SWARM TECHNOLOGIES FOR FUTURE SPACE EXPLORATION MISSIONS

Emanuel Staudinger[1], Dmitriy Shutin[1], Christoph Manß[1], Alberto Viseras Ruiz[1], Siwei Zhang[1]

[1]*German Aerospace Center (DLR) e.V., Institute of Communications and Navigation, Oberpfaffenhofen, Germany, E-mail: firstname.lastname@dlr.de*

ABSTRACT

Modern robotic platforms for in-situ space exploration are single-robots equipped with a number of specialized sensors providing scientists with unique information about a planet's surface. However, there is a number of exploration problems where large spatial apertures of the exploration system are necessary, requiring a completely new perspective on in-situ space exploration and it's required technologies.

Large networks of robots, called swarm, pave the way: agents in a swarm span ad-hoc communication networks, localize themselves based on radio signals, share resources, process data and make inference over the network in a decentralized fashion. By cooperation, local information collected by agents becomes globally available. In this work we present our recent results in development of swarm technologies for future in-situ space exploration missions: a wireless system jointly used for communication and localization, and swarm navigation and exploration strategies to sample and reconstruct static spatial fields.

1 INTRODUCTION

Space exploration has been traditionally relying heavily on remote sensing technologies. While these greatly enhanced our knowledge of the cosmos, it is in-situ exploration systems that will pave the way for human colonization of our solar system and support our search for life on other planets. Already in the seventies the Soviet and American robotic systems demonstrated that in-situ exploration of Moon and Venus is possible. With the Philae Lander the European Space Agency (ESA) has recently shown for the first time that it is possible to land on a comet. In-situ Mars exploration with Curiosity rover, future ExoMars and Mars2020 rovers clearly demonstrate that in-situ Mars exploration will intensify.

Modern robotic rover platforms, like Curiosity or ExoMars exemplify well the state-of-the-art in in-situ space exploration. These are single-unit sophisticated mobile platforms that are primarily remotely operated, and the degree of autonomy is still very low. Equipped with a number of specialized sensors, they provide scientists with valuable and quite unique information about the planets surface. However, there is a number of exploration problems where a large spatial aperture of the exploration system is required, e.g., for seismological investigations, fast terrain scouting, or guiding the exploration system back to the lander over very large distances. Such applications require a completely new perspective on in-situ space exploration that is discussed in this work.

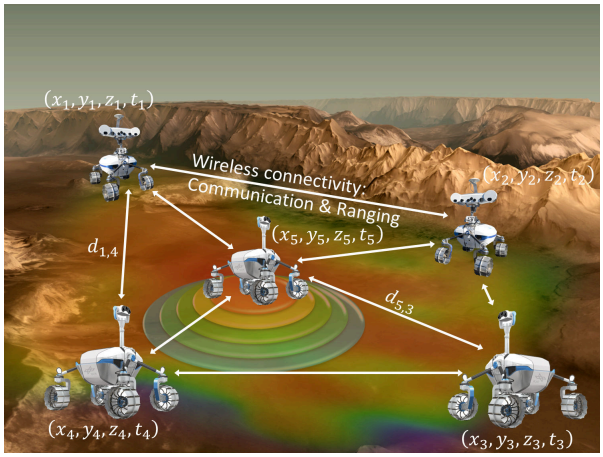


Figure 1: Multi-agent network – a swarm – spanning an aperture for seismological investigations. Position and time of taken measurements are important to reconstruct geological structures. Wireless signals are jointly used for communication and estimating the distances among agents for localization.

In particular, we consider a multi-agent system - a swarm - that consists of multiple mobile robotic platforms designed specifically for a particular sensing task, e.g., seismography. While to some extent inspired by nature, swarms in the context of space exploration do differ from their biological counterparts. Agents in the swarm span a communication network that they use to autonomously localize themselves, share resources, and, most importantly, process data and make inference over the network in a decentralized fashion. The latter makes the whole system more robust against failures of individual agents.

Fig. 1 shows a swarm of rovers spanning a sensor aperture for seismological investigations: each agent can excite the underground with acoustic waves, while the other agents are in a listening mode. By sharing their

measurements in the network, agents are able to preliminarily reconstruct the underground geological structure and can move to new positions with a new sensor aperture. Such an automated seismological sensor network requires inter-agent communication, location and time information to relate measurements for reconstruction, and methods to infer from the globally available reconstruction where agents should take new measurements.

The goal of this work is to demonstrate some of our recent results in development of swarm technologies for future space exploration missions, and we deliberate on the details as follows: in Sec. 2 we give an overview of our proposed wireless communication and radio-localization system. Sec. 3 shows details and results of multi-objective optimization algorithms taking radio-localization and multi-agent control jointly into account to enable new sensor apertures. Two aspects of swarm exploration are addressed in Sec. 4: distributed exploration strategies to learn a representation of a static spatial field and distributed multi-agent coordination under complex constraints. In Sec. 5 we conclude and give an outlook on future work.

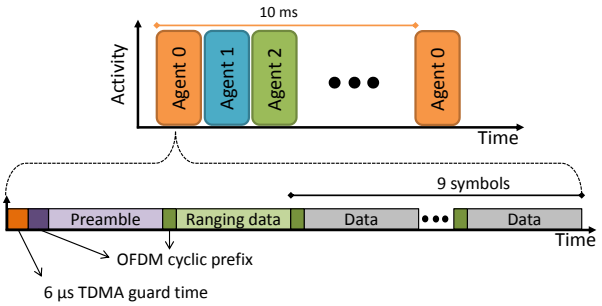


Figure 2: TDMA slot scheme and OFDM symbols per slot. In this example the channel access update rate per agent is set to 10 ms.

Table 1: Swarm communication system gross-throughput for a bandwidth of 25 MHz.

Throughput [Mbit/s]	QAM-4	QAM-16	QAM-64
Per agent	1.7	3.4	5.1
Aggregated	43.1	68.3	102.4

2 WIRELESS COMMUNICATION AND RADIO-LOCALIZATION

The swarm depicted in Fig. 1 can be seen as mobile meshed wireless sensor network. Each measurement from a scientific instrument taken by an agent must be stamped with a time and precise location information. Agents making inference over the network in a decentralized fashion require ad-hoc communication, localization, and timing. Distributed localization and exploration algorithms, as well as distributed control within the swarm require a broadcast communication scheme and high update rates with low to medium sized data packets. Hence, we propose a concept for a swarm Communication-Position-Navigation-Timing (CPNT) system and deliberate on the details summarized as three building blocks next.

2.1 Decentralized Channel Access

Organizing the channel access of the shared radio channel is one of the main challenges. Data packet collisions on the medium access control (MAC) layer result in poor communication performance and poor update rates for distributed algorithms. We propose time division multiple access (TDMA) as an efficient MAC protocol and orthogonal frequency division multiplex (OFDM) signal modulation to combat multipath and enable high-rate communication as the first building block. Each TDMA time slot is accessible for a single agent exclusively to avoid access interference. State-of-the-art communication systems using TDMA require a central unit, e.g., a base station or master, to provide global time slot boundaries on which the users in the network synchronize. This central unit is a single point of failure (SPOF). Hence, we use a decentralized TDMA slot synchronization based on a pulse coupled oscillator (PCO) principle to achieve a self-organized slot synchronization.

The design of the OFDM communication system is mainly based on the maximum expected channel excess delay of 2 µs, a maximum number of 20 agents in the swarm, and an update rate of 100 Hz for the overall network. OFDM signal bandwidth is flexible and chosen as multiple of 25 MHz, with 1024 subcarriers per 25 MHz. Fig. 2 illustrates the TDMA and OFDM signal framing. Each agent has access to the wireless channel every 10 ms and the illustrated increasing TDMA slot assignment is an example. The first OFDM symbol is an agent-specific preamble symbol comprising Zadoff-Chu sequences and is used for TDMA slot synchronization, OFDM time- and frequency-synchronization and channel estimation, identifying the emitting agent, and for ranging. The second OFDM symbol is currently reserved for ranging data, see Sec. 2.2. Nine OFDM symbols are dedicated for universal communication, e.g., exchanging information and distributed control: existing coding and resource allocation techniques can be applied, such as from 3GPP-LTE or WiFi. Tab. 1 summarizes achievable gross-throughputs for a signal bandwidth of 25 MHz and single-input single-output (SISO) communication.

2.2 Ranging with Wireless Signals

The second building block of Swarm-CPNT is ranging: estimating distances between agents based on the emitted radio signal. The swarm is not perfectly synchronized in time, as each agent uses its own local clock. Various time-based ranging techniques based on round-trip delay (RTD) of data symbols exist to mitigate clock offsets and the impact of clock skew on distance estimation at the high cost of multiple channel access resulting in high channel utilization. We focus on two-way RTD to reduce the number of required channel accesses. The ranging initiator encodes its precise transmission time and a data packet identification number (ID) in one OFDM symbol called ranging data symbol, see Fig. 2. A receiver decodes the ranging data symbol and estimates the precise reception time with a maximum-likelihood (ML) estimator based on the preamble symbol. The reply-time between reception of the ranging data symbol and re-transmission at the receiver is encoded and transmitted back to the ranging initiator. The ranging initiator calculates the distance based on the transmission time stamps, the decoded reply-time, and the ML time-estimate of the received OFDM frame.

Two-way RTD ranging is sensitive to clock skew: the distance estimate is biased proportional to the reply-time and the clock skew. The clock skew is time-variant depending on the quality of the clock, e.g., temperature-compensated Quartz crystal clocks commonly become unstable for observation durations above 0.5 sec. Common strategies to mitigate this effect are very short reply times as in ultrawide bandwidth (UWB) ranging devices or very stable clocks. Both strategies pose stringent requirements on practical realizations. We propose to exploit the broadcasting nature of our Swarm-CPNT system: based on the regularly broadcasted OFDM symbols we estimate and track the clock skew with a Kalman filter. The resulting clock skew estimate is used to compensate the initially estimated RTD. Parameters for the process noise covariance in the Kalman filter are determined from Allan deviation measurements of eight software-defined radios (SDRs). Fig. 3 shows distance estimates between two SDRs connected over a radio frequency (RF) splitter with and without clock skew compensation over a duration of about 23 min. Measurements are taken simultaneously and the mean value has been removed. We clearly see the drift in distance estimate for the uncompensated case resulting in distance errors of more than one meter and a 1σ standard deviation of 53 cm. With clock skew tracking and compensation enabled we obtain a constant distance estimate over time with a 1σ standard deviation of 2.4 cm at an estimated signal-to-noise ratio (SNR) of 29.4 dB, see the blue curve in Fig. 3.

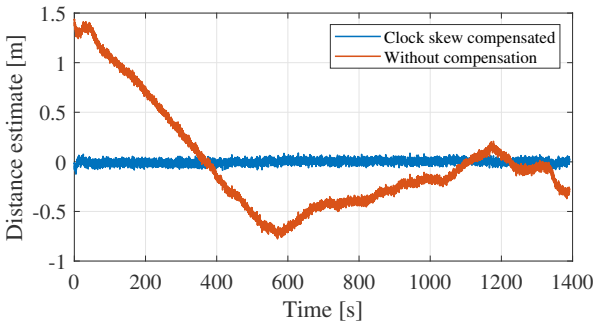


Figure 3: Distance estimates with and without clock skew tracking and compensation over a measurement duration of about 23 min.

2.3 Distributed Localization

Ad-hoc communication and precise ranging provide the basis for the third building block: distributed localization of all agents in a Bayesian framework realized as distributed particle filter [1]. Centralized localization techniques are out of scope due to SPOF of a central entity.

Most work in the field of cooperative localization, also referred to as anchor-free localization if no a-priori known anchors are available, considers localization as a two-step problem [1]: distance estimation (ranging) and location estimation (localization). Each step is optimized individually. In the first step, the distance of each inter-agent link is estimated by the receiver: commonly as single-tap ML estimator calculating the cross-correlation and finding the maximum peak of the correlation function. This maximum peak is considered as geometry line-of-sight (GLOS). In the second step, a non-linear estimator uses the distance estimates of multiple links as measurement input and solves the location equation. The error distribution of the distance estimate must be accurately determined to fuse distance estimates from multiple inter-agent links. A Gaussian error model is commonly assumed for the ranging error: based on the estimated SNR the Cramér-Rao lower bound (CRLB) or Ziv-Zakai lower bound (ZZLB) can be calculated to lower bound the variance of distance estimates. The lower bounded variance is then applied as weight for each inter-agent link. However, the coherence between ranging and localization is not fully exploited. Multipath propagation causes incorrect maximum correlation peak detection resulting in incorrect link weighting and large localization errors.

To overcome the shortcomings of the two-step localization approach the raw received signal samples are taken as measurements to derive the joint likelihood function of location estimate with single channel-tap assumption. We propose a direct signal domain particle filtering al-

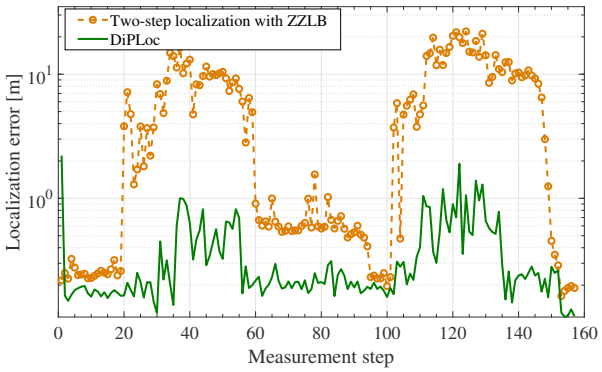
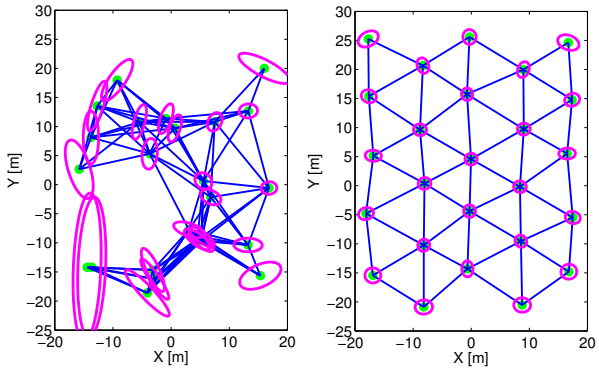


Figure 4: Swarm localization error of state-of-the-art two-step localization with ranging error modeling based on the ZZLB and the proposed DiPLoc filter. Inherent inter-agent link weighting in DiPLoc significantly reduces the localization error caused by erroneous distance estimates [2].



(a) Random formation. *(b) Quasi-lattice formation.*

Figure 5: Localization CRLB for random and quasi-lattice swarm formations [1]. Blue lines show wireless connections between agents indicated as green dots. The localization error ellipse is drawn in magenta.

gorithm for network localization (DiPLoc) based on the derived likelihood function: obtaining location information directly from received signal samples avoiding ranging model approximation [2]. The proposed DiPLoc is not a maximum a posteriori probability (MAP) estimator for a multipath scenario. However, it takes every peak of the correlation function as soft hypotheses and prevents making hard decision in the intermediate step. The high number of inter-agent links jointly support the right hypotheses and reject wrong ones with high probability. Within DiPLoc, multiple links are inherently weighted by the overall likelihood, preserving as much information as possible from the signal domain to the location domain.

We conducted an outdoor experiment with six agents to show the benefit of using DiPLoc over the two-step localization approach. One agent has been maneuvered remotely to create a dynamic measurement track. Fig. 4 shows resulting swarm localization errors for the state-of-the-art two-step localization approach and our proposed DiPLoc. The applied two-step approach lower bounds the ranging variance with the ZZLB to take correlation peak detection ambiguities caused by low SNR into account. We clearly see high localization errors resulting from multipath and incorrect inter-agent link weighting. DiPLoc preserves all information from the received signal and shows a significant gain: the localization error is reduced by a factor of 10.

3 SWARM NAVIGATION

Once agents in a swarm establish a network and are able to localize themselves, they can exploit this information to navigate and explore. Swarm navigation in the scope of this work is understood as computational data driven technique to optimize swarm movements. What makes this approach differ from that of classical single-agent navigation is the fact that due to cooperation the local information collected by the agents, e.g., location information, becomes globally available. Combining distributed swarm control and radio localization permits dynamic spatial topologies which we elaborate in three steps next.

At first, we introduce location-aware formation control. Localization performance also depends on the relative geometry (formation) of the swarm and can be altered through control of individual agents. In example, swarm agents distributed along a straight line will have an unfavorable geometric dilution of precision (GDOP) resulting in large localization errors. A distributed controller can be designed, which optimizes an objective function subject to minimizing the global localization error [1] and avoiding collisions of agents. Fig. 5 shows the resulting localization CRLB for random swarm formation, and for an optimized swarm formation taking the distributed controller into account. Random swarm formations suffer from bad GDOP and low connectivity among agents particularly at the edge of the network: the localization error ellipses become large. The distributed controller ensures optimal localization error for all agents in the swarm. As a result, localization-optimal swarm formations emerge, e.g., to automatically create a distributed sensor array with high precision location information.

In a second step, location-aware formation control can be extended: agents span a phased array and cooperatively detect and estimate the bearing of a low-frequency navigation beacon placed at the landing base [3]. Signal detection and bearing estimation are solved jointly using sparse Bayesian learning with dictionary refinement.

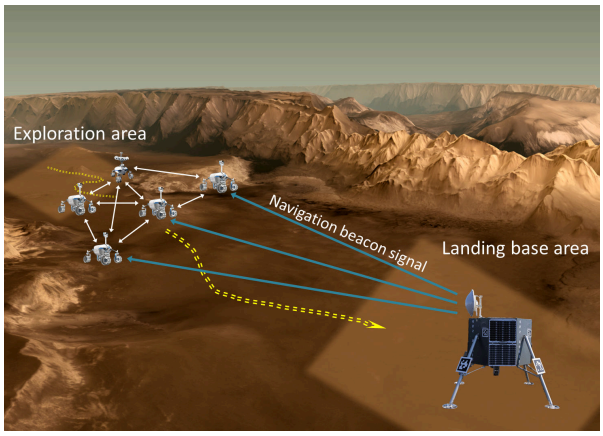


Figure 6: Swarm cooperatively spanning a phased array to determine the bearing of a radio navigation beacon from the landing base. Agents must additionally localize themselves and jointly move towards the landing base.

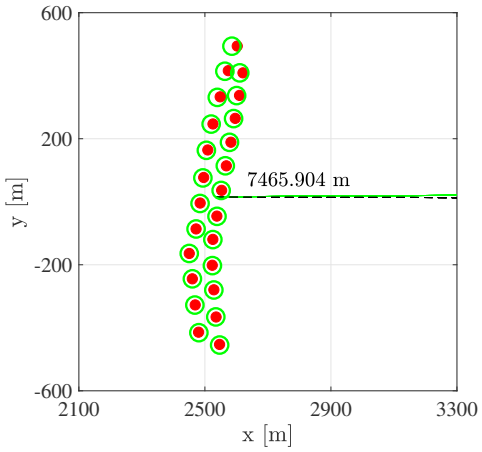


Figure 7: Result of the location information driven algorithm after some time steps. Red dots indicate the true agent positions, and green circles the estimated relative position. The navigation beacon is located on the positive x-axis and is 7465 m *away from the swarm with the dashed line indicating the true bearing. The green line overlapping the dashed one shows the estimated bearing by the swarm array [5].*

Bayesian sparsity is applied to detect the presence of the navigation beacon. Once the navigation beacon has been detected, its parameters are estimated with a gradient-based technique which uses classical average consensus for the gradient and the cost function.

The third and last step combines the previous two steps to enable a new navigation application: swarm return-to-base navigation, see Fig. 6. In swarm return-to-base application, all agents shall automatically be guided back to the very far away landing base [4]. A low-frequency radio navigation beacon at the landing base is used to cover an exploration area much larger than the achievable communication distance among agents from the Swarm-CPNT system. Agents in the swarm estimate their relative location (swarm formation), automatically span a phased array to estimate the bearing of the beacon, and move towards to landing base.

In general, swarm navigation applications are multi-objective optimization problems. For swarm return-to-base the objectives consist of three problems [5]. *Problem 1* - location information seeking: agents shall minimize the localization error. The localization error includes both, the swarm formation estimation error based on our Swarm-CPNT system, and the bearing estimation of the navigation beacon. *Problem 2* - collision avoidance: the distance between agents shall be larger than a certain threshold. *Problem 3* - return-to-base: it must be guaranteed that the swarm arrives at the landing base after a certain number of time steps.

In the following we present numerical results based on the work in [5]. Fig. 7 shows the result of the location driven algorithm after some time steps, taking all three aforementioned problems into account. For bearing estimation only, the agents would span an as large as possible linear array, but relative localization performance of agents spanning a linear array is worse. Hence, a quasi-lattice type linear array of agents results from the location information driven algorithm. Fig. 8 shows the bearing estimation error for four different algorithms tackling the multi-objective optimization problem with details found in [5]. Applying the location driven algorithm yields the lowest bearing estimation error over distance to the landing base showing the benefit of joint distributed localization and swarm control.

4 SWARM EXPLORATION

The ability of swarm agents to establish a network and navigate can be used as a key building block for swarm exploration – computational, data driven techniques which permit agents in a swarm to analyze measured data and move according to some objective data-dependent function. Specifically, agents can exploit the communication network and localization information to (i) spatially relate the sensory measurements, (ii) cooperatively process and analyze the data, and (iii) compute optimal movement strategies. What makes this approach differ from that employing a single agent is the fact that, due to cooperation, the local information collected by agents become globally available. Moreover, if the data processing is designed in an appropriate fashion, a swarm can tolerate outages of individual agents.

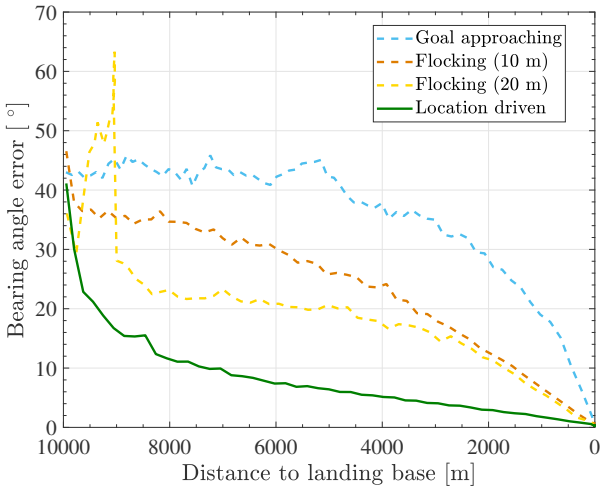


Figure 8: Bearing estimation error over distance to the landing base. The low-RF navigation beacon is located at the landing base [5].

This in turn leads to higher robustness in general with a gradual performance degradation when agents are failing.

In the following we will discuss the use of swarm exploration techniques for the task of exploring unknown static spatial processes. The discussion will address two – often complimentary – aspects of swarm exploration. First, we will consider a distributed estimation strategy that exploits a decentralized cooperative algorithm to learn a representation of a static spatial field from spatially distributed samples using kernel regression [6]. We will extend classical regression by imposing sparsity constraints on the resulting models, which will effectively permit compressing the information "on the fly. Also, this will allow relaxing storage and computational requirements at each agent. Furthermore, using the theory of optimal experiment design [7] we then propose an exploration scheme that quantifies the uncertainty of the learned model and proposes "points of interest" for each agent. Second, we will describe a distributed coordination algorithm for multi-agent exploration under complex constraints, such as obstacles in the environment or agent movement constraints. In particular, we will consider how sampling algorithms can be used for optimal (informative) path planning based on the estimated models of the spatial processes.

4.1 Splitting-over-features approach to distributed learning

We begin by consider a multi-agent system with K mobile agents. Agents are connected in a communication network such that there is a connection between any two agents in the network. In other words, we assume a network to be connected, but not necessarily fully connected, see Fig. 1 as an example swarm topology. Such connectivity requirements are needed to ensure that information collected by an agent in the network can propagate to other agents.

We will assume that each agent is making a scalar sensor measurement $y_k[n] \in \mathcal{Y} \subset \mathbb{R}$, $k = 1, \ldots, K$, e.g., gas concentration, magnitude of a magnetic field, terrain height, etc., at a 2D position $\boldsymbol{x}_k[n] \in \mathcal{X} \subset \mathbb{R}^2$, $k = 1, \ldots, K$, that is estimated using some localization system. In the following we will assume that the positions where agents take measurements are known. This implies that the localization problem has been solved using e.g., SLAM algorithms [8], or our proposed Swarm-CPNT system. Notation $[n]$, $n = 1, \ldots, N_k$, denotes a measurement sample taken by the kth agent and N_k is a total number of measurements done by the agent. Thus, each measurement consists of a pair $\{\boldsymbol{x}_k[n], y_k[n]\}$. Our goal is to reconstruct an unknown spatial function $\tilde{f} : \mathcal{X} \mapsto \mathcal{Y}$ using all collected data in a distributed fashion.

To reconstruct the unknown function $\tilde{f}$ we approximate it with a model $m : \mathcal{X} \mapsto \mathcal{Y}$ that consists of a superposition of atoms (or kernels) $\phi_k(\boldsymbol{x}'; \boldsymbol{x}) : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$, $k = 1, \ldots, K$, such that

$$\tilde{f}(\boldsymbol{x}) \approx m(\boldsymbol{x}) = \sum_{k=1}^{K} \sum_{n=1}^{N_k} w_{k,n} \phi_k(\boldsymbol{x}_k[n]; \boldsymbol{x}). \qquad (1)$$

The atoms $\phi_k(\boldsymbol{x}'; \boldsymbol{x})$ play a role of features. Their form might vary depending on the application; typically, they are selected to reflect spatial correlations in the estimated field in the vicinity of the measurement location $\boldsymbol{x}'$. Radial basis functions [9] exemplify well possible atom choices: they change monotonically with distance from some central point, which in our case represents a measurement location. Also note that within the splitting-over-features approach the features and the corresponding weights $w_{k,n}$ are different for each agent. In other words, each agent constructs its own model of the environment; however, the agents in the swarm share their measured data.

In this work we explicitly assume that some of the weights $w_{k,n}$ are zero. This assumption will reflect the fact that not all features are relevant for representing the measured data. This is realized by imposing sparsity constraints on the weights, as will be explained in the following. Given measurements $\{\boldsymbol{x}_k[n], y_k[n]\}_{n=1}^{N_k}$, $k = 1, \ldots, K$, the learning problem then aims to estimate distributed coefficients $\boldsymbol{w}_k = [w_{k,1}, \ldots, w_{k,N_K}]^T$, $k = 1, \ldots, K$, under assumption of sparsity.

To be able to formulate the corresponding optimization problem, we cast (1) in a more convenient matrix form.

To this end we define a vector function

$$\begin{aligned}\boldsymbol{\phi}_{k,i} = [&\phi_k(\boldsymbol{x}_k[i]; \boldsymbol{x}_1[1]), \ldots, \phi_k(\boldsymbol{x}_1[N_1], \boldsymbol{x}_k[i]),\\ &\phi_k(\boldsymbol{x}_2[1], \boldsymbol{x}_k[i]), \ldots, \phi_k(\boldsymbol{x}_2[N_2], \boldsymbol{x}_k[i]), \ldots,\\ &\phi_k(\boldsymbol{x}_K[1], \boldsymbol{x}_k[i]), \ldots, \phi_k(\boldsymbol{x}_K[N_K], \boldsymbol{x}_k[i])]^T,\end{aligned}$$

which represents an ith feature vector of the kth agent, evaluated at all available measurement locations. Now, let us define a matrix $\boldsymbol{\Phi}_k = [\boldsymbol{\phi}_{k,1}, \ldots, \boldsymbol{\phi}_{k,N_k}]$, which collects all kernels associated with the kth agent. Likewise, we collect the sensor measurements of the kth agent as $\boldsymbol{y}_k = [y_k[1], \ldots, y_k[N_k]]^T$. All sensor measurements collected by the swarm are represented with a vector $\boldsymbol{y} = [\boldsymbol{y}_1^T, \ldots, \boldsymbol{y}_K^T]^T$. Finally, following (1), the model of measured sensor data can be represented as

$$\boldsymbol{y} = \sum_{k=1}^{K} \boldsymbol{\Phi}_k \boldsymbol{w}_k + \boldsymbol{\xi} = \boldsymbol{\Phi}\boldsymbol{w} + \boldsymbol{\xi}, \tag{2}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_K]$ and $\boldsymbol{w} = [\boldsymbol{w}_1^T, \ldots, \boldsymbol{w}_K^T]^T$ are aggregated atoms and parameter vector, respectively, and $\boldsymbol{\xi}$ is an additive zero-mean Gaussian measurement noise. For simplicity we will assume that the noise $\boldsymbol{\xi}$ is a zero mean Gaussian process with a known precision matrix $\boldsymbol{\Lambda}$. Note that while features, and thus agent models $\boldsymbol{\Phi}_k\boldsymbol{w}_k$, are unique for each agent, the measurements are shared. As such, a collaborative response (1) should approximate the whole measurement data as closely as possible. Also, due to possible correlations between the features of different agents, the decision on relevance of the features should likewise be done in a cooperative fashion.

One of the key challenges associated with the considered estimation problem is the fact that unless agent predictions $\boldsymbol{\Phi}_k\boldsymbol{w}_k$ are orthogonal, the models parameters $\boldsymbol{w}_k$, $k = 1, \ldots, K$, will be correlated. This implies that agents have to take measurements with a sufficient separation between the sampling points to avoid overlapping of the corresponding kernels – a complex scheduling and planning task. This incurs a rank deficiency of $\boldsymbol{\Phi}$, and thus the estimator requires an appropriate regularization. A regularization can be incorporated through sparsity constraints on $\boldsymbol{w}$. In particular, we introduce sparsity using sparse Bayesian learning (SBL) techniques [10, 11], as explained in the following.

4.1.1 Sparse Bayesian learning

Using the model (2) we can define a likelihood function $p(\boldsymbol{y}|\boldsymbol{w})$ of model parameters $\boldsymbol{w}$ as

$$p(\boldsymbol{y}|\boldsymbol{w}) \propto \mathrm{e}^{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{w})^T\boldsymbol{\Lambda}(\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{w})}. \tag{3}$$

Classical maximum likelihood approach to estimate $\boldsymbol{w}$ involves maximizing (3); yet this approach is prone to overfitting and might require numerical stabilization if $\boldsymbol{\Phi}$ does not have a full column rank. Also the resulting estimate of $\boldsymbol{w}$ is generally not sparse. In SBL [10, 11] the weights $\boldsymbol{w}$ are additionally constrained using a parametric prior $p(\boldsymbol{w}|\boldsymbol{\gamma}) = \prod_{k=1}^{K}\prod_{l\in\mathcal{I}_k} p(w_{k,l}|\gamma_{k,l})$, where $p(w_{k,l}|\gamma_{k,l}) = \mathrm{N}(0, \gamma_{k,l})$ is a Gaussian pdf with zero mean and variance $\gamma_{k,l}$; the latter are also treated as unknown model parameters. SBL then seeks an estimate of $\widehat{\boldsymbol{w}}$ and $\widehat{\boldsymbol{\gamma}}$ as a solution to the following optimization problem

$$\widehat{\boldsymbol{w}}, \widehat{\boldsymbol{\gamma}} = \underset{\boldsymbol{w},\boldsymbol{\gamma}}{\operatorname{argmax}}\, p(\boldsymbol{w}, \boldsymbol{\gamma}|\boldsymbol{y}) = \underset{\boldsymbol{w},\boldsymbol{\gamma}}{\operatorname{argmax}}\, p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\gamma})p(\boldsymbol{\gamma}|\boldsymbol{y}). \tag{4}$$

In (4) the pdf $p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\gamma}) \propto p(\boldsymbol{y}|\boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{\gamma})$ – the posterior pdf of the parameters $\boldsymbol{w}$ – can be easily determined to be a Gaussian pdf, with the mean $\widehat{\boldsymbol{w}}$ and covariance matrix $\widehat{\boldsymbol{\Omega}}$ given as

$$\widehat{\boldsymbol{\Omega}} = \left(\boldsymbol{\Phi}^T\boldsymbol{\Lambda}\boldsymbol{\Phi} + \boldsymbol{\Gamma}^{-1}\right)^{-1}, \quad \widehat{\boldsymbol{w}} = \widehat{\boldsymbol{\Omega}}\boldsymbol{\Phi}^T\boldsymbol{\Lambda}\boldsymbol{y}, \tag{5}$$

where $\boldsymbol{\Gamma} = \mathrm{diag}(\boldsymbol{\gamma})$ is a diagonal matrix with sparsity parameters $\boldsymbol{\gamma}$ on the main diagonal. Expression (5) can be recognized as a linear minimum mean squared error estimator of the weights $\boldsymbol{w}$, conditioned on sparsity parameters $\boldsymbol{\gamma}$. Note that $\boldsymbol{\gamma}$ act explicitly as regularization coefficients.

The second pdf $p(\boldsymbol{\gamma}|\boldsymbol{y})$ in (4) can be computed as follows

$$\begin{aligned}p(\boldsymbol{\gamma}|\boldsymbol{y}) \propto p(\boldsymbol{\gamma})p(\boldsymbol{y}|\boldsymbol{\gamma}) &= p(\boldsymbol{\gamma})\int p(\boldsymbol{y}|\boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{\gamma})\mathrm{d}\boldsymbol{w}\\ &= p(\boldsymbol{\gamma})|\boldsymbol{\Sigma}|^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{y}},\end{aligned} \tag{6}$$

where $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}^{-1} + \boldsymbol{\Phi}\boldsymbol{\Gamma}\boldsymbol{\Phi}^T$. It is important to note for the following discussion that for solving (4) it suffices to find an estimate of $\widehat{\boldsymbol{\gamma}}$ that maximizes (6) (see [11] for more details).

Now we are ready to discuss a distributed approach to implementing the splitting-over-features approach with SBL. To this end we discuss two distributed algorithms: a distributed Expectation-Maximization (EM) based algorithm, and algorithm that uses alternating directions method of multipliers (ADMM) to estimate the parameters of interest $\boldsymbol{w}$ and $\boldsymbol{\gamma}$.

4.1.2 Distributed SBL with Expectation-Maximization

Consider the posterior pdf $p(\boldsymbol{w}, \boldsymbol{\gamma}|\boldsymbol{y})$. From Bayes theorem, $p(\boldsymbol{w}, \boldsymbol{\gamma}|\boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{w})p(\boldsymbol{w}, \boldsymbol{\gamma})$, where we used the fact that $\boldsymbol{y}$ is conditionally independent of $\boldsymbol{\gamma}$ given $\boldsymbol{w}$. Here our goal is to maximize $p(\boldsymbol{y}|\boldsymbol{w})p(\boldsymbol{w}, \boldsymbol{\gamma})$ with respect to $\boldsymbol{\gamma}$ and $\boldsymbol{w}$ using the expectation-maximization approach [12, 13].

To this end we introduce a latent variable $\boldsymbol{h} = [\boldsymbol{h}_1^T, \ldots, \boldsymbol{h}_K^T]^T$, with sub-vectors $\boldsymbol{h}_k$, $k = 1, \ldots, K$, defined as

$$\boldsymbol{h}_k = \boldsymbol{\Phi}_k \boldsymbol{w}_k + \boldsymbol{\xi}_k, \; k = 1, \ldots, K. \tag{7}$$

These can be conceived as noisy "prediction" of K agents. The additive noise component $\boldsymbol{\xi}_k$ in (7) is obtained by arbitrarily decomposing the total noise $\boldsymbol{\xi}$ in (2) into K independent contributions, such that $E\{\boldsymbol{\xi}_k \boldsymbol{\xi}_l^T\} = 0$ for $k \neq l$, $E\{\boldsymbol{\xi}_k \boldsymbol{\xi}_k^H\} = \boldsymbol{\Lambda}_k^{-1} = \beta_k \boldsymbol{\Lambda}^{-1}$ for $0 \leq \beta_k \leq 1$, and $\sum_{k=1}^K \beta_k = 1$. The relationship between $\boldsymbol{y}$, $\boldsymbol{h}$ then becomes

$$\boldsymbol{y} = \sum_{k=1}^K \boldsymbol{h}_k = [\underbrace{\boldsymbol{I}, \ldots, \boldsymbol{I}}_{K \text{ times}}]\boldsymbol{h} = \boldsymbol{I}_K \boldsymbol{h}, \tag{8}$$

Within the EM algorithm the maximization of the "incomplete" posterior $p(\boldsymbol{w}, \boldsymbol{\gamma}|\boldsymbol{y})$ is done via the maximization of the corresponding complete posterior

$$p(\boldsymbol{h}, \boldsymbol{w}, \boldsymbol{\gamma}|\boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{h}) p(\boldsymbol{h}|\boldsymbol{w}) p(\boldsymbol{w}, \boldsymbol{\gamma}) \tag{9}$$

Applying the EM algorithm requires computing two optimization steps: E-step and M-step [13, 12]. The E-step of the algorithm aims at estimating the complete data and in our cases it can be computed in closed form as

$$\widehat{\boldsymbol{h}}_k = \boldsymbol{\Phi}_k \widehat{\boldsymbol{w}}_k + \beta_k \left(\boldsymbol{y} - \sum_{k=1}^K \boldsymbol{\Phi}_k \widehat{\boldsymbol{w}}_k \right), k = 1, \ldots, K. \tag{10}$$

The M-step requires solving the following optimization

$$\{\widehat{\boldsymbol{w}}', \widehat{\boldsymbol{\gamma}}'\} = \underset{\boldsymbol{w}, \boldsymbol{\gamma}}{\operatorname{argmax}} \sum_{k=1}^K \log p(\boldsymbol{w}_k, \boldsymbol{\gamma}_k) - \frac{1}{2} \|\widehat{\boldsymbol{h}}_k - \boldsymbol{\Phi}_k \boldsymbol{w}_k\|_{\boldsymbol{\Lambda}_l}^2 \tag{11}$$

It can be seen that the use of complete data decouples the optimization (11) in K parallel optimization with respect to the parameters of each agent. The interdependency between the agents is resolved through the complete data $\boldsymbol{h}$. As such, to implement the algorithm in the distributed fashion $\widehat{\boldsymbol{h}}$ in (10) should be computed distributively.

Distributed computation of the E-step.
Consider an agent k and the corresponding estimate $\widehat{\boldsymbol{h}}_k$. From (10) we see that $\widehat{\boldsymbol{h}}_k$ is essentially a combination of a current "local" response $\boldsymbol{\Phi}_k \widehat{\boldsymbol{w}}_k$ and a scaled residual signal $\boldsymbol{y} - \sum_{k=1}^K \boldsymbol{\Phi}_k \widehat{\boldsymbol{w}}_k$; it is the latter term that requires the participation of all agents. We note that

$$\sum_{k=1}^K \boldsymbol{\Phi}_k \widehat{\boldsymbol{w}}_k = K \left[\frac{1}{K} \sum_{k=1}^K \boldsymbol{\Phi}_k \widehat{\boldsymbol{w}}_k \right] = K \overline{\boldsymbol{\Phi} \widehat{\boldsymbol{w}}},$$

where $\overline{\boldsymbol{\Phi} \widehat{\boldsymbol{w}}}$ is an averaged response of all agents. The averaging operation can be computed over a network in an asynchronous fashion using classical averaged consensus type algorithms (see e.g., [14, 15, 16]). The latter perform averaging of information between direct one-hop neighbors in the network. Thus, (10) can be computed as

$$\widehat{\boldsymbol{h}}_k = \boldsymbol{\Phi}_k \widehat{\boldsymbol{w}}_k + \beta_k \left(\boldsymbol{y} - K \overline{\boldsymbol{\Phi} \widehat{\boldsymbol{w}}} \right), k = 1, \ldots, K. \tag{12}$$

by each agent individually once the consensus over $\overline{\boldsymbol{\Phi} \widehat{\boldsymbol{w}}}$ is achieved.

Algorithm summary
The algorithm begins with updating the current measurement set $\{\boldsymbol{x}_k[n], y_k[n]\}_{n=1}^{N_k}$, $k = 1, \ldots, K$, so that the agents can re-compute a vector $\boldsymbol{y}$ and matrices $\boldsymbol{\Phi}_k$ locally. Then, an initialization of parameters $\boldsymbol{w}_k$ and $\boldsymbol{\gamma}_k$, $k = 1, \ldots, K$ is performed. This can be realized as summarized in Algorithm (1). After initialization, two main steps of the algorithm follow: the consensus step to compute $\overline{\boldsymbol{\Phi}_k \widehat{\boldsymbol{w}}_k}$ and the complete data, and a local optimization at the M-step to estimate $\boldsymbol{w}_k$ and $\boldsymbol{\gamma}_k$ given an estimate of complete date $\widehat{\boldsymbol{h}}_k$.

Algorithm 1 Estimation of parameters

1: Initialize $\widehat{\boldsymbol{w}}_k^{[\text{init}]} \leftarrow \left(\boldsymbol{\Phi}_k^T \boldsymbol{\Lambda} \boldsymbol{\Phi}_k \right)^\dagger \boldsymbol{\Phi}_k^T \boldsymbol{\Lambda} \boldsymbol{y}$.
2: Compute $\overline{\boldsymbol{\Phi}_k \widehat{\boldsymbol{w}}_k^{[\text{init}]}}$ using averaged consensus
3: $\widehat{\boldsymbol{h}}_k^{[\text{init}]} \leftarrow$ eq. (12), $\widehat{\boldsymbol{w}}_k^{[\text{init}]} \leftarrow \left(\boldsymbol{\Phi}_k^T \boldsymbol{\Lambda}_k \boldsymbol{\Phi}_k \right)^\dagger \boldsymbol{\Phi}_k^T \boldsymbol{\Lambda}_k \widehat{\boldsymbol{h}}_k$
4: $\widehat{\gamma}_{k,l}^{[\text{init}]} \leftarrow |\widehat{w}_{k,l}^{[\text{init}]}|^2 + \left[\left(\boldsymbol{\Phi}_k^T \boldsymbol{\Lambda}_k \boldsymbol{\Phi}_k \right)^\dagger \right]_{ll}$

Let us also mention that the M-step of the algorithm can also be solved efficiently using any SBL algorithm of choice, e.g., [17, 18, 19].

4.1.3 Distributed SBL using Alternating directions method of multipliers

Another technique that can be used for SBL in a distributed setting is an ADMM algorithm. The corresponding approach to distributed SBL has been discussed in details in [20]; here we only give a summary of the key steps of the algorithm.

Likewise here our goal is to maximize (4) with respect to $\boldsymbol{w}$ and $\boldsymbol{\gamma}$ using a network of agents. Since weights $\boldsymbol{w}$ can be estimated from (5) given $\boldsymbol{\gamma}$, we look into distributed estimation of the latter. To this end, we consider the cost function $\mathcal{L}(\boldsymbol{\gamma}) = -2 \log p(\boldsymbol{y}|\boldsymbol{\gamma})$ in (6) and assume $p(\boldsymbol{\gamma}) \propto$ const. The function $\mathcal{L}(\boldsymbol{\gamma})$ can be upper bounded [18] as $\mathcal{L}(\boldsymbol{\gamma}) \leq \boldsymbol{z}^T \boldsymbol{\gamma} - g^*(\boldsymbol{z}) + \boldsymbol{y}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{y} = \mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{z})$, where $g^*(\boldsymbol{z})$ is the concave conjugate of $\log |\boldsymbol{\Sigma}|$ defined by the duality relationship $g^*(\boldsymbol{z}) = \min_{\boldsymbol{\gamma}} \boldsymbol{z}^T \boldsymbol{\gamma} - \log |\boldsymbol{\Sigma}|$ [21].

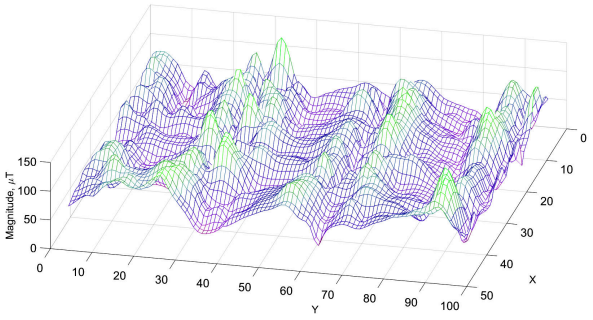


Figure 9: Measured magnetic field data.

Then, $\mathcal{L}(\boldsymbol{\gamma})$ can be minimized indirectly (and more conveniently) via $\mathcal{L}(\boldsymbol{\gamma}, z)$, which is also jointly convex in z and $\boldsymbol{\gamma}$ [18, Lemma 1]. Now, for z fixed at some value $\widehat{z}^{[i]}$ at the i-th iteration of the algorithm, the bound $\mathcal{L}(\boldsymbol{\gamma}; \widehat{z}^{[i]})$ can also be upper bounded as follows [18, Lemma 2]:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\gamma}; \widehat{z}^{[i]}) \leq & \boldsymbol{\gamma}^T \widehat{z}^{[i]} + \boldsymbol{w}^T \boldsymbol{\Gamma}^{-1} \boldsymbol{w} \\ & + \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{w}\|^2_{\boldsymbol{\Lambda}} = \mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{w}; \widehat{z}^{[i]})\end{aligned} \tag{13}$$

which is jointly convex in both $\boldsymbol{\gamma}$ and $\boldsymbol{w}$. Due to this, $\mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{w}; \widehat{z}^{[i]})$ can be "tightened" by interchangeably minimizing it with respect to $\boldsymbol{\gamma}$ and $\boldsymbol{w}$. For any $\boldsymbol{w}$, the bound $\mathcal{L}(\boldsymbol{\gamma}; \boldsymbol{w}; \widehat{z}^{[i]})$ is minimized at value

$$\widehat{\boldsymbol{\gamma}} = \underset{\boldsymbol{\gamma}}{\operatorname{argmin}}\, \mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{w}; \widehat{z}^{[i]}) = \left[\frac{|w_{\cdot,1}|}{\sqrt{z^{[i]}_{\cdot,1}}}, \ldots, \frac{|w_{\cdot,|\mathcal{I}|}|}{\sqrt{z^{[i]}_{|\mathcal{I}|}}} \right]. \tag{14}$$

By inserting the (14) in (13), $\mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{w}; \widehat{z}^{[i]})$ can then be made tight by finding $\boldsymbol{w}$ as a solution to the following optimization problem:

$$\begin{aligned}\widehat{\boldsymbol{w}} &= \underset{\boldsymbol{w}}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{w}\|^2_{\boldsymbol{\Lambda}} + 2 \sum_{m \in \mathcal{I}} \sqrt{z^{[i]}_{\cdot,m}} |w_{\cdot,m}| \\ &= \underset{\boldsymbol{w}}{\operatorname{argmin}} \left\| \boldsymbol{y} - \sum_{k=1}^{K} \boldsymbol{\Phi}_k \boldsymbol{w}_k \right\|^2_{\boldsymbol{\Lambda}} + 2 \sum_k \sum_l \sqrt{z^{[i]}_{k,l}} |w_{k,l}|\end{aligned} \tag{15}$$

The form of the objective function in (15) is also known as the LASSO problem and there have een multiplke approach proposed to solve LASSO optimization in a distributed fashion. ADMM algorithm is one of the most popular solutions; its application to (15) is rather straightforward. In [20] the interested reader will find more details on the ADMM application to SBL.

4.1.4 Exploration strategy

The estimated model of the process can naturally be used to design objective criteria for guiding agents to more informative sampling locations, i.e., explore the process. In other words, we intend to construct the agent trajectory that would improve the inference model parameters. The exploration strategy should (i) keep the number of measurements needed to estimate the parameters low, (ii) obtain parameter estimates with high certainty (high precision). In order to decide how an agent in a swarm should move, mobile agents need a metric that ranks different possibilities.

There are different approaches to defining such a metric for exploration. An often used approach is an uncertainty-driven exploration where the uncertainty (typically variance) of the parameters of interest is used to guide the agents. For linear models the corresponding uncertainty (or error bars) can be computed using standard results (see e.g., [6, 22, 23, 24, 25]). A similar approach we propose to use here. Specifically, we employ use a theory of optimal experiment [7] design and compute the next movement position so as to reduce the posterior uncertainty of the model parameters.

Consider a potential measurement position $\boldsymbol{x}^* \in \mathcal{X}$. If an agent were to make a measurement at this position, the posterior covariance matrix $\boldsymbol{\Omega}^*$ of the new weights can then be computed as

$$\boldsymbol{\Omega}^*(\boldsymbol{x}^*) = \left(\begin{pmatrix} \boldsymbol{\Phi} & \boldsymbol{\phi}^*_k \\ \boldsymbol{\phi}^*_k & 1 \end{pmatrix}^T \boldsymbol{\Lambda} \begin{pmatrix} \boldsymbol{\Phi} & \boldsymbol{\phi}^*_k \\ \boldsymbol{\phi}^*_k & 1 \end{pmatrix} + \begin{pmatrix} \boldsymbol{\Gamma} & \\ & 0 \end{pmatrix} \right)^{-1}$$

where ϕ^*_k is a "to-be-added" feature of the kth agent evaluated at a test point $\boldsymbol{x}^*$. Optimal experiment design then aims to find such a measurement location $\boldsymbol{x}^*$ that minimizes the size of the corresponding covariance matrix $\boldsymbol{\Omega}^*(\boldsymbol{x}^*)$. One possible choice for the size of $\boldsymbol{\Omega}^*(\boldsymbol{x}^*)$ is its determinant, which is also known as a D-criterion [7]. Specifically, we search for an optimal measurement location $\widehat{\boldsymbol{x}}^*$ as a solution to the following optimization problem:

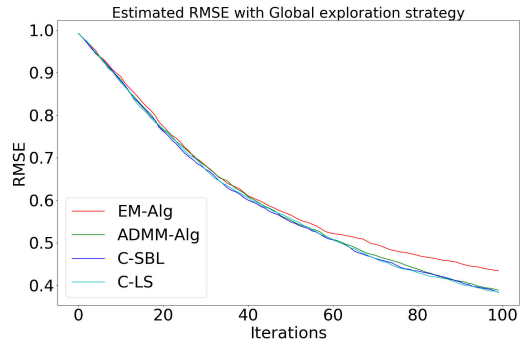
$$\widehat{\boldsymbol{x}}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \log |\boldsymbol{\Omega}^*(\boldsymbol{x})| \tag{16}$$

Using the properties of the matrix determinant the above expression can also be computed in a distributed fashion. We skip the computational details here, yet mention that evaluation of (16) requires several additional averaged consensus iterations per test point.

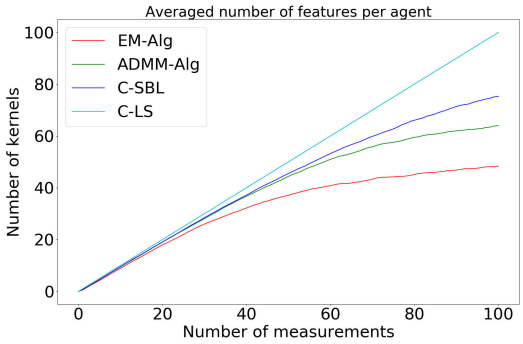
4.1.5 Experimental results

For evaluating the algorithm we are using a static map of an indoor magnetic field with a spatial dimension approx. 9.6 m × 4.3 m. The data has been collected with a spatial resolution of about 10cm. Note that this results in a total of roughly 3200 possible measurement points. The collected data set is visualized in Fig. 9.

We will consider a swarm consisting of $K = 4$ agents, with each agent making 100 measurements. In the first



(a)



(b)

Figure 11: (a) Estimated RMSE and (b) the averaged number of kernels per agent.

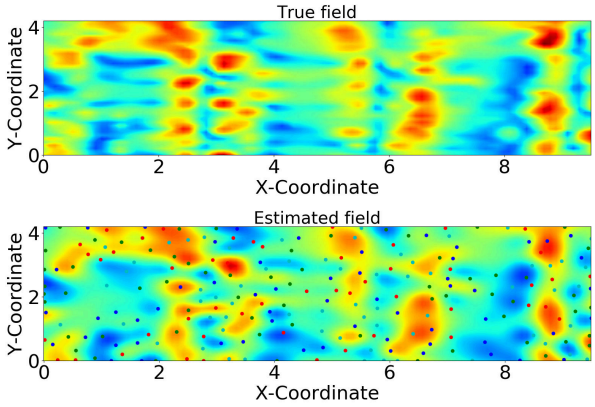


Figure 10: Ground truth and reconstructed magnetic field after 400 *measurements. The dots in the lower plot represent the measurement locations.*

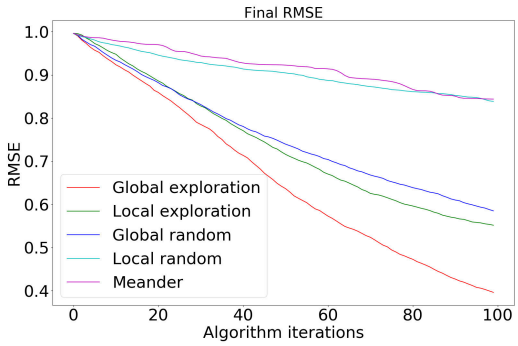
experiment we will compare several estimation algorithms: the EM algorithm (EM-Alg), the ADMM-based distributed estimation described in [20] (ADMM-Alg), the centralized version of SBL algorithm (C-SBL), and a centralized ridge regression solution (C-LS). As an exploration strategy we will use the optimization (16) computed for $\boldsymbol{x} \in \mathcal{X}$, i.e., we will consider the whole exploration domain $\mathcal{X}$ for possible new measurement locations. Such optimization approach we will term *Global Eploration*.

As a performance criterion we will compute the normalized RMSE value between the estimated magnetic field and the ground truth process and the averaged number of features retained in the algorithm per agent. First, in Fig.12b we show the results for the reconstructed the magnetic field after a single algorithm run. As you can see, after 400 measurements, the algorithm was able to learn the key features of the explored process.

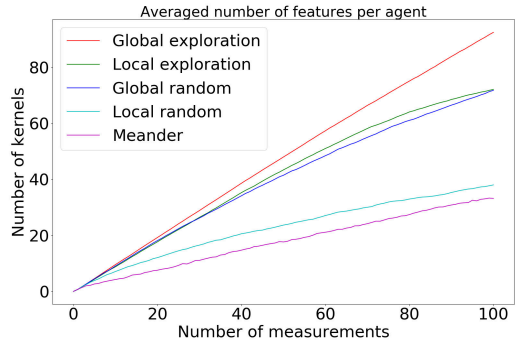
The next plots in Fig.11 show the averaged performance of the algorithm after 10 independent Monte Carlo runs. Specifically, the evolution of the RMSE as a function of number of measurements and the resulting number of kernels are shown. As we can see, the performance of the methods in terms of RMSE is quite compatible, however there are some differences in the number of the estimated kernels. In particular, we see that the resulting model complexity varies: C-LS algorithm does not profit from sparsity at all, i.e., all 100 measurement positions are retained in the model. On the other hand, the EM-Alg, ADMM-Alg and C-SBL remove irrelevant information. In case of ADMM-Alg almost 40% of the irrelevant measurement points have been removed without any reduction of the RMSE as compared to other schemes. EM-Alg removes the points more aggressively (roughly 60% of the measurements are deemed as irrelevant), though at the expense of slightly reduced RMSE.

Next we study the performance of the distributed estimation algorithm with different exploration strategies. To this end we consider only the ADMM algorithm and consider different criteria for selecting the next measurement point. Specifically, in addition to the *Global exploration* we will consider the strategy where (16) is solved in a small vicinity of the current agent position (within approx. 0.5m radius); to this strategy we will refer as *Local Exploration*. Additionally we will also consider random movement of agents over the whole exploration domain $\mathcal{X}$ (*Global Random* strategy), random movements within a radius of 0.5m from the current agent position (*Local Random* strategy), and a systematic *Meander* scanning strategy. The corresponding results are summarized in Fig.12.

As we can see, local and meander strategies exhibit a rather poor information gathering performance. Global random approach does perform better, but still cannot beat the intelligent exploration strategies employing (16). This is due to the fact that the information is collected in a more efficient way. On the other hand, we can also see that reducing the complexity of computing (16) by restricting the evaluation to a certain neighbour-



(a)



(b)

Figure 12: (a) Estimated RMSE and (b) the averaged number of retained kernels per agent for different exploration strategies.

hood of the current position, also known as myopic or greedy approaches, significantly reduce the efficiency of the exploration.

4.2 A Distributed Coordination Algorithm for Multi-Robot Exploration under Complex Constraints

Robots employed in robotic exploration missions have complex dynamics, as robots must adapt their motion to drive on complex terrains. However, most state-of-the-art multi-agent exploration algorithms do not consider robots dynamics (see [26] for an overview). Moreover, most algorithms do not take into account complex inter-agent constraints like e.g. inter-robot collision avoidance or inter-robot communication constraints. The ability to handle collision avoidance and communication constraints is crucial to develop a multi-agent cooperative system.

In this section, we introduce an approach that tackles the two aforementioned issues: handling complex robot dynamics, and inter-agent complex constraints. To this end we propose an algorithm that combines (i) Gaussian processes (GPs) [27] to model a physical process of interest, (ii) a sampling-based planner (RRT) [28] to plan paths that consider robots dynamics, and (iii) a distributed decision-making algorithm (max-sum) [29] to achieve multi-robot coordination.

The combination of the three aforementioned elements allows agents to perform an efficient exploration, subject to inter-agent collision avoidance and communication constraints. Next we present in Sec. 4.2.1 an overview of our sampling-based multi-robot exploration (SBMRE) algorithm [26]. This is followed in Sec. 4.2.2 by an evaluation of the proposed approach in simulations, and in a field experiment where three quadcopters explore a simulated wind field.

4.2.1 SBMRE Algorithm Overview

We depict in Figure 13 a block diagram that describes the algorithm's execution. SBMRE algorithm is executed locally by each robot, and works sequentially, where each full iteration solves the next finite-horizon exploration task. The algorithm works as follows: first, each of the agent individually grows an RRT with its current position as root. Then agents send the set of paths contained in the tree to their neighbors in order to cooperate about the optimal assignment of paths. Since the number of paths in the tree could grow indefinitely, we propose a clustering method that reduces the computational complexity of the cooperation procedure by clustering the set of paths of each of the robots. Next agents send these clusters to their neighbors.

Once an agent receives the clusters, it starts executing max-sum to select the cluster that maximizes a user-defined global utility function that is subject to inter-robot constraints. Here we define this utility function as an information-theoretic function – mutual information (MI) – calculated from the underlying GPs model.

Once agents select their own cluster, they communicate the selected cluster to their neighbors. Then, each agent selects a path within its cluster by evaluating a MI between their current process estimation and their future potential paths.

Next agents traverse the selected paths, and exchange the gathered measurements through the network via a flooding mechanism. These measurements are then employed by the agents to update its GPs model. This loop is repeated till a user-defined stopping criterion, e.g. exploration time or remaining uncertainty about the process, is fulfilled.

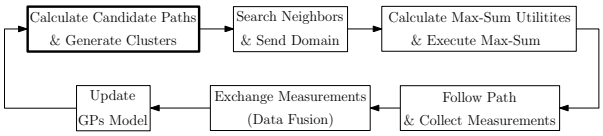
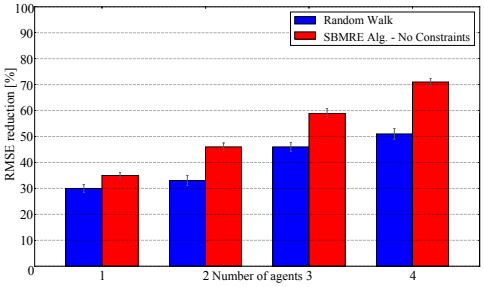
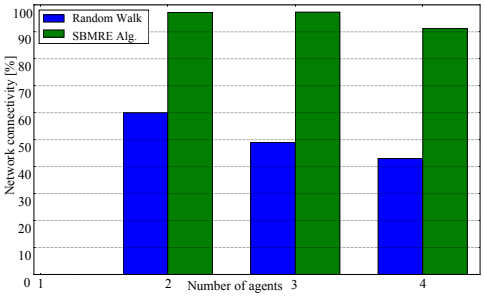


Figure 13: SBMRE algorithm block diagram.



(a) RMSE reduction.



(b) Network connectivity.

Figure 14: SBMRE algorithm's performance. (a)RMSE reduction; (b)Network connectivity.

4.2.2 SBMRE Algorithm Evaluation

We evaluate our SBMRE algorithm in simulations, where a system composed by up to 4 agents explores a wind field. In particular, we evaluate two metrics. First, we evaluate the reduction of the root mean squared error (RMSE) between the initial process estimation and the estimation that results after a 300 s exploration run. Then, we evaluate the ability of robots to maintain network connectivity. Note that our goal is to achieve a 100% in both RMSE reduction and network connectivity. We compare our SBMRE algorithm against a random walk. Results of this evaluation are depicted in Fig. 14. Results demonstrate that SBMRE algorithm clearly outperforms a random walk. Moreover, results indicate that SBMRE algorithm greatly benefits from multi-agent coordination.

In addition to simulations, we carried out a field experiment with 3 quadcopters. Experimental results indicate that we achieve a 300% RMSE improvement with three robots respect to one, which indicates an efficient multi-agent coordination.

5 CONCLUSION AND FUTURE WORK

Several algorithms for distributed sparse Bayesian learning with spliting-over-features approach to data distribution have been discussed. In the proposed algorithm a multi-agent system is used to collaboratively learn a stationary spatial process based on measurements performed by individual agents. Such algorithms can be useful when a swarm is needed to collaboratively and efficiently map a static spatial process. The obtained results demonstrate that the use of model-based exploration strategies, and in particular strategies that aim to reduce the uncertainty of the estimated parameters, perform more efficiently as compared to the systematic scanning techniques. We also discussed a strategy for planing agent movements in more realistic scenarios, i.e., in situation when the environment is populated with obstacles or realistic agent dynamics is taken into account. From practical perspective, both realistic path planing and distributed estimation algorithms have to be combined. In particular, the proposal way points generated by the exploration algorithm should be used as an input for SBMRE algorithm that will then produce realistic movement trajectories. The investigations of such combined approaches is currently underway.

We are currently realizing the Swarm-CPNT concept with software-defined radios for real-world experimentation. The Swarm-CPNT concept is currently extended with antenna arrays for each agent to enable orientation estimation jointly with localization and time synchronization. Swarm navigation does not consider the environment yet, e.g., reachable positions to span an array or obstacles. Swarm exploration and swarm navigation are not yet connected. Hence, future work will focus on the fundamentals how to solve these class of optimization problems efficiently and how resulting algorithms can be realized for individual agents.

Acknowledgement

This work was partially supported by the DLR project *Navigation 4.0* and the project *VaMEx-CoSMiC* supported by the Federal Ministry for Economic Affairs and Energy on the basis of a decision by the German Bundestag, grant 50NA1521 administered by DLR Space Administration.

References

[1] Zhang S, Staudinger E, Sand S, Raulefs R and Dammann A (2014) Anchor-Free Localization using Round-Trip Delay Measurements for Martian Swarm Exploration. In: *Proceedings of IEEE ION PLANS*, Monterey, California, USA.

[2] Zhang S, Staudinger E, Wang W, Gentner C, Dammann A and Sandgren E (2015) DiPLoc: Direct Signal Domain Particle Filtering for Network Localization. In: *Proceedings of ION GNSS+*, ION, Tampa, Florida, USA.

[3] Shutin D and Zhang S (2016) Distributed Sparsity-Based Bearing Estimation with a Swarm of Cooperative Agents. In: *2016 IEEE Global Conference on Signal and Information Processing*, IEEE. URL `http://elib.dlr.de/106292/`.

[4] Zhang S, Froehle M, Wymeersch H, Dammann A and Raulefs R (2015) Location-Aware Formation Control in Swarm Navigation. In: *Globecom Workshop on Wireless Networking, Control and Positioning for Unmanned Autonomous Vehicles (WiUAV), 2015*, IEEE, San-Diego, California, USA.

[5] Zhang S, Raulefs R and Dammann A (2016) Localization-Driven Formation Control for Swarm Return-to-Base Applicationn. In: *European Signal Processing Conference, 2016*, IEEE, EURASIP, Budapest, Hungary.

[6] Bishop CM (2006) *Pattern Recognition and Machine Learning*. Springer, New York. ISBN 0387310738.

[7] Fedorov VV (1972) *Theory of Optimal Experiments*. Academic Press.

[8] Thrun S, Burgard W and Fox D (2005) *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press.

[9] Orr MJL (1996) *Introduction to radial basis function networks*. Technical report, Centre For Cognitive Science.

[10] Tipping M (2001) Sparse Bayesian Learning and The Relevance Vector Machine. In: *J. Mach. Learn. Res.*, 1:pp.211–244.

[11] Wipf D and Rao B (2004) Sparse Bayesian Learning for Basis Selection. In: *IEEE Trans. Signal Process.*, 52(8):pp.2153–2164.

[12] Feder M and Weinstein E (1988) Parameter estimation of superimposed signals using the EM algorithm. In: *IEEE Trans. Acoust. Speech Signal Process.*, 36(4):pp.477–489.

[13] Dempster A, Laird N, Rubin D and Others (1977) Maximum likelihood from incomplete data via the EM algorithm. In: *J. R. Stat. Soc. Ser. B*, 39(1):pp.1–38. URL `http://www.ams.org/leavingmsn?url=http://links.jstor.org/sici?sici=0035-9246(1977)39:1{%}3C1:MLFIDV{%}3E2.0.CO;2-Z{&}origin=MSN`.

[14] Xiao L and Boyd S (2004) Fast linear iterations for distributed averaging. In: *Systems and Control Letters*, 53:pp.65–78.

[15] Boyd S, Ghosh A, Prabhakar B and Shah D (2006) Randomized gossip algorithms. In: *IEEE Trans. Inf. Theory*, 52(6):pp.2508–2530.

[16] Shah D (2007) Gossip Algorithms. In: *Found. Trends Netw.*, 3(1):pp.1–125.

[17] Shutin D, Kulkarni SR and Poor HV (2012) Incremental Reformulated Automatic Relevance Determination. In: *IEEE Trans. Signal Process.*, 60(9):pp.4977 – 4981.

[18] Wipf D and Nagarajan S (2007) A New View of Automatic Relevance Determination. In: *Proc. 21 Annual Conf. Neural Information Processing Systems*, MIT Press, Vancouver, British Columbia, Canada.

[19] Tipping ME and Faul AC (2003) Fast marginal likelihood maximisation for sparse Bayesian models. In: *Proc. 9th Int. Workshop Artificial Intelligence and Statistics*, Key West, FL, USA.

[20] Manss C, Shutin D and Leus G (2018) Distributed Splitting-over-Features Sparse Bayesian Learning with Alternating Direction Method of Multipliers. In: *IEEE Int. Conf. on Acoustic, Speeach anmd Signal Process.*.

[21] Boyd S and Vandenberghe L (2004) *Convex Optimization*. Cambridge University Press, Cambridge, UK.

[22] Rasmussen CE and Williams CKI (2005) *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press. ISBN 026218253X.

[23] Whaite P and Ferrie F (1997) Autonomous exploration: driven by uncertainty. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):pp.193–205. ISSN 01628828. doi: 10.1109/34.584097.

[24] Krause A, Singh A and Guestrin C (2008) Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. In: *J. Mach. Learn. Res.*, 9:pp.235–284. ISSN 1532-4435.

[25] Singh A, Ramos F, Whyte HD and Kaiser WJ (2010) Modeling and Decision Making in Spatio-Temporal Processes for Environmental Surveillance. In: *IEEE International Conference on Robotics and Automation*, pp.5490–5497.

[26] Viseras A, Xu Z and Merino L (2018) Distributed Multi-Robot Cooperation for Information Gathering under Communication Constraints. In: *IEEE International Conference on Robotics and Automation*.

[27] Rasmussen CE and Williams CK (2005) *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

[28] LaValle SM and Kuffner JJ (2001) Randomized kinodynamic planning. In: *The International Journal of Robotics Research*, 20(5):pp.378–400.

[29] Farinelli A, Rogers A, Petcu A and Jennings NR (2008) Decentralised coordination of low-power embedded devices using the max-sum algorithm. In: *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*, International Foundation for Autonomous Agents and Multiagent Systems, pp.639–646.