Online Information Gathering using Sampling-Based Planners and GPs: an Information Theoretic Approach

Alberto Viseras[1], Dmitriy Shutin[1] and Luis Merino[2]

Abstract— Information gathering algorithms aim to intelligently select the robot actions required to efficiently obtain an accurate reconstruction of a physical process, such as an occupancy map, or a magnetic field. Many recent works have proposed algorithms for information gathering. However, these algorithms employ discretization of the state space, which makes them computationally intractable for robotic systems with complex dynamics. Moreover, most algorithms are not suited for online information gathering tasks. This paper presents a novel approach that tackles the two aforementioned issues. Specifically, our approach includes two intertwined steps: a Gaussian processes (GPs)-based prediction that allows a robot to identify highly unexplored locations, and an RRT*-based informative path planning that guides the robot towards those locations. The combination of the two steps allows an online realization of the algorithm, while eliminates the need of discretization. We demonstrate the effectiveness of the proposed algorithm in simulations, as well as with an experiment in which a ground-based robot explores the magnetic field intensity within an indoor environment populated with obstacles.

Fig. 1: Ground-based robot exploring a magnetic field intensity (projected on the ground) within an indoor environment populated with obstacles.

I. INTRODUCTION

Information gathering is a fundamental task in a wide range of robotic applications [1], [2]. The objective is to gather information efficiently, saving resources like time, battery, etc. This may be economically advantageous or even life-critical in search and rescue missions.

In this paper, we devise information gathering strategies that allow a robot to collect information, while optimizing the resources employed. Here we follow an approach that is widely used in the literature: first, an underlying model of the physical process under study together with some metric of information quality is employed to predict the impact of certain robot actions and states. Second, an active non-myopic sensing strategy is used to maximize the mentioned metric [3], [4].

A number of information quality metrics has been proposed for information gathering, like entropy [5], Fisher information [6] or mutual information [7]. The calculation of the information quality metric requires an underlying model for representing the observed process. In this work, we propose the use of Gaussian processes (GPs) [8] for such purpose, which represent a powerful method to model spatial phenomena.

The use of the mutual information for active sensing with GPs has been extensively studied by Krause *et. al.* [9]. However, they consider two assumptions that make the algorithm not suitable for online information gathering problems. First, the authors assume that the hyperparameters that define the GPs covariance function are *a priori* known, which does not typically hold. Additionally, they assume discrete sensor placements and do not consider the robot's motion and constraints. Furthermore, the computation of mutual information is expensive for online information gathering.

Several works that relax the first assumption (i.e., the known hyperparameters) have been also proposed in the literature [10], [2]. [10] derives a non-myopic algorithm that considers the exploration-exploitation trade-off with respect to the GPs hyperparameters. However, [10] considers neither the robot's dynamics constraints, nor the presence of obstacles. In [2], the authors propose a decentralized greedy multi-agent algorithm, where each of the robots gathers information about the process of interest within an obstacle-free environment, and learns the optimal hyperparameters given the current measurements. In contrast to these works we derive an algorithm that handles a complex environment, i.e. an environment that is populated with obstacles and does not require a spatial gridding of explored space.

The central question that we address in this paper is how to efficiently gather information of an unknown physical process, which takes place in a complex environment, with a robot that runs an algorithm online as it collects information. We solve this problem by proposing a two-step strategy. First,

[1]A. Viseras and D. Shutin are with the Institute of Communications and Navigation of the German Aerospace Center (DLR), Oberpfaffenhofen, 82234, Wessling, Germany {alberto.viserasruiz,dmitriy.shutin}@dlr.de

[2]L. Merino is with the Universidad Pablo de Olavide, Crta. Utrera km 1, 41013, Seville, Spain, lmercab@upo.es. The work of L.M. was partially supported by MINECO (Spain) grant OCELLIMAV (TEC-61708-EXP).
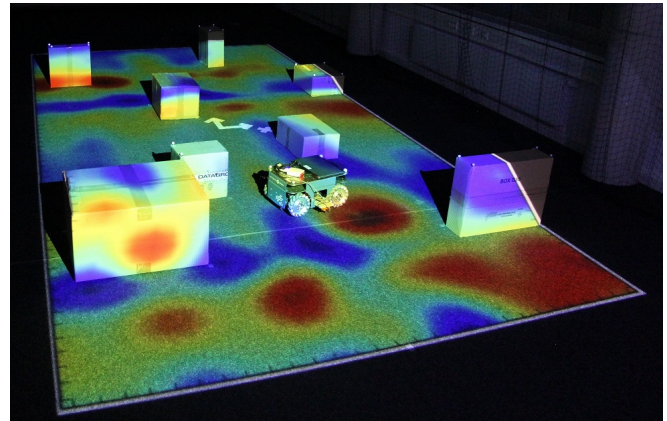
123

the algorithm finds a highly informative location according to a pre-specified information quality metric. This location we term a <u>station</u> – a concept inspired by frontiers in autonomous robotic exploration [11]. Once the station is found, the robot plans a path towards the station. When planning, we trade-off path cost with the information gained while traversing the path to the station. This we term informative path planning. Once the robot reaches the station it will update its GP model, and the robot will look for the next station and then planning a new a path leveraging the updated model.

A. Informative Path Planning

Informative path planning encompasses algorithms that aim to plan a path which is both feasible given the robot's differential constraints and optimal with respect to some information quality metric. Our focus lies specifically in informative path planning for information gathering tasks [12], [13].

In [12] the authors tackle the optimal information gathering problem with a multi-robot system. However, they assume a discretized grid environment that is obstacle-free. The assumption of a grid environment presents two major drawbacks. First, the introduction of the cell's size adds an additional parameter to the algorithm. Second, recent works have proposed information gathering algorithms for robots with large state spaces that include kinematic and/or kinodynamic constraints, like e.g. UAVs [14]. Graph-based algorithms with such motion models become increasingly complex [15]. In this work, we propose an approach that is able to incorporate arbitrary information quality metrics, environments populated with obstacles, and account for arbitrary robot dynamics. Specifically, we propose the adaptation of the asymptotically optimal rapidly exploring random trees (RRT*) [16] for information gathering tasks.

The incorporation of information quality metrics in the RRTs has been already investigated in the literature. The information-rich RRT (iRRT) extends the RRT algorithm by incorporating a Fisher information measure [6]. However, [6] is limited to tracking applications. Also, RRTs have been used for exploration tasks with an UAV [17]. There the authors generate several alternative trajectories with the RRT algorithm and select the one that results in the highest mutual information between the current estimate and the corresponding prediction conditioned on a selected route. We will use this algorithm to benchmark the performance of the proposed informative path planner. In [18], the use of rapidly exploring random cycles (RRC) for persistent monitoring of a spatio-temporal Gaussian random field is proposed. Our focus lies, however, on efficient information gathering of a static physical process; i.e. we aim to gather the maximum information in the minimum amount of time.

The work by [13] is the one that is closest to our information gathering strategy. There, the authors propose the rapidly exploring information gathering (RIG) planner – a sampling based algorithm that is able to solve the information gathering problem under a pre-specified budget constraint.

They assume that the underlying model that describes the process is *a priori* known, and the robot does not need to reach a particular goal position. Our work addresses a similar problem but it differs in two principal aspects: (i) our algorithm does not require prior information of the physical process, which, in contrast to [13], allows an online realization of the algorithm; (ii) our algorithm introduces a trade-off between information gathering and a cost of a particular selected path. The consequence of last aspect leads to a path objective function that incorporates both an information quality metric as well as a cost term. A cost function that trades-off these two terms was also proposed by [19]. They formulate a cost function that trades-off the mutual information and the path cost. In this work, we define a function that combines both the information and the path cost and incorporate it as a utility in our modified RRT* algorithm. In conjunction with intelligent station selection, the informative path planing represents the key contribution of this paper.

The remainder of the paper is organized as follows. Section II states formally the problem. In Section III we describe the proposed information gathering algorithm. Section IV presents the analysis performed to validate the algorithm through simulations. Section V describes the experimental results, followed by conclusions.

II. PROBLEM STATEMENT

We wish to explore an *a priori* unknown physical process with a robot as accurately as possible, in the sense of minimizing the difference between a process estimate and ground truth, and do so efficiently, i.e. consuming as little as possible of the limited resources such as time or energy. To this end, we devise in this paper movement strategies so as to reduce the posterior error over the unseen regions of the exploration space as efficiently as possible.

To achieve this, we make a few simplifying assumptions:

1) This physical process takes place in an environment populated with obstacles that are *a priori* known. This assumption allows us to abstract the exploration of the physical process from the mapping of the environment.
2) The physical process is time-invariant during the information gathering task.
3) The robot's position is known exactly and noise-free. Uncertainty in positioning can also be accounted for using e.g., GPs [20].

Let us now introduce the notation that we will use in the remainder of the paper. The robot position will be denoted by $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^3$, where $\mathcal{X}$ is the environment in which the robot operates. The environment $\mathcal{X}$ corresponds to the free space in the robot's configuration space. The robot's motion model is given by a function $\mathbf{x}(t + dt) = \mathbf{f}(\mathbf{x}(t), \mathbf{u})$ that relates the robot's current position $\mathbf{x}(t)$ and future position $\mathbf{x}(t + dt)$ given a control input $\mathbf{u}$. The physical process at position $\mathbf{x} \in \mathcal{X}$ is given by the variable $y(\mathbf{x}) \subset \mathbb{R}$. Typically, however, the process value is not observed directly, but measured using some sensors. Here we assume a simple sensor model
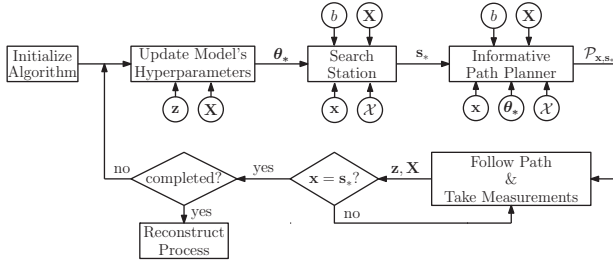
Fig. 2: Algorithm block diagram.

that represents a measured process as $z(\mathbf{x}) = y(\mathbf{x}) + \epsilon(\mathbf{x})$, where noise samples $\epsilon(\mathbf{x})$ are independent and identically distributed according to $\epsilon(\mathbf{x}) \sim \mathcal{N}(0, \sigma_n^2)$, i.e., they follow a Gaussian distribution with zero mean and variance σ_n^2.

In this work we propose an algorithm that allows a robot to autonomously decide where to measure next in order to reconstruct accurately and efficiently the physical process values $y(\mathbf{x})$ at any position $\mathbf{x} \in \mathcal{X}$. Naturally, this requires a model that can accurately represent the observed phenomenon. Here we make use of GPs for this purpose [8].

III. EFFICIENT INFORMATION GATHERING USING SAMPLING-BASED PLANNERS AND GPS

This section presents our proposed algorithm for information gathering. Section III-A shows an overview of the algorithm. Then, we describe in detail the two main components of the algorithm: (i) section III-B presents the algorithm used to search for a highly informative station; and (ii) section III-C describes our novel RRT*-based informative path planner. To finalize, we discuss in Section III-D the particular choice of the information quality metric.

A. Algorithm Overview

We propose an algorithm that allows a robot to autonomously and efficiently gather information of an unknown physical process with a robot of arbitrary dynamics within an environment populated with obstacles. The algorithm consists of two main steps: (i) first, we search for locations of high uncertainty –stations–, and (ii) second, we find a path towards the station that trades off informativeness and cost. The former is realized with a highly efficient algorithm that allows us to identify unexplored areas. And the latter is realized with a more complex algorithm that refines the path towards the station. It is the combination of both steps what allows us to achieve an efficient information gathering.

A block diagram of the whole scheme is shown in Figure 2. Our proposed algorithm works as follows: Firstly, the robot learns the best GPs model given the acquired measurements. This is done by maximizing the GPs log-marginal likelihood [8], finding the hyperparameters $\boldsymbol{\theta}_*$.

Once we estimate $\boldsymbol{\theta}_*$, the robot searches for a highly informative station $\mathbf{s}_*$ using the `SearchStation` algorithm (Section III-B). The algorithm takes as an input the robot's current position $\mathbf{x}$ and free state space $\mathcal{X}$, a budget constraint on the path cost b, and the hyperparameters $\boldsymbol{\theta}_*$ that allow

the robot to calculate the expected information contained at a station. In addition to the station $\mathbf{s}_*$, the algorithm outputs a suboptimal, yet feasible path $\mathcal{P}_s$ – an ordered list of waypoints $\boldsymbol{\tau} \in \mathcal{X}$ – between the robot's current position and the selected station, and the corresponding path utility u_s. More on the computation of the utility and its properties will be discussed in Sec. III-C.

Then the robot plans a trajectory from its current position $\mathbf{x}$ to the station $\mathbf{s}_*$ using the `InformativePlanner` algorithm (Section III-C). The algorithm result is a trajectory $\mathcal{P}_p$, together with its corresponding utility u_p that trades off the information gathering with the path's cost. The algorithm has an anytime nature. Then, it is possible that, for the planning time pre-defined by the user, the algorithm is either not able to find a path or the found path is of worse quality (in terms of used utility) than $\mathcal{P}_s$. To make sure that a solution is found, we compare the two found solutions and select the best path $\mathcal{P}_{\mathbf{x}, \mathbf{s}_*}$ according to our information quality metric.

Finally, the robot follows the trajectory $\mathcal{P}_{\mathbf{x}, \mathbf{s}_*}$ until it reaches the station $\mathbf{s}_*$, while it measures the physical process at positions $\boldsymbol{\tau}$ along the path, and incorporates those measurements and their respective locations to its knowledge database. Then, the main loop is repeated until some stopping criterion is fulfilled, e.g., maximum exploration time, or the remaining process uncertainty. Once the robot finishes gathering information, it can predict the value of the process $\boldsymbol{\mu}_*$ and the associated uncertainty $\boldsymbol{\Sigma}_*$ of the prediction for any point in $\mathcal{X}$ using GPs regression [8].

B. Search for Highly Informative Stations

Let us now consider the searching algorithm for a new station in more detail. As we mentioned, a station is related to a highly unexplored area; i.e. a position that is highly informative according to a pre-specified information quality metric. In addition, the search of a station must fulfill the following two requirements: (i) it must be reachable for the robot; (ii) its calculation must be efficient, and must have an anytime nature to allow the online realization of the algorithm. To realize these requirements, we propose an adaptation of the kynodinamic RRT algorithm [15] where we extend the RRT nodes to incorporate an information measure [13]. Specifically, here each node includes (i) a spatial location $\mathbf{x}_{\text{new}}$, (ii) the expected information at that location $\mathbf{x}_{\text{new}}$, and (iii) the cost of reaching $\mathbf{x}_{\text{new}}$ from the robot current position traversing the tree. Let us emphasize that RRT provides a quick way to "sort out" stations that are not reachable by the robot. Using e.g. RRT* for realizing this test is possible, but computationally less efficient.

Formally, the search of the optimal station $\mathbf{s}_*$ can be formulated as:

$$\mathbf{s}_* = \operatorname*{argmax}_{\mathbf{x}' \in \mathcal{X}} I(\mathbf{x}') \text{ s.t. } c(\mathcal{P}_{\mathbf{x}, \mathbf{x}'}) \leq b, \tag{1}$$

where $I(\mathbf{x}')$ is a measure of the expected information at position $\mathbf{x}'$ (the particular measure employed is described in Section III-D), $c(\mathcal{P}_{\mathbf{x}, \mathbf{x}'})$ is the cost of traversing path $\mathcal{P}_{\mathbf{x}, \mathbf{x}'}$, and b is some predefined maximum cost of a trajectory. We use the time needed to traverse a path as the path cost.

125

The optimization of (1) is realized with the `SearchStation` algorithm. This works as follows: first our adapted RRT grows a set of trajectories of at most cost b. Then we search the RRT node – station – $\mathbf{s}_*$ of highest expected information. Finally, the algorithm outputs a path $\mathcal{P}_s$ that links the robot's current position and the station, and the respective utility u_s of path $\mathcal{P}_s$.

C. Informative Path Planner using RRT*

Once the station is found, the algorithm plans the path between the robot's current position and the station. This path must fulfill the following two requirements: (i) it is feasible given the robot's dynamics and does not incur collisions with obstacles; (ii) it is efficient, in the sense of maximizing the information gathering while minimizing the path cost. To solve this problem, we propose the use of a modified version of RRT* that incorporates both the path's information and cost to the tree's nodes. In addition, we include a pruning condition that controls the tree's growth and keeps the algorithm's computational complexity constant with respect to the environment size.

Formally, we aim to find the optimal path $\mathcal{P}_{\mathbf{x},\mathbf{s}_*}$ between states $\mathbf{x}$ and $\mathbf{s}_*$. This can be formulated as the following optimization problem:

$$\underset{\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B} \subset \mathcal{X}}{\mathrm{argmax}} \quad f(I(\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B}), c(\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B}))$$
$$\text{s.t.:} \qquad \mathbf{x}_A = \mathbf{x}, \mathbf{x}_B = \mathbf{s}_* \qquad\qquad (2)$$
$$c(\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B}) < b.$$

Here $I(\cdot)$ and $c(\cdot)$ are the functions that evaluate the information and cost of the path, $f(\cdot,\cdot)$ is a function that evaluates the information-cost trade-off (the utility), and b a budget for the path cost. We summarize the `InformativePlanner` algorithm in Algorithm 1.

The RRT* algorithm is an evolution of the RRT algorithm that has been shown to be asymptotically optimal. It differs from RRT in two aspects: choosing a parent and rewiring. In contrast to RRT, the RRT* algorithm chooses the parent of the node $\mathbf{x}_{\mathrm{new}}$ as the node from the set of neighboring nodes $\mathcal{V}_{\mathrm{near}}$ that allows us to reach $\mathbf{x}_{\mathrm{new}}$ with the minimum cost. Here we replace the concept of the path cost by the concept of utility. The utility u of a path is a value that weights the importance of a path. In this paper, we formulate the utility, given by function $f(\cdot,\cdot)$, so that it fulfills our information gathering objective; i.e. we aim to gather as much information as possible along the path towards the station while generating trajectories with the minimum cost. The function $f(\cdot,\cdot)$ should grow with I and decrease as the path cost becomes large. Like in [19], [21], we represent this trade-off with the following function:

$$f(I(\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B}), c(\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B})) = \frac{I(\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B})}{c(\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B})}. \qquad (3)$$

The cost metric $c(\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B})$ is chosen as a time needed to traverse the path $\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B}$. The information quality metric $I(\mathcal{P}_{\mathbf{x}_A,\mathbf{x}_B})$, which we calculate with the function `InformationP` will be explained in detail in Section III-D.

Algorithm 1 `InformativePlanner`$(\mathbf{x}, \mathbf{s}_*, \boldsymbol{\theta}_*, \mathbf{X}, b, \mathcal{X})$

1: $\mathcal{V} \leftarrow \{\mathbf{x}\}; \mathcal{E} \leftarrow \emptyset; \mathcal{P}_p \leftarrow \emptyset; u_p \leftarrow -\infty;$
2:
3: **while** ! *StopPlanner* **do**
4:      $\mathbf{x}_{rand} \leftarrow$ `SampleFree`;
5:      $\mathcal{N}_{nearest} \leftarrow$ `Nearest`$(\mathbf{x}_{rand}, \mathcal{V})$;
6:      $\mathbf{x}_{\mathrm{new}} \leftarrow$ `Steer`$(\mathbf{x}_{\mathcal{N}_{nearest}}, \mathbf{x}_{rand})$;
7:      ▷ Check trajectory feasibility and budget constraint
8:      **if** `CollFree`$(\mathcal{P}_{\mathbf{x}_{\mathcal{N}_{nearest}}, \mathbf{x}_{\mathrm{new}}})$ AND $c_{\max} < b$ **then**
9:          $\mathbf{x}_{\max} \leftarrow \mathbf{x}_{\mathcal{N}_{nearest}}$;
10:          ▷ Calculate cost and information of path to node
11:          $c_{\max} \leftarrow c_{\mathcal{N}_{nearest}} + c(\mathcal{P}_{\mathbf{x}_{\mathcal{N}_{nearest}}, \mathbf{x}_{\mathrm{new}}})$
12:          $I_{\max} \leftarrow$ `InformationP`$(\mathbf{x}_{\mathrm{new}}, \mathcal{N}_{nearest}, \mathcal{G}, \boldsymbol{\theta}_*, \mathbf{X})$
13:          ▷ Calculate path utility
14:          $u_{\max} \leftarrow f(I_{\max}, c_{\max})$
15:          $\mathcal{V}_{\mathrm{near}} \leftarrow$ `Near`$(\mathbf{x}_{\mathrm{new}}, \mathcal{V})$;
16:          **for** $\mathcal{N}_{\mathrm{near}} \in \mathcal{V}_{\mathrm{near}}$ **do**   ▷ Choose parent
17:             **if** `CollisionFree`$(\mathcal{P}_{\mathbf{x}_{\mathcal{N}_{\mathrm{near}}}, \mathbf{x}_{\mathrm{new}}})$ **then**
18:                 $c_{\mathrm{new}} \leftarrow c_{\mathcal{N}_{\mathrm{near}}} + c(\mathcal{P}_{\mathbf{x}_{\mathcal{N}_{\mathrm{near}}}, \mathbf{x}_{\mathrm{new}}})$;
19:                 $I_{\mathrm{new}} \leftarrow$ `InformationP`$(\mathbf{x}_{\mathrm{new}}, \mathcal{N}_{\mathrm{near}}, \mathcal{G}, \boldsymbol{\theta}_*, \mathbf{X})$
20:                 $u_{\mathrm{new}} \leftarrow f(I_{\mathrm{new}}, c_{\mathrm{new}})$
21:                 ▷ Parent that results in highest path utility
22:                 **if** $u_{\mathrm{new}} > u_{\max}$ **then**
23:                     $\mathbf{x}_{\max} \leftarrow \mathbf{x}_{\mathcal{N}_{\mathrm{near}}}; c_{\max} \leftarrow c_{\mathrm{new}}$;
24:                     $I_{\max} \leftarrow I_{\mathrm{new}}; u_{\max} \leftarrow u_{\mathrm{new}}$
25:      ▷ Add node to tree
26:      $\mathcal{N}_{\mathrm{new}} \leftarrow\ <\mathbf{x}_{new}, I_{\max}, c_{max}>$;
27:      $\mathcal{V} \leftarrow \mathcal{V} \cup \{\mathcal{N}_{\mathrm{new}}\}; \mathcal{E} \leftarrow \mathcal{E} \cup \{\mathcal{P}_{\mathbf{x}_{\max}, \mathbf{x}_{\mathrm{new}}}\}$
28:      ▷ Discard the nodes that will create a cycle
29:      $\mathcal{V}'_{\mathrm{near}} \leftarrow$ `CyclesFree`$(\mathbf{x}_{\mathrm{new}}, \mathcal{V}_{\mathrm{near}}, \mathcal{G})$
30:      **for** $\mathcal{N}'_{\mathrm{near}} \in \mathcal{V}'_{\mathrm{near}}$ **do**   ▷ Rewire near nodes
31:          $c_{\mathrm{new}} \leftarrow c_{\mathcal{N}_{\mathrm{new}}} + c(\mathcal{P}_{\mathbf{x}_{\mathcal{N}_{\mathrm{new}}}, \mathbf{x}_{\mathcal{N}'_{\mathrm{near}}}})$;
32:          $I_{\mathrm{new}} \leftarrow$ `InformationP`$(\mathbf{x}_{\mathcal{N}'_{\mathrm{near}}}, \mathcal{N}_{\mathrm{new}}, \mathcal{G}, \boldsymbol{\theta}_*, \mathbf{X})$
33:          $u_{\mathrm{new}} \leftarrow f(I_{\mathrm{new}}, c_{\mathrm{new}})$
34:          **if** `CollisionFree`$(\mathcal{P}_{\mathbf{x}_{\mathcal{N}_{\mathrm{new}}}, \mathbf{x}_{\mathcal{N}'_{\mathrm{near}}}})$ **then**
35:             ▷ Rewire if higher path utility
36:             **if** $u_{\mathrm{new}} > f(I_{\mathcal{N}'_{\mathrm{near}}}, c_{\mathcal{N}'_{\mathrm{near}}})$ **then**
37:                 $I_{\mathcal{N}'_{\mathrm{near}}} \leftarrow I_{\mathrm{new}}; c_{\mathcal{N}'_{\mathrm{near}}} \leftarrow c_{\mathrm{new}}$;
38:                 $\mathcal{N}_{\mathrm{parent}} \leftarrow$ `Parent`$(\mathcal{N}'_{\mathrm{near}})$;
39:                 $\mathcal{E} \leftarrow \mathcal{E} \setminus \{\mathcal{P}_{\mathbf{x}_{\mathcal{N}_{\mathrm{parent}}}, \mathbf{x}_{\mathcal{N}'_{\mathrm{near}}}}\} \cup \{\mathcal{P}_{\mathbf{x}_{\mathrm{new}}, \mathbf{x}_{\mathcal{N}'_{\mathrm{near}}}}\}$;
40: $\mathcal{P}_p, u_p \leftarrow$ `FindBestPath`$(\mathbf{x}, \mathbf{s}_*, \mathcal{G})$;   ▷ Find best path to station
41: **return** $\mathcal{P}_p, u_p$;

The second main difference between RRT and RRT* lies on the incorporation of a rewiring process that allows the algorithm to approach the optimal trajectory. Let us remark that RRT* requires that two states must be connected exactly during the rewiring process. This implies that the system must be controllable. The rewiring is done by finding minimum-cost sub-paths that allow reaching the $\mathcal{V}'_{\mathrm{near}}$ nodes. However, the use of non-monotonic utility function (3) compromises the optimality guaranty of RRT*. Despite this, our simulation results suggest that our informative planner is still able to approach the optimal solution.

Furthermore, the non-monotonicity of our utility function (3) requires the inclusion of a mechanism to avoid the creation of cycles in the tree. A cycle is a sequence of vertices starting and ending at the same vertex. Let us emphasize that the inclusion of a mechanism to avoid cycles allows us to, in addition to RRT*, extend the algorithm to non-monotonic path quality functions, such as our proposed path quality measure or the one used by [19]. Cycles can be created

during the rewiring process if a node $\mathcal{N}'_{\text{near}}$, which belongs to the path that connects the robot's position with $\mathcal{N}_{\text{new}}$, could be reached with a higher utility from $\mathcal{N}_{\text{new}}$ than its previous utility. This problem does not arise with a monotonic utility function, since the inclusion of a new node always incurs a higher cost. Here, however, a longer path could have a higher utility if we gather more information along it. For this, we propose a procedure that is implemented before the rewiring process in function `CyclesFree` (line 35). This function takes as input $\mathbf{x}_{\text{new}}$ and $\mathcal{V}_{\text{near}}$, and removes those elements of $\mathcal{V}_{\text{near}}$ that belong to the path that links the robot's position with $\mathbf{x}_{\text{new}}$ by following graph $\mathcal{G}$.

Once we finish the execution of the algorithm, which is given by the $StopPlanner$ criterion, we calculate the best path in terms of utility that allows us to reach state $\mathbf{x}_B$ from $\mathbf{x}_A$ with function `FindBestPath`. This function takes all nodes in the graph and connects $\mathbf{s}_*$ to those nodes that are closer than a distance η from it. Then we calculate the utility of all those possible paths and choose the one with the highest utility. The utility, together with the computed path form the algorithm output.

D. Information Quality Metric

The formulated algorithm relies on an information quality metric to evaluate the informativeness of both, a position of the environment –stations– and a whole path. In this work, we use the entropy as information quality metric. Note that this quantity can be pre-computed at each station before running the path planning algorithm, which makes our metric computationally efficient. Moreover, the entropy has been shown to correlate well with the root mean squared error (RMSE) of a reconstructed process [22]. In other words, a position of high entropy will most likely incur a reduction of the RMSE between the actual process and its model-based estimation. Let us point out that the entropy at a particular location can be easily calculated from the predicted process covariance [2].

The computation of the information of a particular path translates to summing all the pre-computed individual entropies of those positions that belong to the path. However, in contrast to, e.g. MI, the sum of entropies does not have a diminishing property since it does not consider the cross-correlations [9]. To overcome this issue, we propose in this paper the mean entropy as information quality metric. This is primarily motivated by the concept of entropy rate, which is a limit of the joint entropy as the number of observations grows. Entropy rate converges to the averaged entropy as a special case. In fact, in contrast to the sum of entropies, the mean entropy presents a diminishing property. For example, imagine two paths that have the same sum of entropies. An averaged entropy would favor the one that requires fewer measurements. Formally we define our information quality metric as a mean entropy $\bar{H}(\mathcal{P}_{\mathbf{x},\mathbf{x}'})$ along the path:

$$I(\mathcal{P}_{\mathbf{x},\mathbf{x}'}) = \bar{H}(\mathcal{P}_{\mathbf{x},\mathbf{x}'}) = \frac{1}{|\mathcal{P}_{\mathbf{x},\mathbf{x}'}|} \sum_{\boldsymbol{\tau} \in \mathcal{P}_{\mathbf{x},\mathbf{x}'}} H(\boldsymbol{\tau}) \quad (4)$$

where $|\cdot|$ is the cardinality of a set, i.e. the number of vertices of path $\mathcal{P}_{\mathbf{x},\mathbf{x}'}$, $\boldsymbol{\tau} \in \mathcal{X}$ are the positions that compose the path, and $H(\boldsymbol{\tau})$ is the entropy of a single position.

To finalize, let us motivate the use of entropy against a state-of-the-art information metric: mutual information. Mutual information is a metric that has been extensively employed in the literature as we described in Section I. Indeed, it seems like a perfect fit for selecting information sampling locations because it takes into account the cross-correlations of the test points. However, we have observed that such a function is not adequate for algorithms that require an extensive computation of the information quality metric, as it is pointed out in [23]. Simulations results presented in Section IV-B.2 demonstrate this hypothesis.

IV. SIMULATIONS AND DISCUSSION OF RESULTS

In this section we present the simulations setup and performance results of the proposed algorithm. We divide this analysis in two main parts. First, we compare our informative path planner against some state-of-the-art algorithms. Second, we evaluate our whole algorithm in an active sensing task of an unknown physical process that takes place within a complex environment.

A. Simulations Setup

Here we describe the simulation setup used to validate the algorithm with synthetic and real data. We use the PyGPs[1] library to perform the GPs regression and learning of the hyperparameters. We carry out each of the simulations 40 times.

We test our algorithm with simulated and real data as ground truth. These data are stored as grids of 20×20 cells with a resolution of 5 and 10 centimetres. The estimated mean of the GP at the centre of these cells will be used to analyze the error of the estimated model with respect to the ground truth. Please notice that the exploration is performed in continuous space, and no discretization is required.

For the simulation we assume a round-shaped holonomic robot with 5 centimetres radius that moves with a constant speed of 0.2 metres per second. Here we employ a holonomic-robot to abstract the active sensing strategy from the robot's motion. However, the proposed algorithm is valid for arbitrary robot's dynamics given the restrictions imposed by RRT or RRT* algorithms. We also assume that the robot needs an infinitesimally small time to take a measurement. The robot can move in a continuous space, and measurements taken within one cell of the grid are assumed to be equal within the limits of additive noise.

For the `SearchStation` and `InformativePlanner` algorithms we select the following parameters: the parameter η and the distance employed by the function `Near` are both set to the measurement's resolution, i.e., 5 or 10 centimetres. We select the running time as stop criterion for the `SearchStation` and `InformativePlanner` algorithms, with a value of 5 and 10 seconds respectively. The budget

---

[1]pyGPs - A Package for Gaussian Processes.
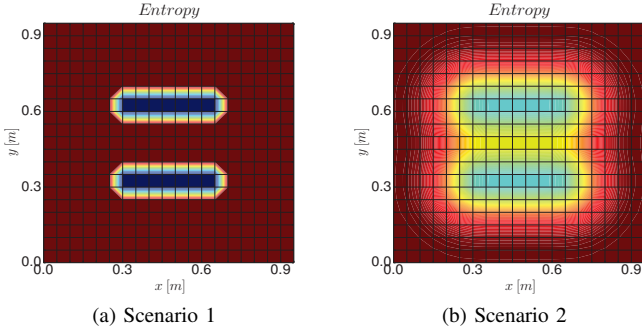http://www-ai.cs.uni-dortmund.de/weblab/static/api_docs/pyGPs/.

(a) Scenario 1   (b) Scenario 2

Fig. 3: Simulation scenarios used to test the performance of our informative path planner algorithm.

	t_{first}[s]	Entropy [bits]	Cost [s]
Mean Entropy	**6.31**	-6.93	**6.79**
Mutual Information	46.71	-3.54	6.86

TABLE I: Analysis of the information function. We compare our proposed information function (mean entropy) with the mutual information. We perform this comparison in terms of the time to find a first path (t_{first}) from the initial position to the station, posterior entropy calculated over the complete environment that results after measuring along the path, and cost of the resulting path.

parameter b is set to 10 seconds, which corresponds to a planning horizon of 2 metres given the robot's speed. We initialize the GPs hyperparameters to the following values $l = 1$, $\sigma_f = 1$, $\sigma_n = 0.1$.

B. Analysis of the Informative Path Planner

1) Setup: In this section, the objective is to analyze the performance of the informative path planner alone. This assumes an available GP model with fixed hyperparameters. The goal of our informative path planner algorithm is, given this model, to find the trajectory that optimizes equation (2) as fast as possible. Since our information function corresponds to the mean entropy, we simulate two scenarios with different entropy structures (see Figure 3):

- Scenario 1 recreates a physical process with low spatial correlation in which a robot has already gathered two patches of measurements (blue areas). We employ the following hyperparameters: $l = 0.02$, $\sigma_f = 0.084$, $\sigma_n = 0.02$.
- Scenario 2 recreates the same scenario, but now we consider a process with higher spatial correlation. Here we set $l = 0.13$, $\sigma_f = 0.084$, $\sigma_n = 0.09$.

We fix the agent's initial position to $(x = 0.2, y = 0.5)$ and set the station at $(x = 0.8, y = 0.5)$ for all the simulations runs.

2) Choice of the Information Quality Metric: We argue in Section III-D that we employ as information metric mean entropy instead of mutual information for computational reasons. We test this hypothesis by running Algorithm 1 and replacing the information metric given by (4) with the mutual information. Results correspond to the average over 40 simulations runs for a planning time of 180 seconds. We show the results in Table I. We can conclude that our information metric, given by equation (4) outperforms the mutual information in an online sensing setting.

3) Performance Analysis: We benchmark our algorithm, to which we will refer as `Algorithm`, against two state-of-the-art sampling-based informative path planning algorithms:

- (i) the technique of [17], where multiple paths are obtained by running the RRT planner several

times, and the paths are then evaluated according to the information metric; this algorithm we will term `Multiples RRT`;
- (ii) the RIG-tree planner [13], to which we will refer as `RIG Algorithm`.

In both cases, the same utility function as in our algorithm is employed. For the RIG-tree we use one of the approaches suggested by the authors in [13]. Specifically, we consider the pruning based on the heuristic that the utility function is modular. Here, we have defined two nodes as co-located if they are within the same cell of the grid. For more details about the implementation, please refer to the original paper [13]. We have also tested several other alternatives proposed by the authors, but they offered a lower performance in our particular setup.

Moreover, the two benchmark algorithms are not designed to meet a goal constraint; they explore the environment with no goal. This makes a comparison with our algorithm difficult. We solve this by selecting all samples that are closer than a distance η from the goal, and then connecting them to the goal position. This results in paths that link the agent's position with the goal.

We compare the three considered algorithms in terms of an independent metric – the posterior entropy – that results after measuring the planned path. Let us remark that we calculate the posterior entropy over the complete environment, and we consider 180 seconds of planning time. In addition, we compare the cost of the resulting paths. Table II shows the results for the three algorithms. We can conclude that our algorithm is the one that offers the best ratio entropy-cost for all scenarios. Specifically, the most relevant scenario for this evaluation is scenario 2 since it presents a higher spatial correlation. Let us remark that for this scenario our algorithm results in a twofold and sevenfold increase respect to the other alternatives while offering a similar path cost.

C. Analysis of the Exploration Strategy

1) Setup: We validate in this section our information gathering strategy in an environment that emulates a corridor with different rooms (see Figure 4). In this section the full technique is analyzed. Specifically, the system uses both `SearchStation` and `InformativePlanner` algorithms for exploration and learns online the parameters of the GP model.
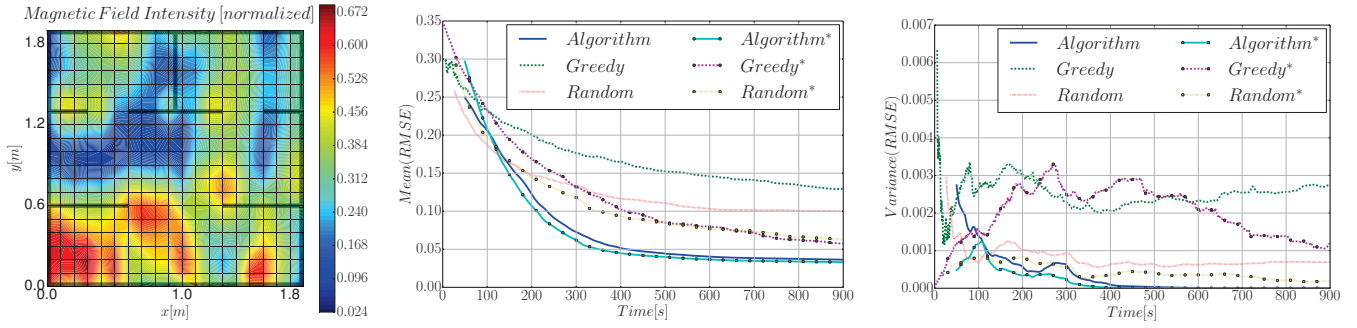
128

Fig. 4: Left: scenarios employed to test our proposed information gathering strategy. The black polygons correspond to the obstacles and the underlying picture is the magnetic field intensity we aim to explore. Right: mean and variance of the root mean squared error (RMSE) between the estimation of the process and the ground truth.

		Entropy [bits]	Path Cost [s]
Scenario 1	Algorithm	**383.25**	**3.03**
	Multiples RRT	383.75	3.26
	RIG Algorithm	383.86	4.24
Scenario 2	Algorithm	**−6.93**	6.79
	Multiples RRT	−3.68	6.23
	RIG Algorithm	0.03	**5.65**

TABLE II: Path cost and posterior entropy evaluated over the complete environment after measuring the resulting path given 180 seconds of planning time.

The simulations employ real data that corresponds to the magnetic field intensity in an indoor environment. This data was collected with a ground-based robot equipped with a magnetometer at the German Aerospace Center (DLR).

2) Performance Analysis: The goal of our active-sensing information gathering algorithm is to reduce the RMSE between ground truth and estimation of the process as fast as possible. Let us remark that we also consider for the calculation of the RMSE the positions that are below the obstacles and those that are not reachable by the robot. In this section, we compare the devised strategy for exploration with the following strategies:

- Greedy approach: the next station is selected as a neighboring cell according to the cell size of the ground truth data, which is then also the next pose for the robot.
- Random approach: an RRT is grown from the current pose for the same planning time and budget b as in SearchStation algorithm. The next station is selected randomly from the leaves of the RRT and the associated path is employed to reach the station.

Let us clarify our motivation to employ the greedy and random benchmarks. The algorithms Multiples RRT and RIG Algorithm are one-shot algorithms. In other words, given an *a priori* known model the algorithms run for some user pre-defined time and produce a path. This is also what our informative planner (Algorithm 3) is designed to do. This was the motivation to compare the algorithms with each other. In this section we evaluate our full exploration

strategy (Algorithm 1) that is able to explore an *a priori* unknown process. Let us remark again that Multiples RRT and RIG Algorithm do not consider this feature. In the literature, two common approaches to deal with an exploration of an unknown process are a random trajectory, and a greedy one [2], [10]. Therefore, we used these two to benchmark our exploration strategy. We believe that the proposed evaluation is the fairest to compare each of the different approaches according to its capabilities.

Figure 4 shows the mean and variance of the RMSE between the actual process and our estimation for all executions. We also test the methods under assumption that the optimal hyperparameters are known and fixed (listed with an asterisk sign). Observe that the proposed informative algorithm clearly outperforms the other strategies. The greedy approach with the optimal hyperparameters is the only one that offers a comparable performance after the 900 seconds mission. Note, however, that it requires prior knowledge about the process.

To finalize, let us compare our full exploration strategy with the RIG Algorithm. Although this algorithm and our strategy have a different motivation, as previously pointed out, we compare both of them to get a better understanding of our proposed algorithm capabilities. Specifically, we consider the following for the RIG Algorithm: (i) the model is *a priori* known; i.e. we know the GPs hyperparameters and they do not need no be estimated, (ii) the utility function corresponds to the mutual information, as suggested by the authors in [13], and (iii) the planning time is 600 seconds and then we let the robot follow and measure along the planned path. Let us remark that these are favorable conditions for the RIG Algorithm as our algorithm assumes an *a priori* unknown model that needs to be estimated online. We have run the simulation 40 times starting from different positions in the environment. Then we have calculated the RMSE between the ground truth and the process estimation after measuring along the calculated path. The average RMSE that we obtained for the RIG Algorithm is 0.27, which is much higher than the one obtained by our algorithm that is 0.05. We believe that the lower performance of the RIG Algorithm lies on the fact that the algorithm grows a

129

single tree to explore the complete environment. Notice that the complexity of adding a new sample grows exponentially as the tree grows, which difficult the exploration of the complete environment. In contrast, our algorithm runs multiple consecutive trees using our devised two-step approach that allows to identify unexplored areas and to perform an efficient exploration.

V. Experiments and Discussion of Results

In addition to the simulations, we have performed an experiment in an indoor laboratory environment populated with obstacles (see Figure 1). Experimental results demonstrate that our proposed strategy is able to decrease the RMSE to approx. $1.24\ \mu T$, which represents a 1200% improvement respect to a random and greedy trajectory. A video that shows the real-time execution of the experiment is attached to the paper.

VI. Conclusions and Future Work

In this work we have presented an active sensing approach for efficient information gathering in environments populated with obstacles using GPs. The GPs model allows to capture spatial correlations of the process, and are used to make predictions about the process value and the corresponding uncertainty at locations not yet visited by the robot. The predicted process uncertainty guides the robot trajectories: it select stations where the uncertainty within a certain range from the current position is the highest. To reach the stations we propose a modification of the RRT* algorithm that also exploits the learned GP model for informative planning. Specifically, the algorithm trades-off the cost of the generated path with the information gained while traversing the generated route.

The presented algorithm has been validated using synthetic data, which illustrates well the impact of information gain on the generated trajectories. Also, an experiment with a single robot has been performed with the goal of exploring a magnetic field in a laboratory populated with obstacles. Results show that the devised information gathering strategy achieves a 1200% improvement respect to the two considered state-of-the-art approaches.

As future work, we aim to extend the proposed algorithm to handle uncertainty both in the robot's pose and motion. In addition we should consider as well the mapping of the environment. Another natural extension for the algorithm is to consider the multi-robot case. This way we could benefit from the correct coordination between robots to achieve a much higher performance.

References

[1] P. Dames, M. Schwager, D. Rus, and V. Kumar, "Active magnetic anomaly detection using multiple micro aerial vehicles," Robotics and Automation Letters, IEEE, 2015.

[2] A. Viseras, T. Wiedemann, C. Manss, L. Magel, J. Mueller, D. Shutin, and L. Merino, "Decentralized multi-agent exploration with online-learning of gaussian processes," in 2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2016, pp. 4222–4229.

[3] A. Singh, A. Krause, and W. J. Kaiser, "Nonmyopic adaptive informative path planning for multiple robots," in Twenty-First International Joint Conference on Artificial Intelligence, 2009.

[4] K. H. Low, J. M. Dolan, and P. Khosla, "Adaptive multi-robot wide-area exploration and mapping," in Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1. International Foundation for Autonomous Agents and Multiagent Systems, 2008, pp. 23–30.

[5] A. Viseras Ruiz and C. Olariu, "A general algorithm for exploration with gaussian processes in complex, unknown environments," in Robotics and Automation (ICRA), 2015 IEEE International Conference on. IEEE, 2015, pp. 3388 – 3393.

[6] D. S. Levine, "Information-rich path planning under general constraints using rapidly-exploring random trees," Master's thesis, Citeseer, 2010.

[7] B. J. Julian, S. Karaman, and D. Rus, "On mutual information-based control of range sensing robots for mapping applications," The International Journal of Robotics Research, vol. 33, no. 10, pp. 1375–1392, 2014.

[8] C. E. Rasmussen and C. K. Williams, Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press, 2005.

[9] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," The Journal of Machine Learning Research, vol. 9, pp. 235–284, 2008.

[10] A. Krause and C. Guestrin, "Nonmyopic active learning of gaussian processes: an exploration-exploitation approach," in Proceedings of the 24th international conference on Machine learning. ACM, 2007, pp. 449–456.

[11] B. Yamauchi, "A frontier-based approach for autonomous exploration," in Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on. IEEE, 1997, pp. 146–151.

[12] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser, "Efficient informative sensing using multiple robots," Journal of Artificial Intelligence Research, pp. 707–755, 2009.

[13] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms," The International Journal of Robotics Research, vol. 33, no. 9, pp. 1271–1287, 2014.

[14] J. L. Nguyen, N. R. Lawrance, R. Fitch, and S. Sukkarieh, "Real-time path planning for long-term information gathering with an aerial glider," Autonomous Robots, pp. 1–23, 2015.

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

[16] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," The International Journal of Robotics Research, vol. 30, no. 7, pp. 846–894, 2011.

[17] K. Yang, S. Keat Gan, and S. Sukkarieh, "A gaussian process-based rrt planner for the exploration of an unknown and cluttered environment with a uav," Advanced Robotics, vol. 27, no. 6, pp. 431–443, 2013.

[18] X. Lan and M. Schwager, "Planning periodic persistent monitoring trajectories for sensing robots in gaussian random fields," in Robotics and Automation (ICRA), 2013 IEEE International Conference on. IEEE, 2013, pp. 2415–2420.

[19] B. Charrow, S. Liu, V. Kumar, and N. Michael, "Information-theoretic mapping using cauchy-schwarz quadratic mutual information," in IEEE Int. Conf. on Robotics and Automation (ICRA), 2015.

[20] M. L. Srikar, T. Svensson, and H. Wymeersch, "Spatial Wireless Channel Prediction under Location Uncertainty," IEEE Trans. Wirel. Commun., 2015, to appear.

[21] K. Ma, Z. Ma, L. Liu, and G. S. Sukhatme, "Multi-robot informative and adaptive planning for persistent environmental monitoring," in Proceedings of the 13th International Symposium on Distributed Autonomous Robotic Systems, DARS, 2016.

[22] S. Vasudevan, F. Ramos, E. Nettleton, and H. Durrant-Whyte, "Gaussian process modeling of large-scale terrain," Journal of Field Robotics, vol. 26, no. 10, pp. 812–840, 2009.

[23] R. Stranders, A. Farinelli, A. Rogers, and N. R. Jennings, "Decentralised coordination of mobile sensors using the max-sum algorithm," in Proceedings of the 21st international joint conference on Artifical intelligence. Morgan Kaufmann Publishers Inc., 2009, pp. 299–304.