saliency for underwater volumetric exploration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 3132–3138.

[75] J. van den Berg, P. Abbeel, and K. Goldberg, "LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 895–913, 2011.

[76] E. Vidal, M. Moll, N. Palomeras, J. D. Hernandez, M. Carreras, and L. E. Kavraki, "Online multilayered motion planning with dynamic constraints for autonomous underwater vehicles," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 8936–8942.

[77] J. Wang and B. Englot, "Fast, accurate Gaussian process occupancy maps via test-data octrees and nested Bayesian fusion," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 1003–1010.

[78] D. J. Webb and J. van den Berg, "Kinodynamic RRT*: Optimal motion planning for systems with linear differential constraints," 2012, *arXiv:1205.5088*. [Online]. Available: https://arxiv.org/abs/1205.5088

[79] M. Yguel, O. Aycard, and C. Laugier, "Update policy of dense maps: Efficient algorithms and sparse representation," in *Field and Service Robotics*. Berlin, Germany: Springer, 2008, pp. 23–33.

[80] D. Youakim, P. Cieslak, A. Dornbush, A. Palomer, P. Ridao, and M. Likhachev, "Multirepresentation, multiheuristic A* search-based motion planning for a free-floating underwater vehicle-manipulator system in unknown environment," *J. Field Robot.*, vol. 37, no. 6, pp. 925–950, Sep. 2020.

**Èric Pairet** (Associate Member, IEEE) received the B.Sc. degree in electronic engineering from the University of Girona, Girona, Spain, in 2015, the Erasmus Mundus M.Sc. degree in computer vision and robotics from the University of Burgundy, Dijon, France, University of Girona, and Heriot-Watt University, Edinburgh, U.K., in 2017, and the M.Sc. degree in robotics and autonomous systems from The University of Edinburgh, Edinburgh, and Heriot-Watt University, Edinburgh Campus, Edinburgh, in 2018, where he is currently pursuing the Ph.D. degree in robotics.

He is currently a Senior Researcher with the Technology Innovation Institute, Abu Dhabi, UAE. His research is focused on techniques capable of extracting and leveraging motion plan abstractions to efficiently define robotics behavior, particularly for object manipulation and robot navigation applications.

**Marc Carreras** (Member, IEEE) received the B.Sc. degree in industrial engineering and the Ph.D. degree in computer engineering from the University of Girona, Girona, Spain, in 1998 and 2003, respectively.

From 1999 to 2019, he has participated in 24 research projects (European and national projects). He is currently an Associate Professor with the Computer Vision and Robotics Institute, University of Girona. He is an author of about 150 publications. He has supervised five Ph.D. dissertations and participated in several European autonomous underwater vehicle (AUV) competitions (five times winner). His research activity mainly focuses on underwater robotics in research topics, such as intelligent control architectures, robot learning, path planning, and AUV design, modeling, and identification.

**Yvan Petillot** (Member, IEEE) received the Telecommunications degree with a specialization in image and signal processing, the M.Sc. degree in optics and signal processing, and the Ph.D. degree in real-time pattern recognition using optical processors from the Université de Bretagne Occidentale, Ecole Nationale Supérieure des Télécommunications de Bretagne (ENSTBr), Brest, France, in 1991, 1992, and 1996, respectively.

He is currently a Professor of robotics and autonomous systems with Heriot-Watt University, Edinburgh, U.K. He is interested in developing robotics solutions in the marine sector, supporting his long-term vision of robot teams operating in hazardous environments, supported by remote human operators. Over the last 20 years, he has made major contributions to robot perception, navigation, and planning in the underwater domain.

**Juan David Hernández** (Senior Member, IEEE) received the B.Sc. degree in electronic engineering from Pontifical Xavierian University, Bogotá, Colombia, in 2009, the M.Sc. degree in robotics and automation from the Technical University of Madrid, Madrid, Spain, in 2012, and the Ph.D. degree in technology (robotics) from the University of Girona, Girona, Spain, in 2017.

He worked as a Robotics Research Engineer with the Netherlands Organization for Applied Scientific Research (TNO), The Hague, The Netherlands, from 2017 to 2018. He was a Post-Doctoral Research Associate with Rice University, Houston, TX, USA, from 2018 to 2019. He was a Senior Engineer for simulation of autonomous systems with Apple Inc., Sunnyvale, CA, USA, from 2019 to 2020. He is currently a Lecturer (Assistant Professor) with Cardiff University, Cardiff, U.K., where he is part of the Center for AI, Robotics and Human-Machine Systems (IROHMS). His research is focused on motion planning algorithms and human–robot collaboration.

Dr. Hernández is a Senior Member of the IEEE Robotics and Automation Society.

**Morteza Lahijanian** (Member, IEEE) received the B.S. degree in bioengineering from the University of California at Berkeley, Berkeley, CA, USA, in 2005, and the M.S. and Ph.D. degrees in mechanical engineering from Boston University, Boston, MA, USA, in 2009 and 2013, respectively.

He was a Research Scientist with the Department of Computer Science, University of Oxford, Oxford, U.K. He is currently an Assistant Professor with the Department of Aerospace Engineering Sciences and Department of Computer Science (by courtesy) with the University of Colorado Boulder, Boulder, CO, USA, and also the Director of Assured, Robust, and Interactive Autonomous (ARIA) Systems Group. He conducted post-doctoral research at the Department of Computer Science, Rice University.