Low-Cost Multi-Robot Localization

Amanda Prorok, Alexander Bahr and Alcherio Martinoli

Abstract Localization is an enabling technology, and a prerequisite for a wide range of robotic tasks. Despite the large amount of work already done in this domain, to date, the solution to the localization problem for fully decentralized, large-scale multi-robot systems is still an open question. In this chapter, we contribute to this particular problem outline by proposing a low-cost method: we describe a fully decentralized algorithm, particularly designed for resource-limited robotic platforms in large-scale systems. In the following sections, we elaborate the components of our method, and demonstrate the utility of our low-cost localization algorithm on groups of up to ten real mobile robots. This chapter is rounded off by bringing our approach into a larger perspective, and by discussing its potential as well as its limitations.

1 Introduction

A variety of tasks performed by multi-robot systems such as search and rescue [12, 13], environmental monitoring [5, 25], and construction of real structures [16, 32] need accurate localization to succeed. Due to the intrinsic nature of such tasks, the individual agents are often confined to a small size and weight, which sets hard limits on on-board resources. Simultaneously, a large portion of the robot's resources may be dedicated to the task at hand, especially when this task requires high-frequency perception-to-action loops, leaving little room for solving the localization problem. These compounding problems pose the challenge of designing systems and algorithms that can flexibly accommodate given restrictions, without compromising performance.

Amanda Prorok, Alexander Bahr and Alcherio Martinoli
Distributed Intelligent Systems and Algorithms Laboratory, School of Architecture, Civil and Environmental Engineering, Ecole Polytéchnique Fédérale de Lausanne (EPFL), Switzerland
e-mail: [amanda.prorok|alexander.bahr|alcherio.martinoli]@epfl.ch

This chapter presents a concise solution to the localization problem for a collaborative team of mobile robots. Various strategies have been followed in past works on collaborative localization—our work distinguishes itself by respecting the following design goals:

Low-cost: The time/energy spent on the localization algorithm must be inferior to that spent on the actual task at hand. Thus, we try to minimize the overall complexity of our algorithm, and simultaneously relax the communication requirements.

Full decentralization: Each robot carries responsibility for its own localization, and runs an independent localization algorithm on-board.

Any-time relative observations: Robot-to-robot observations can be made asynchronously, at any given time. This simultaneously means that there are no connectivity constraints on the robot team, and that the computational time of fusing relative observations with proprioceptive sensing is bounded.

Mobility: Since our system is decoupled and decentralized, we do not constrain mobility by making use of any methods that rely on motion agreements among the robots.

Independence of the environment: In order for our method to be equally suited for indoor and outdoor applications, in structured as well as unstructured environments, it should be self-contained and robust. Thus, we rely only on inter-robot relative sensing, and on the possibility of an initial localization (of one of the robots).

Given its efficiency in solving localization problems for unknown initial conditions and its efficiency in accommodating arbitrary probability density functions, our method of choice is the particle filter. We thus build on the general probabilistic framework of Monte-Carlo Localization (MCL) presented in [7]. In particular, our collaboration strategy exploits associated, inter-robot relative range and bearing observations. In order to accommodate the noise characteristics of typical relative range and bearing measurements, we develop a robot detection model, which is introduced into our localization algorithm. This combination forms the basis of our collaborative paradigm. Given this foundation, the key element of our approach consists of an additional routine, namely a *reciprocal* particle sampling routine, mainly designed to accelerate the convergence of a robot's position estimate (to the correct value), and to mitigate overconfidence. A collaborative localization algorithm composed of the aforementioned robot detection model jointly with the reciprocal sampling routine is very efficient with respect to its non-collaborative counterpart. However, due to the computational overhead induced by the detection model and the reciprocal sampling routine (which scale to the square of the number of particles), such an algorithm may run into real-time running constraints. This can turn out to be particularly prohibitive for platforms with hard limits on available resources. For this reason, we further extend our approach with a *particle clustering* method that reduces the complexity of the overall localization algorithm and also reduces the amount of data to be communicated. This clustering routine is especially designed to accommodate the characteristics of the range and bearing robot detection model,

and does not impose an additional computational burden on the localization algorithm as a whole.

1.1 Related Work

A synopsis of currently available work on multi-robot localization promotes a division of the various approaches into two main categories: a *multi-centralized* approach, and a *decentralized* approach. The *multi-centralized* approach distinguishes itself formally from the *decentralized* approach by imposing that each robot in the team maintains a state vector containing the poses of all robots—in other words, each robot maintains a full-system state estimate, versus an estimate of only its own pose.

The *multi-centralized* [23] approach is indeed the more popular (and classical) approach, as it enables the robots to directly take account of inter-robot dependencies and to estimate correlations. However, it entails some inconveniences. In an early work, Roumeliotis et al. [30] enable the distribution of a Kalman estimation scheme by constructing communicating filters, which allows team-members to propagate their state and covariance estimates independently. Yet, as covariance matrix updates occur during each update step and require information exchange between all robots and a centralized processor, the method is particularly vulnerable to single-point failures. In particular, the requirement to update the information in all robots after a single observation of an individual robot assumes a communication infrastructure without any packet loss. The method scales in $O(N^3)$ with respect to the number of robots, and thus limits its scalability due to the high computational cost. In [19], Martinelli et al. propose an extension to [30], which relaxes the assumptions on relative observations, but without further improving the algorithm's scalability and cost. Howard et al. [10] propose an algorithm based on maximum likelihood estimation, and validate it on a team of four real robots. Their method relies on periodical information broadcasts, and it is unclear how the method scales and how sensitive it is to local minima. In a recent work, Nerurkar et al. [24] address the reduction of computational complexity and single-point failures by implementing a maximum a posteriori estimation method. Nevertheless, the $O(N^2)$ computational cost is significant. Also, the proposed method requires synchronous communication among the robots, and its feasibility still remains to be validated on real robots. Mourikis et al. [21] consider the problem of resource-constrained collaborative localization with the goal of deriving optimal sensing frequencies. Yet, as exteroceptive data is dealt with in a centralized way, the sensing frequencies inevitably decrease with an increasing number of robots, thus limiting the scalability of the approach. Cristofaro et al. [6] present a localization algorithm that arguably alleviates the problems described above. The approach is based on an extended information filter, whose implementation is distributed over the robot team members. However, its computational cost increases for each new observation made and it assumes bidirectional synchronous communication, the feasibility of which remains to be evaluated on

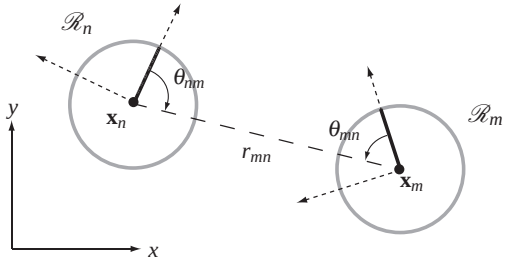


Fig. 1 System of two robots $\mathscr{R}_n$ and $\mathscr{R}_m$ at positions $\mathbf{x}_n$ and $\mathbf{x}_m$, respectively, sharing a common localization frame. The figure illustrates the robots' relative range ($r_{nm} = r_{mn}$) and bearing values (θ_{nm} and θ_{mn}).

real robots. Finally, Leung et al. [15] develop a framework based on 'checkpoints' which facilitates decentralization of a given localization algorithm. Their method, however, still aims to maintain full-system state estimates on all robots, and remains to be evaluated on real robots.

The category of work representing the *decentralized* approach has an alternative take on the collaborative localization problem: each robot maintains an estimate of only its own pose, and fuses relative observations in an opportunistic fashion. Fox et al. [7] first introduced a multi-robot Monte-Carlo localization algorithm for *global* localization, that also relaxes noise assumptions as well as inter-robot dependencies. They propose a method in which robots mutually synchronize their position beliefs upon detection, and show successful global localization on two real robots. However, the method has limited scalability due to overconfidence occurring upon multiple robot detections, and no analysis is provided of the algorithm's processing requirements. Bahr et al. [1] develop a decentralized localization algorithm, based on the extended Kalman filter framework, that is especially well suited for autonomous underwater vehicles with very low data rates. This method, however, allows cyclic updates and, thus, may suffer from overconfidence. In an addition to this work [2], the authors remedy the overconfidence problem, but at the cost of a computationally expensive solution (in particular for a large number of robots and a high frequency of relative observations).

1.2 Problem Formulation

Let us consider a multi-robot system of N robots $\mathscr{R}_1$, $\mathscr{R}_2$, ..., $\mathscr{R}_N$, in a 2D space, where the number N does not necessarily need to be known by the robots (see Figure 1 for a schematic illustration of a two-robot system). For a robot $\mathscr{R}_n$, at time t, the pose $\mathbf{x}_{n,t}$ is given by the Cartesian coordinates $x_{n,t}, y_{n,t}$ and orientation $\phi_{n,t}$. Also, at time t, a robot $\mathscr{R}_m$ is in the set of neighbors $\mathscr{N}_{n,t}$ of robot $\mathscr{R}_n$ if robot $\mathscr{R}_m$ is able to take a range measurement $\tilde{r}_{mn,t}$ and bearing measurement $\tilde{\theta}_{mn,t}$ of robot $\mathscr{R}_n$. Thus, at every moment in time, the neighborhood topology is defined by the physical constraints given by the relative observation sensors deployed on the robots. Also, if $\mathscr{R}_m \in \mathscr{N}_{n,t}$, we make the assumption that the robot $\mathscr{R}_m$ can communicate with the robot $\mathscr{R}_n$. Apart from a sensing modality that enables the robots to observe inter-

robot range and bearing (including a unique robot identifier), they are also equipped with a dead-reckoning self-localization module (e.g., odometry), but do not make use of any exteroceptive sensors capable of feature recognition.

As introduced earlier in this text, every robot runs its own, self-contained, collaborative particle filter, with the goal of localizing itself without any prior knowledge of the initial state or previous measurements. In practice, we assume that one of the robots is localized at the start of an exercise. It turns out that, as time evolves, our method 'propagates' the correct position belief from robot to robot with help of the relative positioning sensors, and that at some point in time all robots are localized (with respect to an upper error bound). The belief of a robot's pose is formulated as

$$\mathbf{Bel}(\mathbf{x}_{n,t}) \sim \{\langle \mathbf{x}_{n,t}^{[i]}, w_{n,t}^{[i]} \rangle | i = 1, ..., M\} = X_{n,t} \tag{1}$$

where M is the number of particles, $\mathbf{x}_{n,t}^{[i]}$ is a sample of the random variable $\mathbf{x}_{n,t}$ (the pose), and $w_{n,t}^{[i]}$ is its weight (or importance factor). The symbol $X_{n,t}$ refers to the set of particles $\langle \mathbf{x}_{n,t}^{[i]}, w_{n,t}^{[i]} \rangle$ at time t belonging to robot $\mathscr{R}_n$. This context formalizes the scope of this chapter: the method that we detail in the following sections solves the localization problem for large robot teams by exploiting collaboration.

We note that the nature of this problem scenario relates well to current real-world scenarios. In particular, in environments where it is hard or even impossible to get a GPS position update, such as underwater or inside buildings, it is always possible to exploit the mobility of one of the team mates to move into a GPS-friendly environment. In underwater robot teams [1], a robot can surface to get a GPS update. Or similarly, in search and rescue robot teams [12], a robot can navigate to the exit of a building. Several authors also comment on the advantages of heterogeneity in robot teams. Bahr et al. [1] note that for optimal localization, it is advantageous to have a few team members that are able to maintain an accurate estimate of their position through sophisticated dead-reckoning sensors, thus enabling a much larger group of robots with less sophisticated sensors to maintain an accurate position. In the same line of thought, Madhavan et al. [18] argue that when the quality of the measurements from absolute positioning sensors deteriorates for certain robots in the team, or if some of the team members do not possess absolute positioning capabilities, those robots can take advantage of other team members with complementary positioning capabilities.

1.3 Case Study

To give the reader a feel for our algorithms, we perform several experiments on a team of Khepera III robots[1] [27]. The Khepera III robot (see Figure 2) has a diameter of 12 cm, making it appropriate for multi-robot experiments in controlled environments. It has a KoreBot extension board providing a standard embedded Linux

[1] http://www.k-team.com/

Fig. 2 Fleet of ten Khepera III robots. The robots are all equipped with an inter-robot relative range and bearing module, which is composed of a ring of 16 infrared light emitting diodes (LEDs).

operating system on an Intel XSCALE PXA-255 processor running at 400 MHz, and uses a communication infrastructure enabled through an IEEE 802.11b wireless card which is installed in a built-in CompactFlash slot. In order to measure the ground truth positioning to evaluate our algorithms, we installed an overhead camera system as detailed in [27], in combination with the open source tracking software SwisTrack [17]. This system allows us to monitor our robots in real-time with a mean error of about 1 cm and a maximum error below 3 cm. The robots are equipped with wheel encoders and use odometry for self-localization. Each robot also uses a relative range and bearing module [29], which provides the relative observations used by the robot detection model. Figure 2 shows ten robots equipped with a relative positioning module. In our experimental space, the boards have a proportional, additive Gaussian range noise with a standard deviation of $\sigma_r = 0.15 \cdot r_{mn}$, a bearing noise of $\sigma_\theta = 0.15$ rad. In the following, we will discuss the localization performance in terms of the mean positioning error of all particles in the robots' beliefs with respect to the ground truth positions obtained from the overhead camera system. This metric implicitly includes the spread of the particle positions, and thus also represents the uncertainty of the position estimate.

2 Collaborative Localization

In this section, we elaborate our collaborative localization algorithm [26], which, together with the Monte-Carlo Localization (MCL) method presented in [7], forms the baseline for our work. For convenience, the complete localization algorithm is shown in Algorithm 1.

2.1 Multi-Robot Monte Carlo Localization

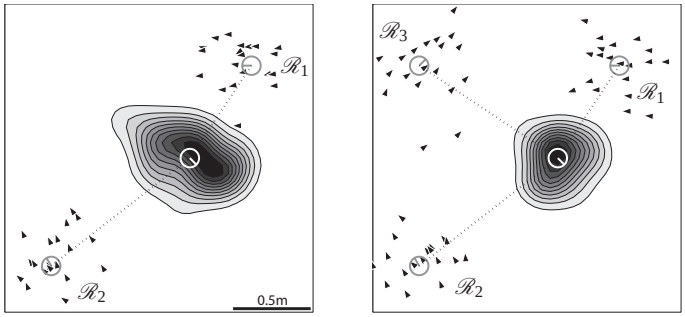
Let us from here on consider a robot $\mathscr{R}_n$ that is detected by robot $\mathscr{R}_m$, and simultaneously receives localization information from robot $\mathscr{R}_m$. If we make the assumption that individual robot positions are independent, we can formulate the update of the belief of robot $\mathscr{R}_n$ at time t with

$$\mathbf{Bel}(\mathbf{x}_{n,t}) = p(\mathbf{x}_{n,t}|u_{n,0..t}) \cdot \int p(\mathbf{x}_{n,t}|\mathbf{x}_{m,t}, r_{mn,t}, \theta_{mn,t}) \mathbf{Bel}(\mathbf{x}_{m,t}) \, \mathrm{d}\mathbf{x}_{m,t} \tag{2}$$

where $u_{n,0..t}$ is the sequence of motion control actions up to time t. For such a collaboration to take place, robot $\mathscr{R}_m$ needs to communicate its range and bearing measurements $\tilde{r}_{mn,t}$, $\tilde{\theta}_{mn,t}$ and $\mathbf{Bel}(\mathbf{x}_{m,t})$ to robot $\mathscr{R}_n$. Thus a communication message is composed as $d_{mn,t} = \langle \tilde{r}_{mn,t}, \tilde{\theta}_{mn,t}, X_{m,t} \rangle$. If several robots in a neighborhood $\mathscr{N}_{n,t}$ communicate with robot $\mathscr{R}_n$, the received information is the set of all relative observations made by those robots of robot $\mathscr{R}_n$ at time t, as well as the belief representations $X_{m,t}$ of all detecting robots $\mathscr{R}_m \in \mathscr{N}_{n,t}$. We denote this data set as $D_{n,t} = \{d_{mn,t} | \mathscr{R}_m \in \mathscr{N}_{n,t}\}$. We note that the collaborative aspect of this formalism lies in the integration of robot $\mathscr{R}_m$'s belief into that of robot $\mathscr{R}_n$ (this update step is shown in Algorithm 1 in line 5). As previously discussed in [7], there are certain limitations to this approach. Due to the fact that robot $\mathscr{R}_m$ integrates its position belief into that of robot $\mathscr{R}_n$ upon detection, subsequent detections would induce multiple integrations of this belief, ultimately leading to an overconfident (and possibly erroneous) belief of the actual pose. Fox et al. remedy this shortcoming by considering two rules: *(i)* their approach does not consider negative sights (no detection) of other robots, and *(ii)* they define a minimum travel distance which a robot has to complete before detecting a same robot again. Although rule *(i)* is a practical consideration, rule *(ii)* limits the scalability and robustness of the approach. In fact, it does not respect our design goals of full mobility and any-time observations (see Sec. 1). We will see in the following sections how our approach tackles this problem by exploiting a reciprocal sampling method.

Algorithm 1 `MultiRob_Recip_MCL`$(X_{n,t-1}, u_{n,t}, z_{n,t}, D_{n,t})$

1: $\overline{X}_{n,t} = X_{n,t} = \emptyset$
2: **for** $i = 1$ to M **do**
3: $\quad \mathbf{x}_{n,t}^{[i]} \leftarrow$ `Motion_Model`$(u_{n,t}, \mathbf{x}_{n,t-1}^{[i]})$
4: $\quad w_{n,t}^{[i]} \leftarrow$ `Measurement_Model`$(\mathbf{x}_{n,t}^{[i]})$
5: $\quad w_{n,t}^{[i]} \leftarrow$ `Detection_Model`$(D_{n,t}, \mathbf{x}_{n,t}^{[i]}, w_{n,t}^{[i]})$
6: $\quad \overline{X}_{n,t} \leftarrow \overline{X}_{n,t} + \left\langle \mathbf{x}_{n,t}^{[i]}, w_{n,t}^{[i]} \right\rangle$
7: **end for**
8: **for** $i = 1$ to M **do**
9: $\quad r \sim \mathscr{U}(0,1)$
10: $\quad$ **if** $r \leq (1-\alpha)$ **then**
11: $\quad\quad \mathbf{x}_{n,t}^{[i]} \leftarrow$ `Sampling`$(\overline{X}_{n,t})$
12: $\quad$ **else**
13: $\quad\quad \mathbf{x}_{n,t}^{[i]} \leftarrow$ `Reciprocal_Sampling`$(D_{n,t}, \overline{X}_{n,t})$
14: $\quad$ **end if**
15: $\quad X_{n,t} \leftarrow X_{n,t} + \left\langle \mathbf{x}_{n,t}^{[i]}, w_{n,t}^{[i]} \right\rangle$
16: **end for**
17: return $X_{n,t}$



(a) (b)

Fig. 3 Example application of the detection model for multiple detecting robots (a) for two robots and (b) for three robots. Here, a set of 20 particles is shown, represented by oriented triangles superimposed over the detecting robots $\mathscr{R}_1$, $\mathscr{R}_2$, and $\mathscr{R}_3$. The detected robot is shown in white. The model's probability density is superimposed on the detected robot. The dotted line and the orientation of the robots show the actual relative range and bearing. The particle positions were generated randomly from a normal distribution ($\sigma_x = \sigma_y = 0.2$ m, and $\sigma_\phi = 0.2$ rad), and range values are perturbed by an additive Gaussian noise with $\sigma_r = 0.15$ and for the bearing values with $\sigma_\theta = 0.15$ rad.

2.2 Range and Bearing Detection Model

The detection model $p(\mathbf{x}_n|d_{mn})$ describes the probability that robot $\mathscr{R}_m$ detects robot $\mathscr{R}_n$ at pose $\mathbf{x}_n = [x_n, y_n, \phi_n]^\mathsf{T}$, given the detection data d_{mn}. This probability density function is applied to the ensemble of particles in the belief of robot $\mathscr{R}_n$, in order to adjust their weights to current relative observations. Given the nature of relative observations, we make use of a locally defined polar coordinate system. Hence, we define the transformation from Euclidean to polar coordinates $\mathbf{T}_e^p(\mathbf{x}_q, \mathbf{x}_p)$ as

$$\mathbf{T}_e^p(\mathbf{x}_q, \mathbf{x}_p) = \begin{bmatrix} r_{qp} \\ \theta_{qp} \end{bmatrix} \tag{3}$$

where

$$r_{qp} = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2} \tag{4}$$

$$\theta_{qp} = \text{atan2}((y_p - y_q), (x_p - x_q)) - \phi_q \tag{5}$$

and $\mathbf{x}_q$ defines the center of the local polar coordinate system. Thus, assuming Gaussian noise and knowledge of the range and bearing standard deviations (σ_r and σ_θ, respectively), and the independence of range and bearing measurements, the detection model is

$$p(\mathbf{x}_n|d_{mn}) = \eta \cdot \sum_{\left\langle \mathbf{x}_m^{[i]}, w_m^{[i]} \right\rangle \in X_m} \Phi\left(\mathbf{T}_e^p(\mathbf{x}_m^{[i]}, \mathbf{x}_n); \boldsymbol{\mu}, \Sigma\right) \cdot w_m^{[i]} \tag{6}$$

where $\Phi(\cdot;\boldsymbol{\mu},\Sigma)$ is a multivariate normal probability density function with mean $\boldsymbol{\mu} = [\tilde{r}_{mn,t}, \tilde{\theta}_{mn,t}]^{\mathsf{T}}$ and where η is a normalization constant. The covariance matrix is $\Sigma = \mathrm{diag}([\sigma_r^2, \sigma_\theta^2])$ (the work in [8] provides experimental evidence for our platform showing that a range and bearing measurement behaves like two independent Gaussian variables). As seen in [26], the detection model can easily be augmented by an additional component in case robot $\mathscr{R}_n$ reciprocally detects robot $\mathscr{R}_m$. Here, for the purpose of our case-study, we use a simple Gaussian distribution in polar coordinates, but all reasonings are valid for completely arbitrary distributions. Indeed, since we use a particle filter, we can keep the same framework for any possible underlying range and bearing hardware not fulfilling the Gaussian noise assumption.

Finally, the detection model incorporating the detection data from multiple detecting robots can be formulated as the update equation shown in Algorithm 2. Figure 3 illustrates the probability density function resulting from the detection model, (a) for two detecting robots, and (b) for three detecting robots. We notice that when detection data from multiple robots is integrated into the range and bearing model, the detection precision increases.

Algorithm 2 `Detection_Model`$(D_{n,t}, \mathbf{x}_t^{[i]}, w_t^{[i]})$

1: $w \leftarrow w_t^{[i]} \cdot \prod_{d_{mn} \in D_{n,t}} p(\mathbf{x}_t^{[i]} | d_{mn})$
2: return w

2.3 Reciprocal Sampling

In addition to using a robot detection model for updating the belief representation $\mathbf{Bel}(\mathbf{x}_{n,t})$, our approach relies on a *reciprocal* sampling method. Let us refer to the iterative process described in Algorithm 1: instead of sampling a new particle pose $\mathbf{x}_{n,t}^{[i]}$ from $\mathbf{Bel}(\mathbf{x}_{n,t-1}^{[i]})$ in line 11, the reciprocal MCL routine in line 13 samples from the detection model $p(\mathbf{x}_n | d_{mn})$, according to Eq. 6. Thus, samples are drawn at poses which are probable given reciprocal robot observations, and which are independent of the previous belief $\mathbf{Bel}(\mathbf{x}_{n,t-1})$. By defining a reciprocal sampling proportion α, particles are sampled from the robot's own belief with a probability $1-\alpha$, and with a probability of α from the probability density function proposed by the detection model. The advantages of this procedure are twofold. Firstly, as the reciprocal sampling method exploits the information available in a whole robot team, it continuously creates particles in areas of the pose space which are likely to be significant, and thus it allows for very small particle set sizes (also shown in [26]). Secondly, by sampling new particles from the detection model, the method introduces a variance proportional to that of the relative detection sensors into the belief of the detected robot (this proportion can be tuned by varying α), and effectively mitigates overconfidence. Algorithm 3 shows the routine where line 4 represents the sampling

step. There are a multitude of methods which can be applied to sample from a given distribution. In our particular case (multi-modal Gaussians), sampling from the detection model $p(\mathbf{x}_n|d_{mn})$ is cheap. For more complex probability density functions, sophisticated and efficient methods such as slice sampling [22] can be employed.

The idea of extending standard MCL with additional sampling methods was first shown in [31]. The resulting algorithm named *Mixture* MCL was shown to increase the robustness of single-robot global localization. Our method differs from that one in that it extends to collaborative multi-robot localization algorithms by sampling from the detection model of one or several mobile robots (whose positions are initially unknown) as opposed to sampling from the detection model of a potentially large set of static environmental features (whose positions have to be known or mapped a priori). Indeed, for complex environments, the method in [31] must be preceded by a fingerprinting process.

Algorithm 3 `Reciprocal_Sampling`$(D_{n,t}, \overline{X}_{n,t})$

1: **if** $D_{n,t} = \emptyset$ **then**
2: $\quad \mathbf{x} \leftarrow$ `Sampling`$(\overline{X}_{n,t})$
3: **else**
4: $\quad \mathbf{x} \sim \prod_{d_{mn} \in D_{n,t}} p(\mathbf{x}|d_{mn})$
5: **end if**
6: return $\mathbf{x}$

We illustrate the effect of reciprocal robot detections by performing a short experiment involving two Khepera III robots, one of which is initially localized. Figure 4 shows the localization error for the second, initially unlocalized robot: In comparison to the standard sampling algorithm (Algorithm 1 with $\alpha = 0$), we see that the reciprocal sampling algorithm (Algorithm 1 with $\alpha > 0$) reduces the localization error by taking better advantage of information available on the localized teammember. Additionally, in this case where the first robot is well localized during this short time span, an increased reciprocal sampling proportion α is more efficient due to the higher probability of drawing accurate reciprocal samples.

Figure 5 shows results obtained in an experiment of 3.5 minutes duration involving ten robots (with one of the robots initially localized). The plots discuss the sensitivity of our algorithm with respect to the number of particles M, as well as its robustness with respect to communication failures. Figure 5(a) shows the localization performance (averaged over time and robots) for a variable number of particles. Larger particle sets contribute to an improved localization accuracy. Yet, an 8-fold increase in the number of particles produces a reduction of only 25% of the localization error. This result coincides with the conclusions made in [26], where it was shown that by increasing the number of particles, the performance converges to that of an ideal localization filter with an infinity of particles. Figure 5(b) shows the localization performance for variable message failure rates. Increasing failure rates induce a graceful degradation of the localization performance. This result confirms

the algorithm's robustness with respect to communication failures, which ultimately reinforces the underlying asynchronous nature of our collaborative paradigm.

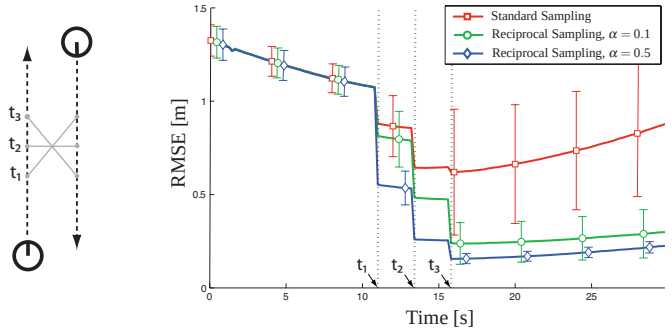


Fig. 4 (a) Schematic illustration of two robots driving past each other. Three detections are made. (b) Localization error for an initially unlocalized robot. It detects a localized robot three times along its path. The standard and reciprocal sampling algorithms (employing 50 particles) are tested 1000 times on the data set. The times at which the observations are made are marked by dotted lines (11.2s, 13.6s, 16s).

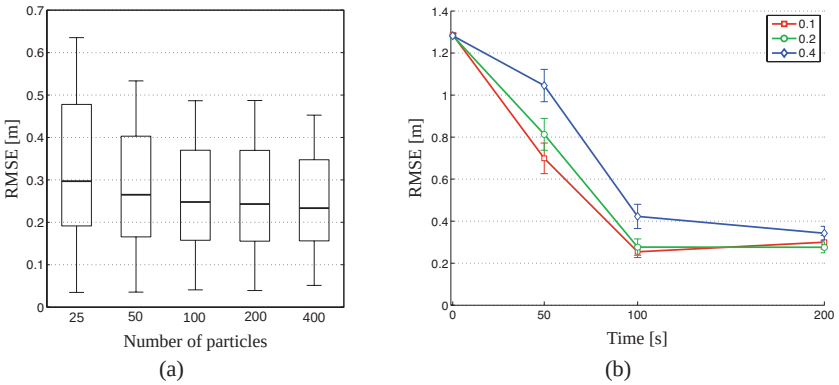


Fig. 5 Localization error for 100 evaluations of the reciprocal sampling algorithm, employing 100 particles per robot and a reciprocal sampling rate $\alpha = 0.06$. (a) Boxplots show the 25th, 50th and 75th percentile, with whiskers containing 85% of the data (for all robots and time). The algorithm is tested employing $\{25, 50, 100, 200, 400\}$ particles per robot. (b) Average error over all robots. Detection data messages are corrupted by a failure rate of $\{0.1, 0.2, 0.4\}$. The errorbars show 95% confidence intervals.

3 Particle Clustering

The algorithm complexity of the detection model $p(\mathbf{x}_n|d_{mn})$ (Eq. 6) leads to $O(M^2)$ for Algorithm 1. This cost can be prohibitive for a large number of particles M (i.e., large with respect to available computational resources). Also, a multi-robot system may have communication constraints that make sending large particle sets infeasible. Hence, even though the method applied in this paper allows for very small particle sets [26], we resort to a clustering method to further reduce the computational and communication overhead.

Let us consider a case where robot $\mathscr{R}_m$ detects robot $\mathscr{R}_n$. For better clarity in the following derivations, we will assume that $|\mathscr{N}_{n,t}| = 1$. The goal of the clustering method is to reduce the number of operations needed to compute the probability density function $p(\mathbf{x}_n|d_{mn})$. Thus, for every detection that it makes, robot $\mathscr{R}_m$ resorts to a clustering method which summarizes its set X_m composed of M particles to a set $\hat{X}_m$ composed of K cluster abstractions (or centroids), reducing the overall computational cost to $O(MK)$ (this clustering routine is detailed later, in Algorithm 4 of Section 3.1). The resulting partition of the particle set is denoted $\mathscr{C}_m$, with $|\mathscr{C}_m| = K$. An individual cluster $c_m^{[k]} \in \mathscr{C}_m$ is defined as the set of particles

$$c_m^{[k]} = \{\langle \mathbf{x}_m^{[i]}, w_m^{[i]} \rangle \mid f(\langle \mathbf{x}_m^{[i]}, w_m^{[i]} \rangle, \cdot) = k\}, \tag{7}$$

where f is a function mapping a particle to a cluster index. Also, we define $\overline{c}_m^{[k]}$ as the data abstraction of cluster $c_m^{[k]}$, representing all particles in its set by the tuple

$$\overline{c}_m^{[k]} = \langle \hat{\mathbf{x}}_m^{[k]}, \hat{w}_m^{[k]}, \hat{\boldsymbol{\mu}}_m^{[k]}, \hat{\boldsymbol{\Sigma}}_m^{[k]} \rangle, \tag{8}$$

where $\hat{\boldsymbol{\mu}}_m^{[k]}$ is a two dimensional vector and $\hat{\boldsymbol{\Sigma}}_m^{[k]}$ is a covariance matrix. Thus, $\hat{X}_m = \{\overline{c}_m^{[k]} \mid c_m^{[k]} \in \mathscr{C}_m\}$ is the set of K cluster abstractions. Finally, we denote the clustered detection data as $\hat{d}_{mn} = \langle \tilde{r}_{mn,t}, \tilde{\theta}_{mn,t}, \hat{X}_m \rangle$, which is sent in place of the unclustered detection data d_{mn}. Formally, given the notation introduced above, finding an optimal particle clustering is equivalent to solving the following optimization problem

$$\min_{\hat{d}_{mn}} \mathbf{D}(p(x_n|d_{mn}) \,||\, \hat{p}(x_n|\hat{d}_{mn})), \tag{9}$$

where $\hat{p}$ is an approximated detection model, and $\mathbf{D}$ a distance measure between two probability density functions. Jain et al. [11] point out that in a typical clustering task, the actual grouping (or clustering) and cluster data abstraction (or cluster representation) are separate components of the task and are commonly treated sequentially. Hence, we deal with our problem by dividing it into the two following sub-problems: *(i)* we consider the set of particles X_m and find an optimal way to create a partition $\mathscr{C}_m$, and *(ii)* we consider an arbitrary cluster $c_m^{[k]}$ in $\mathscr{C}_m$ and find an optimal way to determine its cluster abstraction $\overline{c}_m^{[k]}$. For a given set X_m, these two steps together ultimately lead to a set of cluster abstractions $\hat{X}_m$, which, instead of X_m, is included into the detection data tuple $\hat{d}_{mn}$ for every new detection made.

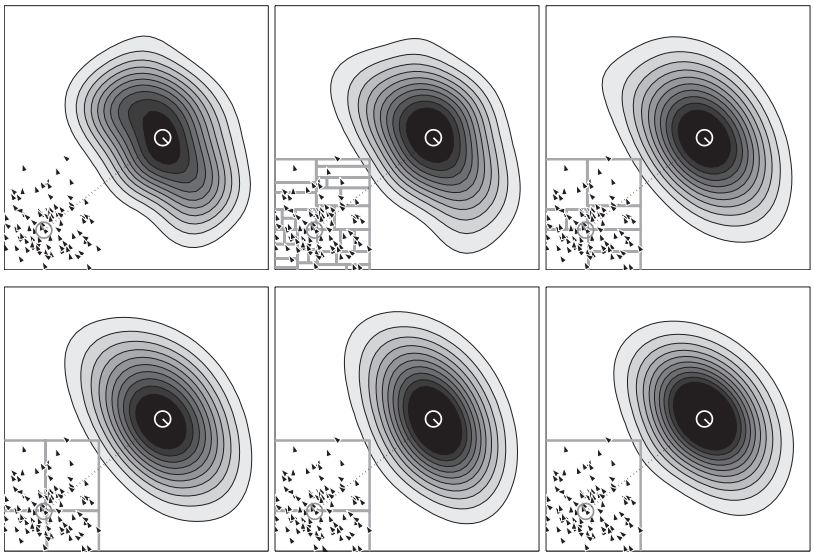


Fig. 6 The detection model (here with range and bearing noise $\sigma_r = 0.15$ and $\sigma_\theta = 0.15$) is projected on the detected robot (in white). Final cluster partitions are superimposed on the particles of the detecting robot. From left to right, top to bottom, the number of clusters K employed by the clustering algorithm is: 100, 32, 8, 4, 2, 1, for a total number of particles $M = 100$.

The following paragraphs detail our low-cost clustering approach that aims to meet these specifications.

Algorithm 4 `Cluster`$(X_{m,t}, K)$

$\hat{X}_m \leftarrow \emptyset$
$c_m^{[1]} \leftarrow X_m$
$\mathscr{C}_m \leftarrow c_m^{[1]}$
for $k = 1$ to $K-1$ **do**
  $k_{max}, \dim \leftarrow$ `find_highest_variance_cluster`$(\mathscr{C}_m)$
  $c_m^{[k_{max}]}, c_m^{[k+1]} \leftarrow$ `split_cluster`$(c_m^{[k_{max}]}, \dim)$
  $\mathscr{C}_m \leftarrow \mathscr{C}_m + c_m^{[k+1]}$
end for
for $k = 1$ to K **do**
  $\overline{c}_m^{[k]} \leftarrow$ `assign_data_abstraction`$(c_m^{[k]})$
  $\hat{X}_m \leftarrow \hat{X}_m + \overline{c}_m^{[k]}$
end for
return $\hat{X}_m$

3.1 Clustering Algorithm

The optimal, combinatorial solution to the clustering problem of Equation 9 requires the evaluation of a very large number of partitions (the number of ways to partition a set of M data points into K non-empty clusters is given by Stirling number of the second kind). Even though efficient approaches have been proposed [14], combinatorial solutions still remain prohibitively expensive. Given the usefulness of clustering in a large range of disciplines, many non-combinatorial clustering approaches have been proposed [11]. Yet, since our goal is to reduce the final complexity of our algorithm, the complexity of the actual clustering algorithm must be at most equal to $O(MK)$. One of the most commonly used low-cost clustering methods is the k-means algorithm [20]. It starts off with a random initial cluster assignment and iteratively reassigns clusters until a convergence criterion is met or a maximum number of iterations L is attained. Although the algorithm has a low time complexity $O(MKL)$, its main disadvantage is that it is sensitive to the initial cluster assignment. The variant ISODATA algorithm [3] is also an iterative clustering algorithm with a time complexity of $O(MKL)$, with the additional capability to split and merge clusters according to predefined threshold values. It is therefore more flexible than the k-means and able to find the optimal partition, provided that the user is able to define correct threshold values. Non-iterative, incremental clustering algorithms have the advantage that they are even less time consuming than iterative algorithms. The leader algorithm [9] is the simplest of that kind. Data points are incrementally assigned to existing clusters based on a distance metric, with new clusters being created if all distance measures exceed a predefined criterion. Yet, given the algorithms incremental nature, the final clustering result is dependent on the order of the assignments made.

We take inspiration from the methods described above to develop a non-iterative, order-independent, non-parametric approach that produces a predefined number of K clusters. Our solution is inspired by the construction of multidimensional binary trees [4], and consists of a 2-dimensional sorting algorithm which repetitively separates the particle set along the mean of the dimension producing the highest variance, until the predefined maximum number of clusters K is attained. We note that splitting along the median instead of the mean incurs a higher complexity. A description of this algorithm is shown in Algorithm 4. The function in line 5 has a complexity $O(M)$, the function in line 6 has a complexity $O(|c_m^{[k_{max}]}|)$, and function in line 10 has a complexity $O(|c_m^{[k]}|)$. Hence, the total algorithm cost is $O(MK)$. Figure 6 shows examples of final cluster partitions for six different total numbers of clusters, performed on an identical set of 100 particles. We note that, even for maximal clustering ($K = 1$), the detection model is well approximated.

3.2 Cluster Abstraction

For an arbitrary cluster $c_m^{[k]}$, we have the non-summarized detection data $d_{mn}^{[k]} = \langle \tilde{r}_{mn,t}, \tilde{\theta}_{mn,t}, c_m^{[k]} \rangle$. The problem of finding an optimal cluster abstraction $\overline{c}_m^{[k]}$ can, thus, be formalized as

$$\mathbf{D_{KL}}(p||\hat{p}) = \int_{-\infty}^{\infty} p(\mathbf{x}_n|\hat{d}_{mn}^{[k]}) \log \frac{p(\mathbf{x}_n|\hat{d}_{mn}^{[k]})}{\hat{p}(\mathbf{x}_n|\hat{d}_{mn}^{[k]})} \mathrm{d}\mathbf{x}_n \tag{10}$$

where $\mathbf{D_{KL}}$ is the Kullback-Leibler divergence, and $\hat{d}_{mn}^{[k]} = \langle \tilde{r}_{mn,t}, \tilde{\theta}_{mn,t}, \overline{c}_m^{[k]} \rangle$ is the summarized detection data. In [28], we showed the following. Given a point $\hat{\mathbf{x}}_{\mathbf{m}}^{[\mathbf{k}]} = [\hat{x}_m^{[k]}, \hat{y}_m^{[k]}, \hat{\phi}_m^{[k]}]^\mathsf{T}$, and the probability density function

$$\hat{p}(\mathbf{x}_n|\hat{d}_{mn}^{[k]}) = \Phi\left(\mathbf{T}_e^p(\hat{\mathbf{x}}_m^{[k]}, \mathbf{x}_n); \hat{\boldsymbol{\mu}}_m^{[k]}, \hat{\Sigma}_m^{[k]}\right), \tag{11}$$

the Kullback-Leibler divergence between p and $\hat{p}$ is minimal if

$$\hat{\boldsymbol{\mu}}_m^{[k]} = \frac{1}{|c_m^{[k]}|} \sum_{\mathbf{x}_m^{[i]} \in c_m^{[k]}} \mathbf{v}_m^{[k,i]}, \tag{12}$$

$$\hat{\Sigma}_m^{[k]} = \frac{1}{|c_m^{[k]}| - 1} \sum_{\mathbf{x}_m^{[i]} \in c_m^{[k]}} \left(\mathbf{v}_m^{[k,i]} - \hat{\boldsymbol{\mu}}_m^{[k]}\right)\left(\mathbf{v}_m^{[k,i]} - \hat{\boldsymbol{\mu}}_m^{[k]}\right)^\mathsf{T} \tag{13}$$

are the mean and covariance of $\mathbf{v}_m^{[k,i]} = \mathbf{T}_e^p(\hat{\mathbf{x}}_m^{[k]}, \breve{\mathbf{x}}_m^{[i]})$, with

$$\breve{x}_m^{[i]} = x_m^{[i]} + r_{mn}\cos(\theta_{mn} + \phi_m^{[i]}) \tag{14}$$

$$\breve{y}_m^{[i]} = y_m^{[i]} + r_{mn}\sin(\theta_{mn} + \phi_m^{[i]}). \tag{15}$$

We note that the above equations do not take into account the uncertainty of the range and bearing observations. Thus, we propose a variant detection model $\hat{p}$ (cf. Equation 6) that explicitly takes into account noise. We have

$$\hat{p}(\mathbf{x}_n|\hat{d}_{mn}) = \eta \cdot \sum_{\overline{c}_m^{[k]} \in \hat{X}_m} \Phi\left(\mathbf{T}_e^p(\hat{\mathbf{x}}_m^{[k]}, \mathbf{x}_n); \hat{\boldsymbol{\mu}}_m^{[k]}, \hat{\Sigma}_m^{[k]} + \Sigma\right) \cdot \hat{w}_m^{[k]} \tag{16}$$

where $\hat{\boldsymbol{\mu}}_m^{[k]}$ and $\hat{\Sigma}_m^{[k]} + \Sigma$ approximate the true mean and covariance, respectively, in the presence of noise (we remind the reader that $\Sigma = \mathrm{diag}([\sigma_r^2, \sigma_\theta^2])$). Indeed, finding a closed form solution for the true values is intractable. However, if the set of particles $c_m^{[k]}$ is densely populated, our approximation is very good. Moreover, if the particle positions coincide, and if for a given cluster $c_m^{[k]}$ the point $\hat{\mathbf{x}}_m^{[k]}$ is its center of mass, the solution is optimal. Hence, we complete the data abstraction

$\overline{c}_m^{[k]} = \langle \hat{\mathbf{x}}_m^{[k]}, \hat{w}_m^{[k]}, \hat{\boldsymbol{\mu}}_m^{[k]}, \hat{\Sigma}_m^{[k]} \rangle$ (cf. Equation 8) with $\hat{\mathbf{x}}_m^{[k]}$ as the weighted center of mass, and $\hat{w}_m^{[k]}$ the cumulative weight

$$\hat{\mathbf{x}}_m^{[k]} = \frac{1}{\hat{w}_m^{[k]}} \cdot \sum_{\langle \mathbf{x}_m^{[i]}, w_m^{[i]} \rangle \in c_m^{[k]}} w_m^{[i]} \cdot [x_m^{[i]}, y_m^{[i]}, \phi_m^{[i]}]^\mathsf{T} \tag{17}$$

$$\hat{w}_m^{[k]} = \sum_{\langle \mathbf{x}_m^{[i]}, w_m^{[i]} \rangle \in c_m^{[k]}} w_m^{[i]}. \tag{18}$$

Finally, we note that the constraints given by our approximated detection model $\hat{p}$ motivate the choice of a clustering algorithm which clusters densely located particles into common clusters (a condition which is satisfied by Algorithm 4).

Figure 7(a) shows the Kullback-Leibler divergence between the full and the approximated detection models p and $\hat{p}$, calculated from a data set gathered by ten robots. The more clusters we employ in the clustering method, the smaller the divergence to the true probability density function. This shows that our clustering method produces a valid representation of the original probability density functions. Figure 7(b) shows the localization performance when employing the clustering method for a variable number of clusters K. We note that the difference of performance between maximal clustering (K = 1) and modest clustering (K = 32) is very small. Finally, to illustrate the localization process, Figure 8 shows eight snapshots based on real data from an experiment performed over an interval of 126s during which one robot (in red) is initially localized. Each robot employed 100 particles with a reciprocal proportion $\alpha = 0.06$, and used the clustering routine with maximal clustering ($K = 1$). This experiment concludes the validation of our approach by showing how ten robots are able to converge to correct position estimates in a nevertheless simple, but effective demonstration scenario.

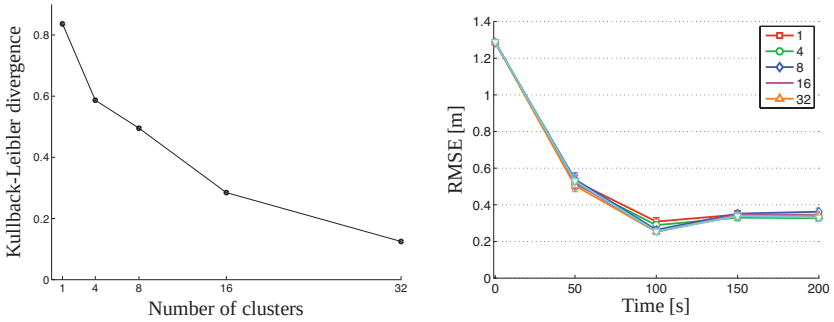


Fig. 7 (a) The Kullback-Leibler divergence between the full and approximated detection models, as a function of the number of clusters employed by the clustering method. (b) Average localization error over 100 evaluations. The localization algorithm is tested, employing the clustering method using {1,4,8,16,32} clusters. The errorbars show 95% confidence intervals.

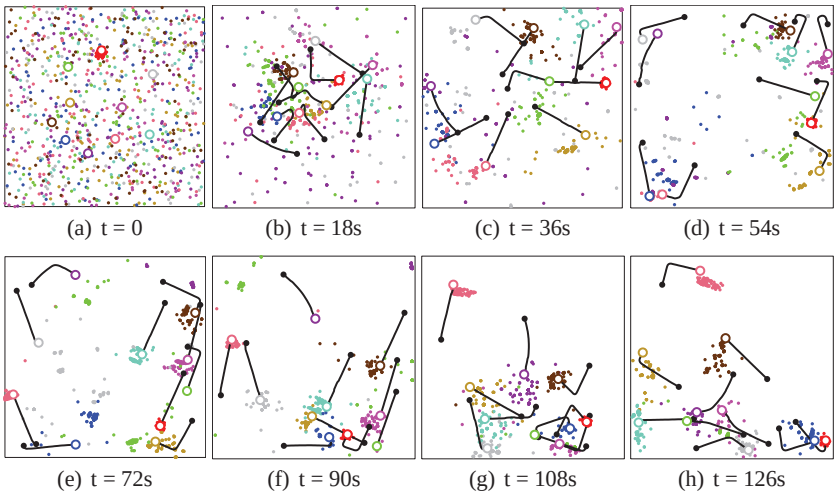


Fig. 8 The figure shows eight snapshots with 18s intervals of an experimental run on the team of ten Khepera III robots. Each robot employed 100 particles with a reciprocal proportion $\alpha = 0.06$, and used the clustering routine with $K = 1$. The black lines show the trajectories completed in the time intervals between snapshots, with the filled black dots representing the robot positions at the end of the previous snapshots. The red robot was initially localized.

4 Conclusion

In this chapter, we presented a fully scalable, probabilistic, multi-robot localization algorithm based on the Monte Carlo method. Its maximal overall complexity is $O(|\mathscr{N}|MK)$, where $|\mathscr{N}|$ is the number of neighboring robots (at a given time, for a given robot in the system), M the number of particles, and K an adjustable number of clusters produced by the clustering algorithm. This clustering method has shown to produce increasingly accurate probability density function representations for large K, and when employed in practice, has shown to perform well even for very small K. Furthermore, given the asynchronous paradigm of our collaboration strategy, the algorithm's update rate is much higher than the inter-robot message communication rate. Thus, the number of detected neighbors $|\mathscr{N}|$ is in practice no higher than 1, and the complete routine complexity is reduced to $O(MK)$. Thus, the algorithm is fully scalable with respect to the number of robots in the system. In addition, the algorithm poses no communication constraints and shows a graceful performance degradation in case of message failures. Our approach was experimentally validated on a team of ten real robots.

Finally, we note that a continuation of this work should consider the following aspects in particular. We evaluated our approach on a baseline experimental setup, where the belief of a robot's position is well represented by a single particle cluster. Hence, more complex scenarios, including obstacles and multi-modal sensor models, may exhibit a significant spread of performance when clustering. In such cases,

a trade-off between the number of clusters K and accuracy must be determined. Also, in severely multi-modal distributions, the construction of the cluster centroid must be revisited. In the same line of thought, more work needs to be done to explore arbitrarily distributed, non-Gaussian detection models as an extension to our generalizable framework.

References

[1] A. Bahr, J. J. Leonard, and M. F. Fallon. Cooperative Localization for Autonomous Underwater Vehicles. *Int. Journal of Robotics Research*, 28(6):714–728, 2009.

[2] A. Bahr, M. R. Walter, and J. J. Leonard. Consistent cooperative localization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3415–3422, 2009. doi: 10.1109/ROBOT.2009.5152859.

[3] G. H. Ball and D. J. Hall. ISODATA, a novel method of data analysis and classification. Technical report, 1965.

[4] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.

[5] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less Low-Cost Outdoor Localization for Very Small Devices. *IEEE Personal Communications*, 7(5):28–34, 2000.

[6] A. Cristofaro, A. Renzaglia, and A. Martinelli. Distributed Information Filters for MAV Cooperative Localization. In *International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2010.

[7] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. A Probabilistic Approach to Collaborative Multi-Robot Localization. *Autonomous Robots*, 8:325–344, 2000.

[8] S. Gowal and A. Martinoli. Bayesian Rendezvous for Distributed Robotics Systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2765–2771.

[9] J. A. Hartigan. *Clustering Algorithms*. John Wiley and Sons, Inc., New York, NY, 1975.

[10] A. Howard, M J Mataric, and G. S. Sukhatme. Localization for mobile robot teams: A distributed MLE approach. *Experimental Robotics VIII*, pages 146–155, 2003.

[11] A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, 31(3):265–323, 1999.

[12] J. S. Jennings, G. Whelan, and W. F. Evans. Cooperative search and rescue with a team of mobile robots. In *International Conference on Advanced Robotics (ICAR)*, pages 193–200, 1997.

[13] G. Kantor, S. Singh, R. Peterson, D. Rus, A. Das, V. Kumar, G. Perreira, and J. Spletzer. Distributed Search and Rescue with Robot and Sensor Teams. *Springer Tracts in Advanced Robotics*, 24:529–538, 2006.

[14] L. G. W. Koontz, P. Narendra, and K. Fukunaga. A Branch and Bound Clustering Algorithm. *IEEE Transactions on Computers*, C-24(9):908–915, 1975.

[15] K. Y. K. Leung, T. D. Barfoot, and H. Liu. Decentralized Localization of Sparsely-Communicating Robot Networks: A Centralized-Equivalent Approach. *IEEE Transactions on Robotics*, 26(1):62–77, 2010. doi: 10.1109/TRO.2009.2035741.

[16] Q. Lindsey, D. Mellinger, and V. Kumar. Construction of cubic structures with quadrotor teams. *Proceedings of Robotics: Science & Systems VII*, 2011.

[17] T. Lochmatter, P. Roduit, C. Cianci, N. Correll, J. Jacot, and A. Martinoli. SwisTrack - A Flexible Open Source Tracking Software for Multi-Agent Systems. In *Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4004–4010, 2008.

[18] R. Madhavan, K. Fregene, and L. E. Parker. Distributed Cooperative Outdoor Multirobot Localization and Mapping. *Autonomous Robots*, 17(1):23–39, 2004.

[19] A. Martinelli, F. Pont, and R. Siegwart. Multi-Robot Localization Using Relative Observations. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2797–2802, 2005.

[20] J. McQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[21] A.I. Mourikis and S. I. Roumeliotis. Optimal sensor scheduling for resource-constrained localization of mobile robot formations. *IEEE Transactions on Robotics*, 22(5):917–931, 2006.

[22] B. Neal. Slice Sampling. *The Annals of Statistics*, 31:705–757, 2003.

[23] E. Nerurkar and S. Roumeliotis. Asynchronous Multi-Centralized Cooperative Localization. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8, 2010. doi: 10.1109/IROS.2010.5650133.

[24] E. Nerurkar, S. Roumeliotis, and A. Martinelli. Distributed maximum a posteriori estimation for multi-robot cooperative localization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1402–1409, 2009.

[25] P. N. Pathirana, N. Bulusu, A. V. Savkin, and S. Jha. Node localization using mobile robots in delay-tolerant sensor networks. *IEEE Transactions on Mobile Computing*, 4(3):285–296, 2005. doi: 10.1109/TMC.2005.43.

[26] A. Prorok and A. Martinoli. A Reciprocal Sampling Algorithm for Lightweight Distributed Multi-Robot Localization. In *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3241–3247, 2011.

[27] A. Prorok, A Arfire, A. Bahr, J R Farserotu, and A. Martinoli. Indoor navigation research with the Khepera III mobile robot: An experimental baseline with a case-study on ultra-wideband positioning. In *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010. doi: 10.1109/IPIN.2010.5647880.

[28] A. Prorok, A. Bahr, and A. Martinoli. Low-Cost Collaborative Localization for Large-Scale Multi-Robot Systems. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4236–4241, 2012.

[29] J. Pugh, X. Raemy, C. Favre, R. Falconi, and A. Martinoli. A Fast On-Board Relative Positioning Module for Multi-Robot Systems. *IEEE Transactions on Mechatronics*, 14(2):151–162, 2009.

[30] S. I. Roumeliotis and G. A. Bekey. Distributed Multirobot Localization. *IEEE Transactions on Robotics and Automation*, 14(5):781–795, 2002.

[31] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo Localization for Mobile Robots. *Artificial Intelligence*, 128(1-2):99–141, 2000.

[32] J. Werfel, Y. Bar-Yam, D. Rus, and R. Nagpal. Distributed construction by mobile robots with enhanced building blocks. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2787–2794, 2006.