Multirobot Online Construction of Communication Maps

Jacopo Banfi[1], Alberto Quattrini Li[2], Nicola Basilico[3], Ioannis Rekleitis[2], and Francesco Amigoni[1]

***Abstract*— The importance of communication in many multirobot information-gathering tasks requires the availability of reliable communication maps. These provide estimates of the radio signal strength and can be used to predict the presence of communication links between different locations of the environment. In the problem we consider, a team of mobile robots has to build such maps autonomously in a robot-to-robot communication setting. The solution we propose models the signal's distribution with a Gaussian Process and exploits different online sensing strategies to coordinate and guide the robots during their data acquisition. Our methods show interesting operative insights both in simulations and on real TurtleBot 2 platforms.**

I. Introduction

Communication is a fundamental activity for multirobot systems, for exchanging information between robots that need to cooperate in some tasks. Applications like surveillance or search and rescue heavily rely on sharing knowledge among robots to ensure situation awareness and to enable informed autonomous decision making during the mission. The importance of this issue in multirobot applications is being increasingly recognized as testified by the rich literature on communication-aware multirobot systems [1], [2].

Deciding "where to go next" is a key problem in many multirobot settings, usually addressed by optimizing the selection of locations according to some task-related objective function [3]. Communication often comes as a further requirement of seeking locations from where robots can exchange data on a wireless connection, either with a fixed base station [4] or with teammates [5]. Independently from the application domain, robots often need to possess knowledge about the possibility of establishing wireless communication links between *arbitrary pairs of locations* before moving there. In this work, we call such knowledge *communication map* and we address the problem, for a team of robots, to build it from the measurements collected in the environment.

In the literature, communication-aware multirobot systems often assume to have a communication map in advance. Such a map is specified by a graph whose nodes and edges represent locations and communication links, respectively. The works presented in [1], [6] are two representative examples of this approach in the domain of informative path planning and exploration. In such settings, the construction of the communication map must happen offline and by means of a link detection mechanism which, given a map of the environment, computes which pairs of nodes should be connected with an edge because communication is possible. As discussed in [7], one important limit of this approach is that robots' capability of communicating over wireless channels is heavily influenced by a number of environmental features from which graph-based representations necessarily abstract away (examples are the density of obstacles or the presence of interferences). This makes offline link detection either too conservative (e.g., limited-distance line-of-sight), or unreliable (e.g., wall propagation model [8]) and poses the need for methods based on signal strength measurements gathered on the field. In this paper, we introduce a formal representation of communication maps exploiting Gaussian Processes [9]. GPs are a tool widely used in robotics especially in applications involving spatial physical phenomena that have to be mapped or monitored [10]–[12]. We then tackle the problem for a team of mobile robots to autonomously construct such maps from on-the-field measurements of WiFi signal strength.

Fig. 1: Four robots used to build a communication map.

Similar versions of this problem have been addressed in [13], for single-robot exploration and radio source localization, in [14], for radio signal mapping with a stationary radio source and no coordination between robots, and in [15]. The latter proposes offline methods to compute efficient joint paths for small teams of robots with the aim of collecting signal strength measures from predefined locations. We consider a more challenging online scenario with robot-to-robot communications and propose some multirobot sensing strategies to build low uncertainty maps with reasonable costs. We developed and evaluated our methods in simulation and on a team of TurtleBot 2 platforms (see Fig. 1).

II. Communication Maps

We consider m mobile robots deployed in a known environment where free space is denoted with $\mathcal{A} \subset \mathbb{R}^2$ and $\mathbf{p}_i \in \mathcal{A}$ is any location that can be occupied. We

[1] J. Banfi and F. Amigoni are with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy {jacopo.banfi,francesco.amigoni}@polimi.it.

[2] A. Quattrini Li and I. Rekleitis are with the Deparment of Computer Science and Engineering, University of South Carolina, Columbia, SC, USA {albertoq,yiannisr}@cse.sc.edu.

[3] N. Basilico is with the Department of Computer Science, University of Milan, Milano, Italy nicola.basilico@unimi.it.

assume that the robots can localize themselves within a global coordinate system and that are endowed with an omni-directional transceiver. For example, a WiFi adapter for transmitting and receiving data with peers over the radio channel within a given maximum range.

A *communication map* should provide information about the availability of links between ordered pairs of locations. We define it as a function $\hat{f} : \mathcal{A} \times \mathcal{A} \rightarrow \mathbb{R}_{\leq 0}$ estimating the (equally defined) radio signal strength f between locations $\mathbf{p}_i$ and $\mathbf{p}_j$. Such measure is defined as the receiving power in dBm at location $\mathbf{p}_j$ with respect to a source placed at $\mathbf{p}_i$. It directly relates to the link estimation over radio transmissions: the closer it gets to zero the more reliable the transmissions from $\mathbf{p}_i$ to $\mathbf{p}_j$ and, therefore, the more likely the availability of a high-bandwidth communication link from $\mathbf{p}_i$ to $\mathbf{p}_j$. In sufficiently large environments, we can exclude *a priori* the availability of some communication links given the indicative range of the transceiver R_c. This allows us to focus on locations lying within R_c.

To ease notation, let us assume that $\mathbf{x}_{ij} = (\mathbf{p}_i, \mathbf{p}_j)$ and call $\hat{f}(\mathbf{x}_{ij})$ the estimate of the signal strength from $\mathbf{p}_i$ to $\mathbf{p}_j$. Communication links need not to be symmetric so, in general, $f(\mathbf{x}_{ij}) \neq f(\mathbf{x}_{ji})$ [16].

A Gaussian Process is a set of random variables where each finite subgroup follows a Gaussian multi-variate distribution [9]. Exploiting a GP seems a natural choice to build $\hat{f}$ for the following reasons. First, it must describe a physical phenomenon (the signal strength distribution over $\mathcal{A}^2$) which might exhibit significant spatial correlations. Second, besides providing estimates for the signal strength, it must also characterize the uncertainty of such estimates. Such a scenario is similar to robotic mapping problems with entropy-like measures over the partially explored regions of the environment. For instance, it is reasonable to expect that communication-aware navigation strategies using $\hat{f}$ to evaluate candidate locations would be able to make more informed decisions by weighting the uncertainty of the obtained signal strength estimates. These requirements are met by GPs in a principled way and, for this reason, they find employment in several robotic applications like exploration [11], environmental monitoring [10], and localization [7].

Within this framework, we aim at estimating $\hat{f}$ as a posterior distribution fitted over a set of noisy observations made by robots which explore and coordinate in the environment to gather signal strength measurements. Let us assume that the robot team as a whole collected q measurements over the environment. Call $\mathbf{Y} = [y^1, y^2, \ldots, y^q]^T$ the set of those measurements and $\mathbf{X} = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^q]^T$ the set of the corresponding pairs of locations from where they have been collected (recall that $\mathbf{x}^i \in \mathcal{A}^2$). Clearly $y^i = f(\mathbf{x}^i) + \epsilon$ where the additive sensing error is assumed to be i.i.d. and $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$. The spatial correlation between any two values of f is expressed with a covariance function taking as input the two location pairs corresponding to each value. Such a function is denoted as $k(\mathbf{x}, \mathbf{x}')$ and can be defined in different ways. In this work, following the mainstream approach we adopt a radial basis kernel (RBF):

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\Big(- \frac{|\mathbf{x} - \mathbf{x}'|^2}{2l^2} \Big). \tag{1}$$

where σ_f^2 is the signal variance and l^2 length scale.

To ease notation, given $X_1 = [x_1^1, \ldots, x_1^a]^T$ and $X_2 = [x_2^1, \ldots, x_2^b]^T$, we denote with $K(X_1, X_2)$ the $a \times b$ matrix where $K_{ij} = k(x_1^i, x_2^j)$ and with I_q the $q \times q$ identity matrix. The correlation between the observed function values is then given by

$$\text{cov}(\mathbf{Y}) = K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I_q \tag{2}$$

The kind of GP considered in this work (with a zero mean function) is then fully specified by the parameter vector $\theta = [\sigma_n^2, \sigma_f^2, l^2]^T$ which is computed as the one maximizing the observations log-likelihood. Namely, $\theta^* = \arg\max_\theta \log p(\mathbf{Y} \mid \mathbf{X}, \theta)$ where:

$$\log p(\mathbf{Y} \mid \mathbf{X}, \theta) = \\ -\frac{1}{2}\Big(\mathbf{Y}^\mathbf{T} \text{cov}(\mathbf{Y})^{-1} \mathbf{Y} - \log|\text{cov}(\mathbf{Y})| - n \log 2\pi \Big). \tag{3}$$

The obtained parameters can then be used to calculate an estimate of the signal strength in unobserved regions by evaluating the posterior. In particular, called $\mathbf{W} = [\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^l]^T$ a set of arbitrary location pairs for which a signal strength estimate is requested, $p(f(\mathbf{W}) \mid \mathbf{X}, \mathbf{Y}) \sim \mathcal{N}(\mu_\mathbf{W}, \mathbf{\Sigma}_\mathbf{W})$ where the mean vector is obtained as $\mu_\mathbf{W} = K(\mathbf{W}, \mathbf{X}) \text{cov}(\mathbf{Y})^{-1} \mathbf{Y}$ and represents the estimate $\hat{f}(\mathbf{W})$, while the covariance matrix is given by $\mathbf{\Sigma}_\mathbf{W} = K(\mathbf{W}, \mathbf{W}) - K(\mathbf{W}, \mathbf{X}) \text{cov}(\mathbf{Y})^{-1} K(\mathbf{W}, \mathbf{X})^T$. In particular, the main diagonal of $\mathbf{\Sigma}_\mathbf{W}$ is usually called predictive variance and is used to measure the uncertainty of estimates in $\mathbf{W}$.

The GP provides a mechanism to integrate noisy readings collected in the environment into a posterior distribution of the signal strength that can be used to obtain link estimates with quantified uncertainty. To deploy such a mechanism in real multirobot settings we need to tackle two additional issues. The first is to design a mission execution scheme according to which robots repeatedly coordinate, gather sensor data, share information, and update the communication map maximizing Eq. (3). The second problem is to design utility functions to optimize the online selection of joint data-gathering locations. Both problems are central in the definition of what we call a *sensing strategy*.

III. Sensing Strategies

A signal strength measurement at the joint location $\mathbf{x}_{ij}$ is obtained by a robot at location $\mathbf{p}_i$ polling another robot at location $\mathbf{p}_j$. The polling frequency between two moving robots can be set arbitrarily, but, in practice, high frequencies collecting large datasets would require large computational efforts for the GP parameter estimation (maximizing Eq. 3 takes $O(\#samples^3)$). As a consequence, the number of measurements must be limited to a set of few but significant samples. Our sensing strategies deal with this problem by privileging data acquisition locations that are expected to induce high reductions in the current map's uncertainty.

Moreover, robot teams might have non-homogeneous computational capabilities, a sensitive issue for the GP parameter estimation process. We capture this by considering two settings. In the *homogeneous* setting, we assume that any robot is equipped with sufficient computational power to construct the GP model. In the *non-homogeneous* setting, we restrict this feature to an elite of robots. Real multirobot systems are often composed by many cheap/basic platforms that can navigate and partner up in taking measurements but cannot do intensive on-board computations which only a limited number of more expensive robots can afford.

Both strategies are based on a *leader-follower* paradigm, because of the necessary coordination between measurements and robots. Leaders are robots in charge of maintaining a communication map by iteratively estimating the GP parameters that best fit the data acquired so far. They are also in charge of selecting the best locations to be visited in coordination with the corresponding followers. In general, we allow for multiple leaders to be present in the robotic team, assuming a rendez-vous phase at the end of the mission in which the leaders meet to merge their communication maps in the final one. Coordination between different groups of leader-followers is achieved by broadcasting or selectively relaying relevant information to be shared in a multi-hop fashion. In practice, the two strategies give rise to a "grouply distributed" multirobot system. Each group of robots acts as an autonomous entity and benefits from episodic encounters with robots belonging to other groups, while, at the same time, each follower is subject to the directives of the corresponding leader. Both strategies favor selection of locations in regions of $\mathcal{A}^2$ currently displaying high predictive variance, trying to spread the robots.

A. Pairwise Mapping in Homogeneous Settings

With the Pairwise Mapping (PM) strategy the team is divided in pairs of robots, where one robot acts as *leader* and the other as *follower*. The leader iteratively drives itself and the follower to take measurements in the locations $\mathbf{p}_l^*, \mathbf{p}_f^* \in \mathcal{A}^2, |\mathbf{p}_l^* - \mathbf{p}_f^*| \leq R_c$ currently displaying a high predictive variance in the current communication map. While moving to such locations, the two robots poll other robots in the environment for additional measurements. The polling frequency is adapted from the estimated mission length, to limit the number of acquired samples. Coordination between different leader-follower pairs is achieved in two ways. First, each robot broadcasts the waypoints of its current path, along with those of its corresponding teammate, so that the other leaders know which regions of $\mathcal{A}^2$ can be excluded from planning as they will be visited and sensed by other robots. Second, each robot maintains an updated collection of all the data gathered by the team by periodically asking its teammates to selectively relay the portion of the collected dataset not yet received by any other team member. This is done to make the most updated dataset available to each leader for training a new GP at replanning time.

In case two robots are not able to communicate from two target destinations $\mathbf{p}_l^*, \mathbf{p}_f^*$, a recovery mechanism is adopted.

Algorithm 1 PM planning algorithm for a pair r_l, r_f

Input: $\mathcal{D}$ (dataset), $\mathbf{p}_l, \mathbf{p}_f$ (current positions), $\mathcal{P}$ (teammate paths), $\mathcal{M}$ (environment map), n_s^{PM} (# samples), $d_{\min}$ (minimum distance)
Output: $\mathbf{p}_l^*, \mathbf{p}_f^*$(target positions), $\mathbf{p}_l^b, \mathbf{p}_f^b$(backup positions)
1: $\mathcal{J} \leftarrow$ `extrapolateJointLocations`($\mathcal{P}$)
2: $\mathcal{X} \leftarrow$ `sampleLocations`($\mathcal{M}$, n_s^{PM}, $\mathcal{J}$, $d_{\min}$)
3: $\theta \leftarrow$ `learnHyperParams`($\mathcal{D}$)
4: $\mathbf{\Sigma}_{\mathcal{X}} \leftarrow$ `predictUncertaintyGP`($\mathcal{X}$, $\mathcal{D}$, θ)
5: $\mathbf{p}_1^*, \mathbf{p}_2^* \leftarrow \arg\max_{(\mathbf{p}_1,\mathbf{p}_2)\in\mathcal{X}}\{\mathbf{\Sigma}_{\mathbf{xx}} + \mathbf{\Sigma}_{\mathbf{x}'\mathbf{x}'}\}(\mathbf{x} = (\mathbf{p}_1, \mathbf{p}_2), \mathbf{x}' = (\mathbf{p}_2, \mathbf{p}_1))$
6: $\mathbf{p}_l^*, \mathbf{p}_f^* \leftarrow$ `assignMinMaxDist`($\mathbf{p}_1^*, \mathbf{p}_2^*, \mathbf{p}_l, \mathbf{p}_f$)
7: $\mathbf{p}_l^b, \mathbf{p}_f^b \leftarrow$ `backupDest`($\mathbf{p}_l^*, \mathbf{p}_f^*, \mathcal{M}$)

Specifically, a pair of backup destinations are selected and assigned to robots. The selection criterion, see below, guarantees the presence of a communication link. Called r_l and r_f the leader and follower robots, the PM strategy is formally defined by these steps:

(1) r_l and r_f are connected;
(2) r_l selects the target destinations, $\mathbf{p}_l^*, \mathbf{p}_f^*$, and backup destinations $\mathbf{p}_l^b, \mathbf{p}_f^b$ (see Algorithm 1), informing r_f;
(3) r_l and r_f agree on a deadline t_d to reach $\mathbf{p}_l^*$ and $\mathbf{p}_f^*$;
(4) r_l and r_f move to $\mathbf{p}_l^*$ and $\mathbf{p}_f^*$, opportunistically exchanging the collected signal data and the undertaken path with other teammates, and possibly polling them to get additional measurements;
(5) if r_l and r_f are connected before t_d, go to (1); otherwise, go to the next step;
(6) r_l and r_f set $\mathbf{p}_l^b$ and $\mathbf{p}_f^b$ as destinations; as soon as they regain connection, go to (1).

In Step (3), the value of t_d can simply be chosen as the maximum estimated arrival time of the two robots, augmented by some tolerance value; 4% in our experiments.

Algorithm 1 describes how each leader chooses the next pairs of target and backup destinations. The algorithm takes as input the collected signal data $\mathcal{D} = \{\mathbf{X}, \mathbf{Y}\}$, the current leader-follower positions $\mathbf{p}_l, \mathbf{p}_f$, the paths currently undertaken by the other leader-follower pairs $\mathcal{P}$, and a map of the physical environment $\mathcal{M}$. As additional parameters, the algorithm requires the number of samples n_s^{PM} to generate, and a value $d_{\min}$ used to exclude some location pairs from planning as they will be probably close to pairs already visited by other subteams. Initially (Lines 1-2), the algorithm samples candidate pairs of locations $\mathcal{X}$ in which to send r_l and r_f. Specifically, it first computes the set $\mathcal{J}$ of joint waypoints that the other leader-follower teams will traverse while moving to their target destinations, assuming a constant speed along their path; then, it selects n_s^{PM} samples from $\mathcal{A}^2$ at distance not greater than R_c, pruning those that are not at least d_{min} far apart (in travel distance) from each location pair in $\mathcal{J}$. In Line 3, a new GP is trained with the gathered data, and in Lines 4-5 the most uncertain pair $\mathbf{p}_1^*, \mathbf{p}_2^* \in \mathcal{X}$ is selected. The two locations are then assigned to the robots to minimize the maximum traveled distance (hence optimizing the energy consumption), thus producing $\mathbf{p}_l^*, \mathbf{p}_f^*$ (Line 6). Finally, in Line 7 backup destinations $\mathbf{p}_l^b, \mathbf{p}_f^b$ are chosen as the pair of points closest to the target locations, and for which a safe communication link is guaranteed. (Section IV

presents a possible way of computing backup destinations.)

B. Region Mapping in Non-Homogeneous Settings

The Region Mapping (RM) strategy is again based on a leader-follower paradigm, but this time leader robots (the only ones capable of computing the GP model) can dispose of an arbitrary but fixed number of followers $R_f = \{r_{f_1}, \dots, r_{f_k}\}$. As in the previous strategy, leaders are in charge of maintaining the GP model. However, now they iteratively drive the followers in *regions* R^* with high predictive variance. Once a region has been selected, the leader moves in its center $\mathbf{p}_c$. At the same time, the followers move towards safe positions $\mathcal{S}_f = \{\mathbf{p}_1^s, \dots, \mathbf{p}_k^s\}$ from which they can acknowledge that the leader has reached its goal. Then, the followers move along pre-computed paths $\mathcal{P}_f = \{p_1, \dots, p_k\}$ that can significantly reduce the prediction uncertainty in the region of $\mathcal{A}^2$ centered in $\mathbf{p}_c$. If a follower r_{f_i} cannot communicate with the leader at the end of its path, it will move to its backup destination chosen from a pre-computed set $\mathcal{B}_f = \{\mathbf{p}_1^b, \dots, \mathbf{p}_k^b\}$. The coordination among teams is achieved by letting the leaders choose regions to visit whose centers are sufficiently far apart. As with PM strategy, robots can also acquire additional measurements, while selectively relaying the gathered dataset. The rationale behind this strategy is that it might be more convenient to lower down the uncertainty around a fixed leader position now, rather than take only sparse measurements with the possible need of re-visiting the same region later. Formally, leader r_l and followers R_f act as follows:

(1) r_l is initially connected to each follower in R_f;
(2) r_l decides the region R^* to explore next, computing $\mathbf{p}_c$, $\mathcal{P}_f$, $\mathcal{S}_f$, $\mathcal{B}_f$ (see Algorithm 2);
(3) r_l moves to $\mathbf{p}_c$, while the followers move to $\mathcal{S}_f$;
(4) followers follow their paths in $\mathcal{P}_f$, r_l remains still;
(5) followers regain connection with r_l either in their last path waypoints, or by resorting to their backup destinations in $\mathcal{B}_f$; go to (1).

Before passing from Step (3) to Step (4), once the leader arrives at $\mathbf{p}_c$, it waits until all followers communicate with the leader. In this way, excessive delays of some of them can be handled by reorganizing the paths of the remaining ones.

Algorithm 2 describes the RM strategy. The algorithm takes as input the collected signal data $\mathcal{D}$, the current leader and followers positions $\mathbf{p}_l$, $\mathbf{P}_f$, the centers of the regions being visited by other groups $\mathbf{P}_c$ as known by r_l, a map of the physical environment $\mathcal{M}$, the number of regions to sample n_s^{RM}, a parameter $d_{\min}$ used for coordinating the spreading of the different groups, and the number of waypoints to visit in the new region n_w. As with the previous strategy, a suitable choice of $d_{\min}$ avoids that two different subteams of robots with a close replanning time decide to start mapping the same region of the environment.

Algorithm 2 RM planning algorithm for a group r_l, R_f

Input: $\mathcal{D}$ (dataset), $\mathbf{p}_l$, $\mathbf{P}_f$ (current followers positions), $\mathbf{P}_c$ (other region centers), $\mathcal{M}$ (environment map), n_s^{RM} (# region samples), $d_{\min}$ (minimum distance), n_w (# waypoints)
Output: $\mathbf{p}_c$ (new region center), $\mathcal{P}_f$ (followers' paths), $\mathcal{S}_f$ (initial positions), $\mathcal{B}_f$ (backup positions)
1: $\mathcal{R} \leftarrow$ `sampleRegions`$(\mathcal{M}, n_s^{\text{RM}}, \mathbf{P}_c, d_{\min})$
2: $\theta \leftarrow$ `learnHyperParams`$(\mathcal{D})$
3: $\overline{\Sigma}[\mathcal{R}] \leftarrow$ `meanVarianceRegionsGP`$(\mathcal{R}, \mathcal{D}, \theta)$
4: $R^* \leftarrow \arg\max_{R \in \mathcal{R}} \overline{\Sigma}[R]$
5: $\mathcal{W} \leftarrow$ `getWaypoints`$(R^*, n_w, \mathcal{D}, \theta)$
6: $\mathcal{S}_f \leftarrow$ `initialDests`$(\mathbf{P}_f, \mathcal{W}, \mathcal{M})$
7: $\mathcal{P}_f \leftarrow$ `assignMinMaxDistPaths`$(\mathcal{S}_f, \mathcal{W}, \mathcal{M})$
8: $\mathcal{B}_f \leftarrow$ `backupDest`$(\mathcal{P}_f, \mathcal{M})$

In Line 1, n_s^{RM} candidate circular regions $\mathcal{R}$ with diameter $2R_c$ are selected. In Line 2, a new GP is trained with the gathered data, and in Lines 3-4 the most uncertain region $R^* \in \mathcal{R}$ is selected as the region displaying the highest mean predictive variance. In Line 5, a set $\mathcal{W}$ of n_w waypoints to visit is selected from a fine-grained discretization of $\mathcal{A} \cap R^*$ according to the following method: iteratively choose the point $\mathbf{p}_w$ displaying the highest sum of predictive variance when paired with $\mathbf{p}_c$ and sufficiently far apart from the already chosen waypoints (this distance threshold could also be the same $d_{\min}$). Notice that this spacing is required because it would be inefficient to chose a waypoint very close to another one, since the uncertainty of the former will be already reduced by visiting the latter. In Line 6, the initial safe destinations for the followers $\mathcal{S}_f$ are selected as follows: first, followers are iteratively assigned to the closest locations in $\mathcal{W}$ guaranteeing a safe communication link to $\mathbf{p}_c$, consistently updating $\mathcal{W}$ so that, at the end of this first phase, $\mathcal{W} \cap \mathcal{S}_f = \emptyset$. Then, if any follower remains unassigned, its corresponding safe destination is chosen as the closest point from its current position in $\mathcal{A}$ guaranteeing a safe communication link. In Line 7, the remaining waypoints are assigned to the robots with the aim of minimizing the maximum followers' traveled distance, and finally in Line 8 the backup destinations are computed as in the PM strategy.

Note that, in Line 7, the objective of minimizing the bottleneck traveled distance gives rise to the Multiple Traveling Salesman Path Problem, which is NP-hard [17]. This problem can be formalized in terms of a simple Mixed Integer Linear Program (MILP) as follows. With a slight notation overload, let f be a generic follower robot, and $V(f)$ be the set $\mathcal{W} \cup \{\mathbf{p}_f^s\}$. We define three sets of variables: x_{ij}^f, with $i, j \in V(f)$, is a binary variable taking value 1 iff in the path of f a visit to i is followed by a visit to j; y_i^f, with $i \in \mathcal{W}$, is a binary variable taking value 1 iff waypoint i is visited by f; u_i^f, with $i \in \mathcal{W}$, is a continuous variable for representing the possible position of waypoint i in the path of f. A continuous variable b is also defined to represent the objective function value. The MILP model reads as follows:

$$\min b \quad \text{s.t.} \tag{4}$$

$$\sum_{j \in V(f)} x^f_{\mathbf{p}^s_f j} = \sum_{i \in V(f)} x^f_{i \mathbf{p}^s_f} \quad \forall f \in R_f \tag{5}$$

$$\sum_{j \in V(f)} x^f_{\mathbf{p}^s_f j} = 1 \quad \forall f \in R_f \tag{6}$$

$$\sum_{i \in V(f)} x^f_{iw} = \sum_{j \in V(f)} x^f_{wj} \quad \forall f \in R_f, \forall w \in \mathcal{W} \tag{7}$$

$$\sum_{j \in V(f)} x^f_{wj} = y^f_w \quad \forall f \in R_f, \forall w \in \mathcal{W} \tag{8}$$

$$\sum_{f \in R_f} y^f_w = 1 \quad \forall w \in \mathcal{W} \tag{9}$$

$$b \geq \sum_{i \in V(f)} \sum_{j \in \mathcal{W}} d_{ij} x^f_{ij} \quad \forall f \in R_f \tag{10}$$

$$2 \leq u^f_w \leq |\mathcal{W}| + 1 \quad \forall f \in R_f, \forall w \in \mathcal{W} \tag{11}$$

$$u^f_i - u^f_j + 1 \leq |\mathcal{W}|(1 - x^f_{ij}) \quad \forall f \in R_f, \forall i, j \in \mathcal{W}. \tag{12}$$

Constraints (5) and (6) enforce, for each follower, a path that starts and ends at the corresponding $\mathbf{p}^s_f$. Constraints (7) and (8) guarantee path consistency. Constraints (9) impose that each waypoint must be visited by exactly one follower. Constraints (10) bind the objective function to the maximum traveled distance (notice that starting points are excluded from the inner summation, as we are not interested in building a tour). Finally, Constraints (11) and (12) avoid the presence of subtours. The time required to solve this model to optimality rapidly grows with the size of the input. However, preliminary experiments showed that it is usually better to keep the size of $\mathcal{W}$ relatively small (i.e., only a few points), as robots can also periodically obtain measurements while moving along their planned paths.

IV. Experimental Evaluations

We validate our strategies with Stage [18] simulations (with teams of 2, 4, and 6 robots) and on four TurtleBot 2 robots. (We used ROS [19] and GPy [20] as framework for GPs.) For both strategies, backup locations are determined according to this criterion: two robots can always communicate within a distance of $R_c/3$, or $R_c/2$ if in line-of-sight.

PM and RM are compared against a baseline strategy, RAND, where robots independently move to random destinations while polling teammates. To ensure acceptable real-time performance, we conducted a preliminary experimental phase to set appropriate polling frequencies for PM, RM, and RAND. (Differently from [14], we assume that our GPs have a constant mean function, so that only covariance parameters need to be optimized.) Note that RAND is an uninformed strategy not requiring any GP training for deciding where to go next. Still, we assume that the data it collects will be used, at some point, to build a communication map. For PM and RM, we discard the samples acquired by a leader and its followers during the computation of plans, since such robots will remain still at a fixed position. The polling period of RAND is set to a higher value compared to PM and RM to ensure a fair comparison based on a comparable amount of gathered samples. Specifically, we use the following polling periods: for 2 s robots, 3 s for PM and RM, 3.5 s for RAND; for 4 robots, 5 s for PM and RM, 10 s for RAND; for 6 robots, 10 s for PM and RM, 18 s for RAND.

Fig. 2: Simulation environments (Office, left, and Cluttered, right), approximate size 80 m × 30 m.

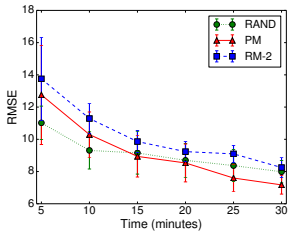
The strategies are evaluated by considering the quality of the GPs that would be obtained by merging all the collected data in a global rendez-vous after 5 min (see Fig. 3(a) for the GPs training times with 4 robots in one of the simulation environments). In particular, quality is measured both in terms of Rooted Mean Squared Error (RMSE) on a given test set (10000 samples for simulations, 2000 for real robots, collected randomly), as well in terms of the average *predictive standard deviation* (i.e., the squared root of the predictive variance) of the predictions. The latter metric is fundamental for an online scenario and should be as high as possible, as other information is typically not available.

A. Simulations

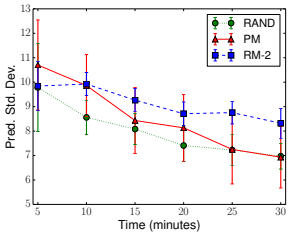
We simulate communications using the signal propagation model of [8]. The signal strength S at a distance d_m from the emitting source is computed as $S = P_{d_0} - 10P_l \log_{10}(d_m/d_0) - W_f \min(C, N_w)$, where P_{d_0} denotes the signal strength at the reference distance d_0, P_l is the path loss factor, W_f is the wall attenuation factor, N_w is the number of walls between emitter and receiver, and C is the maximum number of walls. In our case, we set P_{d_0} =−38 dBm, d_0 =1 m, $P_l = 2.3$, $W_f = 3.37$, $C = 5$. A bidirectional communication link is available between any two simulated robots if $S \geq -93$ dBm in both directions. The indicative communication range R_c is set to 50m.

We select two representative environments of realistic size depicted in Fig. 2. "Office" is a portion of the "sdr_site_b" environment from the Radish repository [21], while "Cluttered" is inspired from the "grass" environment of the MRESim repository [22]. The mission duration is set to 30 minutes. For the RM strategy, we assume the presence of at most 2 leaders (we use RM-N to denote the results for the RM strategy where each group is composed of N robots). We use the following set of parameters, obtained from preliminary experiments: n_s^{PM}=10000, n_s^{RM}=100 for all the team sizes, $d_{\min}$=25, 20, 15 m for 2, 4, and 6 robots, respectively. The n_w parameter is chosen to allow a fair coverage of a region, compatibly with the mission duration and accounting for the available number of robots. In particular, n_w=$3|R_f|, 2|R_f|, 1|R_f|$ for 2, 4, and 6 robots, respectively. We execute 5 runs for each experimental setting.

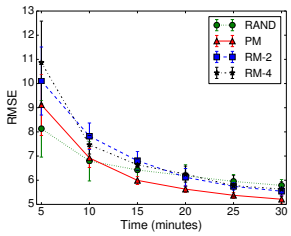
Fig. 3 reports the results obtained in the Office environment. Focusing on the prediction performance for 2 robots (Fig. 3(a)-(b)), we can see that all the strategies are able to significantly lower down both the RMSE and the predictive standard deviation. Comparing PM against RM, we can



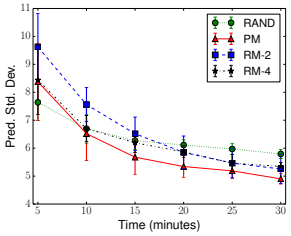
(a) 2 robots, RMSE



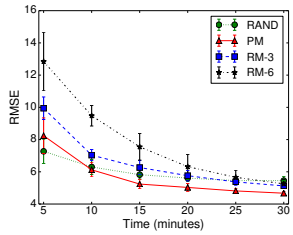
(b) 2 robots, pred. std. dev.



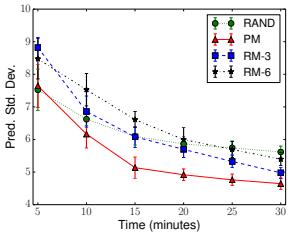
(c) 4 robots, RMSE



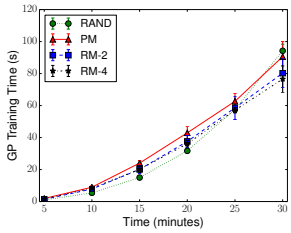
(d) 4 robots, pred. std. dev.



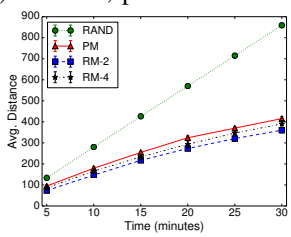
(e) 6 robots, RMSE



(f) 6 robots, pred. std. dev.



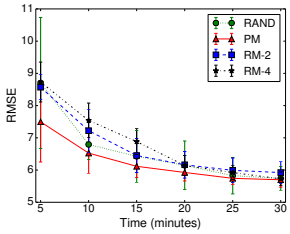
(g) 4 robots, GP training time



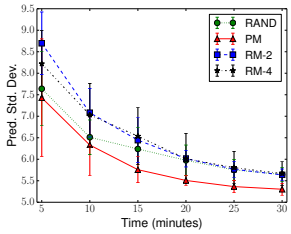
(h) 4 robots, avg. distance

Fig. 3: Simulation experiments, Office environment.

observe an advantage in terms of both performance metrics. However, compared to RAND, PM does not seem to provide a significant advantage: in fact, RAND also provides the best performance in terms of predictive standard deviation across the whole mission. This means that, when the extension of the spatial phenomenon to learn is large (compared to the available robots), the advantage provided in terms of performance may be not enough to justify the use of complex mapping strategies. Looking at the results with 4 (Fig. 3(c)-(d)) and 6 (Fig. 3(e)-(f)) robots, we can observe an advantage in the use of PM compared to RAND, with RM still not behaving as well as PM overall. For the RMSE, the advantage offered by PM is slight, but often statistically significant (e.g., p-value=0.004064 in one-way ANOVA at 30 min between PM and RAND for 4 robots). Looking at the predictive standard deviation, we can observe a substantial advantage in the usage of PM towards the middle of the mission, which is consistently maintained until the end and

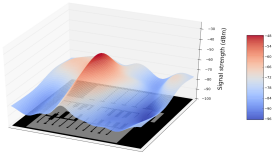


(a) 4 robots, RMSE

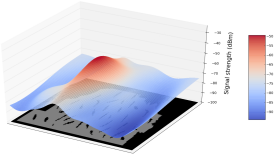


(b) 4 robots, pred. std. dev.

Fig. 4: Simulation experiments, Cluttered environment.



(a) Office



(b) Cluttered

Fig. 5: Maps for a source located in the center of the simulation environments (6 robots, PM strategy).

is statistically significant (e.g., p-value=0.00005615 at 30 min between PM and RAND for 6 robots). In general, the results obtained by RM suggest that it could not be convenient to spend too much time on lowering down the uncertainty of a single region. However, we argue that, by setting the n_w parameter equal to 1, we could obtain a performance similar to that of PM even for 2 and 4 robots. The relatively comparable performance of RAND comes from the fact that this strategy uniformly sample the spatial phenomenon (recall that robots keep poll each other). Obviously, RAND has a remarkable downside in the distance it requires the robots to travel, making it a very inefficient sampling strategy. Fig. 3(h) shows an example of such results for 4 robots.

The results for the Cluttered environment are very similar to those of Office, and all the above considerations still hold. For reasons of space, we only report in Fig. 4 the results obtained for 4 robots. Fig. 5 shows two communication maps built for a source located in the center of the simulation environments for 6 robots and the PM strategy.

B. Experiments with Real Robots

We also deployed and tested our algorithms on a team of four TurtleBots 2. Each platform is equipped with a Microsoft Kinect and an on-board laptop with an integrated WiFi card. We run some communication map construction missions in the Swearingen Engineering Center at the University of South Carolina. The maps used for localization are built in a setup phase where a single robot is manually driven around the environment to collect readings which are processed by the ROS gmapping package [23]. An example of map is depicted in Fig. 6 (the size is about 50 by 30 m). We tuned the parameters selecting those values which achieved the best performance in simulations up to some refinement, using insights from simulated experiments. For example, the timeout t_d needs to be increased due to locomotion noise

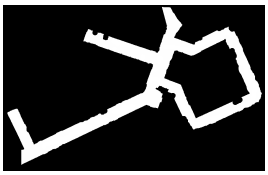


Fig. 6: Part of a floor in the department of Computer Science and Engineering at the University of South Carolina.

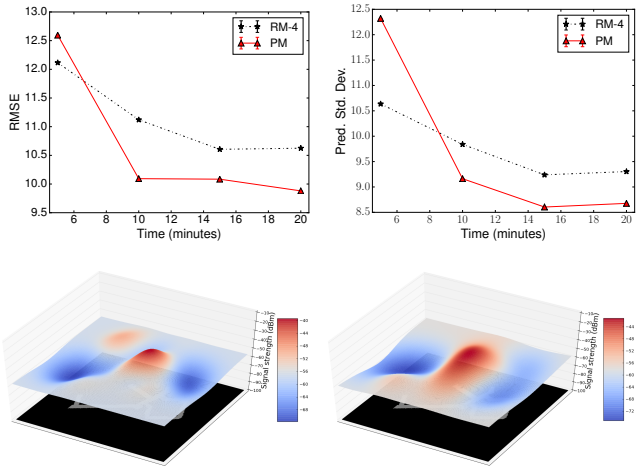


Fig. 7: Comparison between PM and RM and the estimated communication map from the T-intersection of the corridors at the center of the map in Fig. 6 (left PM; right RM).

which, in the real world, is clearly non-negligible.

The trends of the quantitative results are comparable to those obtained in simulation (see Fig. 7), and the robots travel comparable distances. However, the values of uncertainty and MSE are higher compared to simulation: real signal strength is much more complex than the model used. Despite this, the communication map that is built by GP looks consistent with the obstacles; e.g., Fig. 7 shows a 2D communication map, by fixing one location for the two strategies. Since we experience some noise in localization, the associated WiFi strength measurements few times are not accurate. Further, although our approach relies not only on the selected destination location, but also on the paths of each robot to compute the utility function, few times the robots interfered with the motion of others, especially using the PM strategy, which has a lower coordination level compared to RM.

V. Conclusions

In this work, we designed and tested multirobot sensing strategies for mapping the availability of WiFi communication links in an environment. We provided a formalism based on Gaussian Processes to represent the WiFi signal strength distribution and devised sensing strategies to build and maintain it with teams of (homogeneous or non-homogeneous) autonomous mobile robots. Results from simulation and real-world experiments show how distributed coordination schemes can effectively perform such mapping task.

In future works, we will consider problem settings where the task of building communication maps coexists with other online tasks, like exploration or search. Assessing how the optimal resolution of one task can benefit or penalize the other is an interesting challenge towards the definition of strategies that try to combine them.

Acknowledgment

The authors would like to thank the generous support of the National Science Foundation grant (NSF 1513203 and 1637876).

References

[1] G. Hollinger and S. Singh, "Multirobot coordination with periodic connectivity: theory and experiments," *IEEE T Robot*, vol. 28, no. 4, pp. 967–973, 2012.

[2] E. Stump, N. Michal, V. Kumar, and V. Isler, "Visibility-based deployment of robot formations for communication maintenance," in *Proc. ICRA*, 2011, pp. 4498–4505.

[3] N. Basilico and F. Amigoni, "Exploration strategies based on multi-criteria decision making for searching environments in rescue operations," *Auton Robot*, vol. 31, no. 4, pp. 401–417, 2011.

[4] Y. Pei, M. Mutka, and N. Xi, "Connectivity and bandwidth-aware real-time exploration in mobile robot networks," *Wirel Commun Mob Comput*, vol. 13, no. 9, pp. 847–863, 2013.

[5] E. Jensen, E. Nunes, and M. Gini, "Communication-restricted exploration for robot teams," in *AAAI Multiagent Interaction without Prior Coordination Workshop*, 2014.

[6] J. Banfi, A. Quattrini Li, N. Basilico, F. Amigoni, and I. Rekleitis, "Asinchronous multirobot exploration under recurrent connectivity constraints," in *Proc. ICRA*, 2016, pp. 5491–5498.

[7] B. Ferris, D. Fox, and N. Lawrence, "WiFi-SLAM using Gaussian Process latent variable models." in *Proc. IJCAI*, 2007, pp. 2480–2485.

[8] P. Bahl and V. N. Padmanabhan, "Radar: An in-building RF-based user location and tracking system," in *Proc. INFOCOM*, 2000, pp. 775–784.

[9] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[10] R. Marchant and F. Ramos, "Bayesian optimisation for informative continuous path planning," in *Proc. ICRA*, 2014, pp. 6136–6143.

[11] 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 *Proc. ICRA*, 2016, pp. 4222–4229.

[12] A. Singh, F. Ramos, H. Whyte, and W. Kaiser, "Modeling and decision making in spatio-temporal processes for environmental surveillance," in *Proc. ICRA*, 2010, pp. 5490–5497.

[13] J. Twigg, J. Fink, L. Paul, and B. Sadler, "RSS gradient-assisted frontier exploration and radio source localization," in *Proc. ICRA*, 2012, pp. 889–895.

[14] J. Fink and V. Kumar, "Online methods for radio signal mapping with mobile robots," in *Proc. ICRA*, 2010, pp. 1940–1945.

[15] M.-Y. Hsieh, V. Kumar, and C. Taylor, "Constructing radio signal strength maps with multiple robots," in *Proc. ICRA*, 2004, pp. 4184–4189.

[16] K. Heurtefeux and F. Valois, "Is rssi a good choice for localization in wireless sensor network?" in *Proc. AINA*, 2012, pp. 732–739.

[17] I. Rekleitis, A. New, E. Rankin, and H. Choset, "Efficient boustrophedon multi-robot coverage: an algorithmic approach," *Ann Math Artif Intel*, vol. 52, no. 2, pp. 109–142, 2008.

[18] R. Vaughan, "Massively multiple robot simulations in stage," *Swarm Intel*, vol. 2, no. 2-4, pp. 189–208, 2008.

[19] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[20] GPy, "GPy: A gaussian process framework in python," http://github.com/SheffieldML/GPy, since 2012.

[21] A. Howard and N. Roy, "The robotics data set repository (Radish)," http://radish.sourceforge.net/, 2003.

[22] V. Spirin, J. de Hoog, A. Visser, and S. Cameron, "MRESim, a multi-robot exploration simulator for the rescue simulation league," in *Proc. RoboCup*, 2014, pp. 106–117.

[23] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE T Robot*, vol. 23, no. 1, pp. 34–46, 2007.