Cooperative localization in mobile networks using nonparametric variants of belief propagation

Vladimir Savic, Santiago Zazo

Abstract

Of the many state-of-the-art methods for cooperative localization in wireless sensor networks (WSN), only very few adapt well to mobile networks. The main problems of the well-known algorithms, based on nonparametric belief propagation (NBP), are the high communication cost and inefficient sampling techniques. Moreover, they either do not use smoothing or just apply it offline. Therefore, in this article, we propose more flexible and efficient variants of NBP for cooperative localization in mobile networks. In particular, we provide: i) an optional 1-lag smoothing done almost in real-time, ii) a novel low-cost communication protocol based on package approximation and censoring, iii) higher robustness of the standard mixture importance sampling (MIS) technique, and iv) a higher amount of information in the importance densities by using the population Monte Carlo (PMC) approach, or an auxiliary variable. Through extensive simulations, we confirmed that all the proposed techniques outperform the standard NBP method.

Keywords: cooperative localization, mobile networks, belief propagation, particle filter, smoothing, message approximation, population Monte Carlo, auxiliary particle filter, importance sampling

1. Introduction

Cooperative localization in mobile networks is an important problem, as the availability of location information can enable many applications [1, 2, 3, 4], such as tracking vehicles on roadways without the global positioning system (GPS), firefighters in buildings under fire, forklifts in a warehouse, animals in woods, and search-and-rescue. We consider the case in which some small number of sensors, called *anchor nodes*, obtain their coordinates via GPS [5] (or by installing them at points with known coordinates), and the rest, *target nodes*, must determine their own coordinates. We assume that distance measurements are available between appropriate pairs of sensors. They can be obtained [1, 3] using time of arrival (TOA), time difference of arrival (TDOA), or received signal strength (RSS). If the reference sensors were capable of high-power transmission, they would be able to make measurements with all the anchor nodes. This represents *single-hop* positioning. However, we prefer to use energy-conserving devices without unnecessarily using energy for long-range communication. In this case, each sensor only has available noisy measurements of its distance to several neighboring sensors (not necessarily anchor nodes). It is still necessary that there is minimum of three (for 2D) or four (for 3D) anchor nodes in the network, but not necessarily directly connected to all the target nodes. This technique, also known as *multi-hop* (or *cooperative*) positioning, is fully distributed since each target node is responsible for locating itself using only information from its neighborhood. In case of mobile positioning, target nodes are attached to the objects that should be tracked. Anchor nodes are usually static, but this is not necessary (especially, if they are equipped with GPS).

1.1. Related work

A number of methods for cooperative localization has been proposed, but most of them are adapted for the localization in static networks [6, 7, 8, 9, 10, 11, 12]. Repeating these static localization algorithms can provide location estimates in mobile networks, but this approach is suboptimal due to the lack of additional information provided by the mobility of the sensor nodes. Some works already take this information into account, for example [13, 14, 15, 16, 17]. Moreover, the goal of most localization methods [6, 8, 11, 12, 13] is just to estimate the position of all the target nodes, without associated uncertainty. Since the uncertainty of the estimate is crucial for most applications, a Bayesian approach [9, 10, 16] can be applied. Its goal is to estimate the posterior marginal probability density function (PDF) of the

target's position, given the priors, and measurements. Since this approach is generally intractable, it is necessary to use some message-passing method [18] and also to approximate all PDFs using particle-based approximations [19, 20]. One suitable framework is *nonparametric belief propagation* (NBP), which was initially proposed for static networks [9]. Variants of this method have been already used for cooperative localization in mobile networks [16, 17]. In [16], the authors propose a particle-based distributed message passing method defined on a factor graph. Comparing with NBP, which is defined on a Markov Random Field (MRF), the main difference is the capability to work with higher-order potentials. In [17], the authors use NBP method for distributed tracking of mobile robots. For this application, it is also necessary to estimate the speed of the targets (not only the positions). It also takes advantage of the bidirectional nature of NBP to send the messages from the future to present (known as *smoothing*). However, this is only possible in the case of offline postprocessing.

1.2. Contributions

In this paper, we extend the standard NBP method [9] for mobile positioning. Our main contributions are as follows:

- We provide a flexible algorithm for mobile positioning, in which NBP can provide three different estimates (repeated, filtered, and smoothed) by running it once. In this paper, we specifically focus on 1-lag smoothing since it can be done almost in real time.
- We develop a novel low-cost communication protocol for mobile positioning based on NBP. It is based on i) communication of the beliefs (instead of the messages), ii) approximation of the beliefs, and iii) censoring (i.e., avoiding the transmission of uninformative data). It is almost as accurate as standard NBP (in which the particles from the messages are transmitted).
- We improve the standard mixture importance sampling (MIS) by adding uniformly distributed particles. This makes NBP robust against outliers.
- In order to increase the amount of information in the importance densities, we propose two novel NBP variants: *population Monte Carlo* NBP (PMC-NBP), and *auxiliary* NBP (ANBP). Both of them outperform standard NBP.

The remainder of this paper is organized as follows. In Section 2, we provide the overview of static positioning using NBP. In Section 3, we propose new framework for mobile positioning. Solutions for communication issues are proposed in Section 4, and improved sampling techniques in Section 5. Finally, simulation results are presented in Section 6, and the paper is concluded in Section 7.

2. Overview of static positioning using NBP

2.1. Statistical model

Let us assume that we have N_s sensors (N_a anchors and N_t targets) scattered in some region. The 2D location of a sensor r is denoted by x_r. The target node u obtains a noisy measurement d_{ru} of its distance from some neighboring node r:

$$d_{ru} = \|x_r - x_u\| + v_{ru} \qquad v_{ru} \sim p_v \tag{1}$$

where, for the noise v_{ru}, we can assume that p_v is Gaussian (for TOA), log-Gaussian (for RSS), or any other empirical distribution (found using experiments in the deployment area). This measurement provides us with a likelihood function $p(d_{ru}|x_r, x_u)$ between each pair of neighboring sensors (r, u). In addition, each sensor r has some prior distribution denoted $p_r(x_r)$. This prior could be an uninformative one (i.e., a uniform distribution over the whole deployment area) for the targets, and the Dirac Delta function for the anchors. Then, the joint PDF is given by:

$$p(x_1, ..., x_{N_t}|\{d_{ru}\}) \propto \prod_{(r,u)\in E} p(d_{ru}|x_r, x_u) \prod_{r\in V} p_r(x_r) \tag{2}$$

where (V,E) is an undirected graph which consists of a set of nodes (or vertices) V, and a set of edges E. We also need to define a detection area. For large-scale sensor networks, it is reasonable to assume that only a subset of pairwise distances will be available, primarily between sensors which are located within the some radius R. For simplicity, we use this model, but better approximations of the probability of detection can be obtained using empirical data from the deployment area [9, 21].

Our goal is to compute the posterior marginal PDF $p(x_n|\{d_{ru}\})$ (for each target node n) by marginalizing the joint posterior PDF, which is not tractable for the localization problem. Therefore, we need to find these PDFs using some message-passing method.

2.2. Belief propagation (BP)

Belief propagation [18] is a way of organizing the "global" computation of marginal PDFs in terms of smaller local computations within the graph. It is one of the best-known message-passing methods for distributed inference in statistical physics, artificial intelligence, computer vision, localization, etc. The whole

computation takes a time proportional to the number of links in the graph. This is significantly less than the exponential time that would be required to compute the posterior marginal PDFs naively.

In the standard BP algorithm, taking that the underlying graphical model is a Markov Random Field (MRF), the belief (posterior marginal PDF) at a node r, $M_r(x_r)$, is proportional to the product of the local evidence at that node $\psi_r(x_r)$, and all the messages coming into node r:

$$M_r(x_r) \propto \psi_r(x_r) \prod_{u \in G_r} m_{ur}(x_r) \tag{3}$$

where x_r is a state of node r, and G_r denotes the neighbors of node r. The messages are determined by the message update rule:

$$m_{ur}(x_r) \propto \int_{x_u} \psi_u(x_u)\psi_{ru}(x_r, x_u) \prod_{k \in G_u \setminus r} m_{ku}(x_u) dx_u \tag{4}$$

where $\psi_{ru}(x_r, x_u)$ is the pairwise potential between the nodes r and u. On the right-hand side, there is a product over all the messages going into node u except for the one coming from node r. In other words, the message from node u to node r represents the "opinion" of node u about the location of node r. The messages and beliefs are represented in probabilistic form, but not necessarily normalized.

The relationship between the graph and the joint PDF may be quantified in terms of potential functions ψ, which are defined over each of the graph's cliques. A clique (C) is a subset of nodes such that for every two nodes in C, there exists a link connecting the two. So the joint PDF can be written as:

$$p(x_1, ..., x_{N_u}) \propto \prod_{cliques\, C} \psi_C(\{x_i : i \in C\}) \tag{5}$$

For the distance-based localization, we only need potential functions defined over variables associated with single nodes and pairs of nodes. Single-node potential (prior information about the position) at each node r, and the pairwise potential (likelihood function: $\psi_{ru}(x_r, x_u) = p(d_{ru}|x_r, x_u)$) between nodes r and u, are respectively given by:

$$\psi_r(x_r) = p_r(x_r), \tag{6}$$

$$\psi_{ru}(x_r, x_u) = p_v(d_{ru} - \|x_r - x_u\|). \tag{7}$$

Using (2), we can write the joint posterior PDF, as a function of the potentials:

$$p(x_1, ..., x_{N_t}|\{d_{ru}\}) \propto \prod_{r \in V} \psi_r(x_r) \prod_{(r,u) \in E} \psi_{ru}(x_r, x_u) \tag{8}$$

Now we can marginalize this PDF using the BP algorithm, and use its mean value (or mode) and variance to characterize the target positions. We will use a different form of the BP algorithm, in order to adapt it to the iterative localization scenario, where it is more practical to compute beliefs in each iteration, and use them to update messages. This form can be easily found by replacing (3) in (4), and it is given by the following belief update and message update rules (at iteration i):

$$M_r^i(x_r) \propto \psi_r(x_r) \prod_{u \in G_r} m_{ur}^i(x_r) \tag{9}$$

$$m_{ur}^i(x_r) \propto \int_{x_u} \psi_{ur}(x_r, x_u) \frac{M_u^{i-1}(x_u)}{m_{ru}^{i-1}(x_u)} dx_u \tag{10}$$

In the first iteration of this algorithm, it is necessary to initialize the message to an uninformative state, and the beliefs to the prior ($m_{ur}^1 = 1$ and $M_r^1 = p_r$ for all u, r). Then we can iterate (9) and (10) until sufficient convergence. For tree-like graphs, the number of iterations should be at most the length of the longest path in the graph. In case of loopy graphs, there is no such guarantee, but convergence is often achieved after a similar number of iterations.

2.3. Nonparametric Belief Propagation (NBP)

The presence of nonlinear relationships and potentially (highly) non-Gaussian uncertainties in sensor localization makes standard BP undesirable[1]. However, particle-based representation via *nonparametric belief propagation* (NBP) [9] enables the application of BP to localization in WSN. In NBP, the belief and message update equations, (9) and (10), are performed using stochastic approximations, in two phases: i) first, drawing particles from the belief $M_r^i(x_r)$, ii) then using these particles to approximate each outgoing message m_{ru}^i. The main advantage of this approach is the ability to provide information about location estimation uncertainties (in contrast to deterministic approaches), which are not necessarily Gaussian.

2.3.1. Computing NBP messages

In the first iteration, we draw particles from the initial belief, $M_r^1 = p_r$. In order to decrease the number of particles, we draw them uniformly within a *bounded box* [15, 21]. The bounded box of node r is defined as the rectangular area in which the node is localized. It is created using 1-hop and (eventually) 2-hop distances from the anchor nodes. Given N_p weighted particles $\{W_r^{j,i}, X_r^{j,i}\}$ from the belief $M_r^i(x_r)$ obtained at iteration i,

[1] Note that in some specific cases (e.g., Gaussian measurements), a parametric BP can be used [22].

we can compute weighted particles of the outgoing BP message m^i_{ru}. The distance measurement d_{ru} provides information about how far sensor u is from sensor r, but no information about its relative direction. To draw a particle of the message ($x^{j,i+1}_{ru}$), given the particle $X^{j,i}_r$ which represents the position of the sensor r, we simply select a direction $\theta^{j,i}$ at random uniformly in the interval $[0, 2\pi)$. We then shift $X^{j,i}_r$ in the direction of $\theta^{j,i}$ by an amount which represents the estimated distance between nodes u and r:

$$x^{j,i+1}_{ru} = X^{j,i}_r + (d_{ru} + v^j)[\cos(\theta^{j,i}) \quad \sin(\theta^{j,i})] \tag{11}$$

The particles are weighted by the reminder of message update rule (10):

$$w^{j,i+1}_{ru} = \frac{W^{j,i}_r}{m^i_{ur}(X^{i,j}_r)} \tag{12}$$

For the denominator in the previous equation (and also for (14)), we need to know the parametric form of the message. We approximate it using *kernel density estimate* (KDE)[2]. The optimal value for the bandwidth h^{i+1}_{ru} can be obtained in a number of ways. The simplest technique is to apply the "rule of thumb" estimate [21, 23], but there are also more advanced methods, e.g., *generalized cross entropy* (GCE) estimator (see [27], Alg. 1). In this paper, we apply the GCE approach, since it yields a smaller integrated squared error.

We also need messages coming from the anchor nodes, which can be found using (10) and the belief of the anchor node x^*_r ($M^i_r(x_r) = \delta(x_r - x^*_r)$):

$$m^{i+1}_{ru}(x_u) \propto \psi_{ru}(x^*_r, x_u) \tag{13}$$

Note that, in contrast to [9, 21], we do not transmit messages over unobserved edges (2-hop, 3-hop,...) since we seek a fast method for mobile networks.

2.3.2. Computing NBP beliefs

To estimate the belief $M^{i+1}_u(x_u)$ using (9), we draw particles from the product of several KDEs of the messages. However, it is very difficult to draw particles from this product. Thus, we use the sum of the messages as a *proposal* distribution $q^{i+1}_u(x_u)$, and then reweight all particles. This procedure is well-known as MIS [23].

Denote the set of neighbors of u, having observed edges to u and not including anchors, by G^0_u; and the set of all neighbors by G_u. In order to draw N_p particles, we create a collection of kN_p weighted particles (where $k \geq 1$ is a parameter of the sampling algorithm) by drawing $kN_p/\left|G^0_u\right|$ particles from each message m_{ru} ($r \in G^0_u$), and assigning, to each particle $X^{j,i+1}_u$, a weight equal to the ratio:

$$W^{j,i+1}_u = \frac{\prod_{v \in G_u} m^{i+1}_{vu}(X^{j,i+1}_u)}{q^{i+1}_u(X^{j,i+1}_u)} \tag{14}$$

where the proposal distribution $q_u(x_u)$ is given by:

$$q^{i+1}_u(x_u) = \sum\nolimits_{v \in G^0_u} m^{i+1}_{vu}(x_u) \tag{15}$$

Some of the calculated weights are usually much larger then the rest, especially after several iterations. This means that the particle-based estimate will be dominated by the influence of a few particles, and the estimate could be erroneous. This problem is known as *sample depletion* [19]. To avoid this problem, we draw N_p values independently from the collection $\{W^{j,i+1}_r, X^{j,i+1}_r\}$ with the probability proportional to their weight, using *resampling with replacement* [23]. This means that we create N_p equal-weight particles drawn from the product of all the incoming messages.

A node is located when some convergence criterion is met. We can use Kullback-Leibler (KL) divergence [23], a common measure of the difference between two distributions. For the particle based beliefs in our algorithm, the KL-divergence between beliefs in two consecutive iterations, is given by:

$$KL^{i+1}_u = \sum_j W^{j,i+1}_u \log \frac{W^{j,i+1}_u}{M^i_u(X^{j,i+1}_u)} \tag{16}$$

where we used the approximation $M^{i+1}_u(X^{j,i+1}_u) \approx W^{j,i+1}_u$. The algorithm stops when KL^{i+1}_u drops (for all nodes) below the predefined threshold. However, given the structure of the graph (e.g., communication radius, and diameter of the deployment area), we can predefine the number of iterations so as to decrease the complexity of the algorithm. We use the latter approach in this article.

Finally, it is important to mention that the proposed algorithms (BP and NBP) are not exact in networks with loops. The problem is *double counting* [18], a situation in which the same evidence (as part of the message) is passed around the network multiple times and mistaken for new evidence. This will cause overconfident beliefs of the position estimates. We already considered this problem and proposed a few solutions in static networks [21, 24, 25]. Obviously, the same approaches can be applied for the mobile networks. However, to make the algorithm as fast as NBP, we recommend *uniformly-reweighted* NBP (URW-NBP) [25] as the simplest extension of NBP. In this article, we will consider net-

[2] Approximation of the distribution $p(x)$: $\hat{p}(x) = \sum_j w^j K^h(x - x^j)$ given a kernel $K^h(x)$. The most common kernel function (K^h) is the spherically symmetric Gaussian kernel: $K^h(x) = N(x, 0, hI)$, where h is the bandwidth which controls the variance [9, 26].

works with a negligible number of loops, so NBP will be good enough for the all analyses.

3. Mobile positioning using NBP

From now on, we assume that the target nodes are moving within the deployment area, and that the anchor nodes are still static. Our goal is to adapt the static NBP method to mobile positioning.

3.1. Graphical model

We start with the example of a graphical model, in Fig. 1, which illustrates three target nodes (m, n, and p) in three consecutive time frames ($t-1$, t, and $t+1$). For instance, to locate node n at time t, we need the messages from its neighbors (m and p) as in static NBP, plus two additional messages from the past and the future. Thus, to extend static NBP, we just need to define the pairwise potential and the messages between two consecutive time frames. It is also worth noting that the connectivity between nodes can change over time.

The pairwise potential between two consecutive time frames (for a target node n) $\psi_{t-1,t}(x_{n,t-1}, x_{n,t})$ (we refer to it as *kinematic potential*) represents the correlation between positions in these time frames, which depends on the target dynamic. For example, if we can keep track of the speed v_{t-1} at time $t-1$, and have the model of the distribution p_w of the process noise w (which represents the random variation of the speed due to the acceleration), kinematic potential is given by:

$$\psi_{t-1,t}(x_{n,t-1}, x_{n,t}) = p_w(\left\|x_{n,t} - x_{n,t-1}\right\| - v_{t-1} \cdot T_S). \quad (17)$$

where T_s represents the sampling interval. Note that it is usually hard to measure the process noise, so a Gaussian approximation is a common choice. However, since we prefer to keep the non-Gaussian nature of NBP, we apply the model (as in [15]) which only requires the knowledge of the maximum speed of the target V_{max}. This is usually easy to find for most applications (e.g., 5 m/s for people, 1 m/s for forklifts, 30 m/s for cars, etc.). Given V_{max}, the kinematic potential of node n can be written as:

$$\psi_{t-1,t}(x_{n,t-1}, x_{n,t}) = \begin{cases} 1, & if \ \left\|x_{n,t-1} - x_{n,t}\right\| \leq V_{max} \cdot T_s, \\ 0, & otherwise. \end{cases} \quad (18)$$

Using this potential, we can predict the possible positions of node n at time t, given the estimate at time $t-1$, and vice versa for smoothing. Note that, if we can measure the dynamic of the target (e.g., using an accelerometer or a pedometer), we can use a more informative kinematic potential [28].

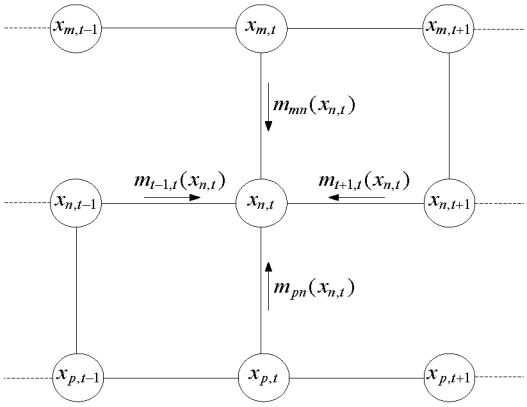


Figure 1: Example of a graphical model for mobile positioning, which illustrates three target nodes (m, n, and p) in three consecutive time frames ($t-1$, t, and $t+1$).

3.2. BP and NBP

Now we can extend the message passing method described in Section 2 for mobile networks. Denote the belief of node n at time t, in the last iteration of static BP (see (9)), as $M^S_{n,t}(x_{n,t})$. To adapt the graphical model to mobile networks (according to Figure 1 and equation (9)), we can write the belief of node n in mobile networks as:

$$M^{FS}_{n,t}(x_{n,t}) = m_{t-1,t}(x_{n,t}) M^S_{n,t}(x_{n,t}) m_{t+1,t}(x_{n,t}) = M^F_{n,t}(x_{n,t}) m_{t+1,t}(x_{n,t}) \quad (19)$$

where $m_{t-1,t}(x_{n,t})$ represents the *filtering message* (the message from past to present), $m_{t+1,t}(x_{n,t})$ the *smoothing message* (the message from future to present), and $M^{FS}_{n,t}(x_{n,t})$ is the belief which includes the filtering and smoothing messages. This belief can be available after N_t time frames (for N_t-lag smoothing). We will focus on 1-lag smoothing[3] which can provide $M^{FS}_{n,t}(x_{n,t})$ at time $t+1$. By excluding $m_{t+1,t}(x_{n,t})$ (i.e., $m_{t+1,t}(x_{n,t}) = 1$), we can also define the filtered belief $M^F_{n,t}(x_{n,t})$, which is available in real-time.

Using the message update rule (10), we can define the filtering message (from $t-1$ to t), and the smoothing message (from $t+1$ to t). They are, respectively, given by:

$$m_{t-1,t}(x_{n,t}) \propto \int_{x_{n,t-1}} \psi_{t-1,t}(x_{n,t-1}, x_{n,t}) \frac{M^{FS}_{n,t-1}(x_{n,t-1})}{m_{t,t-1}(x_{n,t-1})} dx_{n,t-1} \quad (20)$$

[3]Note that s-lag smoothing ($s > 1$) is less useful due to the "accumulation" of the uncertainty through a multi-hop path (even if the future measurements are very informative).

$$m_{t+1,t}(x_{n,t}) \propto \int_{x_{n,t+1}} \psi_{t+1,t}(x_{n,t+1}, x_{n,t}) \frac{M^{FS}_{n,t+1}(x_{n,t+1})}{m_{t,t+1}(x_{n,t+1})} dx_{n,t+1} \tag{21}$$

Using (19), we can simplify previous equations:

$$m_{t-1,t}(x_{n,t}) \propto \int_{x_{n,t-1}} \psi_{t-1,t}(x_{n,t-1}, x_{n,t})\, M^{F}_{n,t-1}(x_{n,t-1}) dx_{n,t-1} \tag{22}$$

$$m_{t+1,t}(x_{n,t}) \propto \int_{x_{n,t+1}} \psi_{t+1,t}(x_{n,t+1}, x_{n,t})\, M^{S}_{n,t+1}(x_{n,t+1}) m_{t+2,t+1}(x_{n,t+1}) dx_{n,t+1} \tag{23}$$

Moreover, since we prefer to use 1-lag smoothing, we discard further information from the future (i.e., we set $m_{t+2,t+1}(x_{n,t+1}) = 1$):

$$m_{t+1,t}(x_{n,t}) \propto \int_{x_{n,t+1}} \psi_{t+1,t}(x_{n,t+1}, x_{n,t})\, M^{S}_{n,t+1}(x_{n,t+1}) dx_{n,t+1} \tag{24}$$

Regarding the nonparametric approximation, we can reuse the weighted particles from the static scenario $\{W^{S,j}_{n,t}, X^{S,j}_{n,t}\}$, and use the filtering and the smoothing message to reweight them:

$$W^{FS,j}_{n,t} = m_{t-1,t}(X^{S,j}_{n,t}) \cdot W^{S,j}_{n,t} \cdot m_{t+1,t}(X^{S,j}_{n,t}) = W^{F,j}_{n,t} \cdot m_{t+1,t}(X^{S,j}_{n,t}) \tag{25}$$

Messages, (22) and (24), can be computed via Monte Carlo integration, i.e.:

$$m_{t-1,t}(x_{n,t}) \propto \sum_j \psi_{t-1,t}(X^{S,j}_{n,t-1}, x_{n,t})\, W^{F,j}_{n,t-1} \tag{26}$$

$$m_{t+1,t}(x_{n,t}) \propto \sum_j \psi_{t+1,t}(X^{S,j}_{n,t+1}, x_{n,t})\, W^{S,j}_{n,t+1} \tag{27}$$

As we can see, this method is very flexible, since there are three different beliefs available: $M^{S}_{n,t}$, $M^{F}_{n,t}$, and $M^{FS}_{n,t}$. The belief $M^{F}_{n,t}$ should be used in real-time applications, while the belief $M^{FS}_{n,t}$ should be used in all applications in which we can afford waiting one more sampling interval (T_S). On the other hand, the belief $M^{S}_{n,t}$ should not be used for tracking, but it is useful for testing the target dynamic[4].

4. Communication protocol

Our second goal is to decrease the communication cost by: i) broadcasting the beliefs (instead of the messages), ii) approximating the beliefs without a significant effect on the localization performance, and iii) avoiding the transmission of uninformative data.

4.1. Broadcasting beliefs

Naturally, one can assume that the messages should be exchanged between each pair of the neighboring nodes. However, this produce a huge communication overhead since each node would need to send one message to each of the neighbors. The main problem of this approach is that it does not take advantage of the broadcast in WSN (i.e., the transmitted message is needed at just one neighbor, not all). If we recall equation (11), we can see that the particles from the messages are constructed using particles from the: i) current belief of the node which transmits the message (source node), ii) measured distance, and iii) random angle. Since the samples of the distance can be measured by each node (prior to localization) and stored into memory, they should not be transmitted. The samples of the angles are drawn from the uniform distribution (see (11)), so they can be computed at the destination. Thus, only particles of the beliefs, which are not available at the destination node, have to be transmitted. Finally, it is also necessary to compute the weights using (12), which requires the outgoing message from the previous iteration. This problem can be solved by computing the messages twice: once at the source node, and once at the destination node.

The main benefit of this approach is that each node has to broadcast only one package[5] instead of n_d packages in case of message transmission (where n_d is the node degree). This is paid by increased computational cost since the messages must be computed twice. However, it is already well-known [1] that the communication is much more energy-consuming than computation. The proposed protocol is summarized in Alg. 1.

4.2. Package approximation

For the described protocol, we would need to transmit N_p particles (i.e., N_p weights, and $2N_p$ coordinates). However, we can avoid this using the following approximations:

- We resample with replacement in order to avoid the transmission of the weights[6].

[4] For example, it can be used to learn V_{max} (if not known in advance).

[5] To avoid confusion, we use the term "packages" for the data that will be transmitted, in contrast to the term "messages" which refers to NBP messages, which are never transmitted. Although one package usually includes more scalar values (depending on the hardware platform), in order to simplify the analysis, we assume that it includes just one scalar value.

[6] Note that we can also avoid the transmission of the bandwidth, but it would only slightly decrease the communication cost.

Algorithm 1 Communication protocol at time t (without approximation and censoring)

```
for all nodes do
    Obtain sufficient number of distance samples (from each neighbor)
    Initialize all belief and messages
    for all iterations do
        Compute particles from outgoing messages and reweight them
        Broadcast particles of the current belief
        Compute particles from income messages and reweight them
    end for
end for
```

- We approximate the unweighted set of particles with a Gaussian mixture (GMM), and transmit only their parameters.
- Upon receiving the GMM parameters, we re-draw the set of particles from this mixture.

Since resampling is already part of the NBP algorithm (Section 2.3.2), it does not affect accuracy. Regarding the GMM approximation, we expect that (given sufficient mixture components), it will not affect significantly the localization performance. Since the main problem of cooperative localization is the presence of the multi-modal beliefs (caused by non-rigid graphs and/or multi-modal measurement noise), we expect a GMM with very few components (N_m) to be the appropriate choice. It is preferable to set N_m to a slightly larger value than the expected number of modes, and then remove all the components with small weights. We can cluster the unweighted set of particles using the k-means algorithm [29], or expectation-maximization (EM) [30]. The latter one can provide slightly better results, but with higher complexity. Thus, we recommend the use of k-means, especially for mobile networks.

4.3. Package censoring

We can additionally decrease communication using package censoring, i.e., by avoiding the transmission of packages which provide little information. To that end, we do the following:

- In the first iteration, we only transmit the bounds of the bounded box (i.e., 4 scalar values, which define the rectangle). Then, the particles can be drawn at the destination node.
- We do not transmit beliefs in the last iterations since they will never be used to update messages.
- Packages from anchor nodes are never transmitted (except their coordinates, if not known in advance).
- We do not transmit beliefs at iteration i which are very similar to the beliefs in iteration $i-1$. The similarity can be measured using the KL divergence.

We expect that these steps will, without any effect on the accuracy, significantly decrease the communication cost. Note that we can also avoid receiving packages, as done in [31]. This technique should be applied if receiving the data is energy-consuming, and also in the case of single-cast communication.

5. Sampling techniques

Our final goal is to improve the sampling techniques used in the standard NBP. We propose three techniques: i) mixture importance sampling with reference particles (MIS-RP), ii) Population Monte Carlo (PMC), and iii) the method based on an auxiliary variable.

5.1. MIS with reference particles

The MIS technique defined by (15) usually provides a very good set of particles, and outperforms a number of techniques as shown in [23, 32]. However, this might not be the case in the presence of the huge outliers (e.g., if obstacles are moving around).

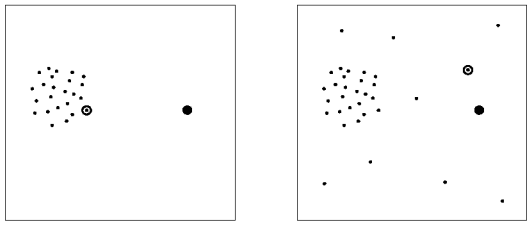
According to the results in mobile robot localization [33], it is always useful to add a small number of uniformly distributed particles. These particles are essential for re-localization in the rare event that the sensor loses track of its position. We call these additional particles, *reference particles* (RP). In our case, this will especially happen if the messages from the neighboring target nodes provide wrong particles, but either anchor nodes or the kinematic message provide good weights. The problem is illustrated in Figure 2. Without RP (Figure 2a), MIS provides the set of particles in which the best candidate is very far from the true position (e.g., due to the outliers). With RP (Figure 2b), additional particles have been added uniformly in the whole area, so the best candidate is closer[7] to the true position.

Therefore, the importance density (at iteration $i+1$) can be written in the form:

$$q_u^{i+1}(x_u) = \sum\nolimits_{v \in G_u^0} m_{vu}^{i+1}(x_u) + \delta_{RP} \cdot Unif(x_u) \qquad (28)$$

where $Unif(x_u) \propto 1$ within deployment area, $Unif(x_u) = 0$ otherwise, and δ_{RP} is the weight of the uniform distribution (in other words, the percentage of

[7] To simplify the example, we assumed the MAP estimate, but the same conclusion is valid for the MMSE estimate.



(a) (b)

Figure 2: Possible positions of target nodes in the case of (a) MIS, and (b) MIS with reference particles (MIS-RP). The true position of the target node is marked with a black circle, and the best particle candidates are encircled.

Algorithm 2 Population Monte Carlo (PMC) (for node u)

1: Choose initial importance function: $q_u^1(x_u)$
2: **for all** iterations $m = 1 : N_{PMC}$ **do**
3: Draw particles: $X_u^{j,m} \sim q_u^m(x) \quad (j = 1...N_p)$
4: Compute weights: $W_u^{j,m} = \frac{p(X_u^{j,m})}{q_u^m(X_u^{j,m})}$
5: Normalize weights: $W_u^{j,m} = \frac{W_u^{j,m}}{\sum_j W_u^{(m)}}$
6: Resample with replacement
7: Update importance density: $q_u^{m+1}(x_u) = \sum_j \mathcal{K}_h(x_u - X_u^{j,m})$
8: **end for**

reference particles). δ_{RP} should be small (e.g., 10-20%) in order to keep the computational cost reasonable. Note that this importance density is legitimate since it is non-zero at places where the distribution, that is being approximated, is non-zero. Thus, in case of regular situations (when messages provide good particles), these additional particles will not cause any problem (i.e., after reweighting, their weights will be close to zero).

5.2. Population Monte Carlo (PMC)

PMC [34, 35] is an iterative importance sampling technique in which the importance density changes with every iteration in order to produce particles that better represent the target distribution. The standard importance sampling technique is a special case of PMC by running just one iteration. The general form of PMC is illustrated in Alg. 2.

In order to use PMC for cooperative localization, we need to choose the importance density that we want to improve. Thus, we can choose the density used for MIS (given by (15)) or MIS-RP (given by (28)) as prior. Distribution $p(X_u^{j,m})$ used for reweighting in Alg. 2 is given by the numerator of (14). For the KDE, we again use a spherical Gaussian Kernel with bandwidth h. Finally, in each iteration of Alg. 2, we draw a new set of particles from the KDE as follows:

$$X_u^{j,m} = X_u^{j,m-1} + \epsilon_u^j \cdot [\cos(\theta_u^j) \quad \sin(\theta_u^j)] \qquad (29)$$

where $\epsilon_u^j \sim N(0, h)$ and $\theta_u^j \sim Unif[0, 2\pi)$. Note that we used simplified notation by removing the NBP iteration index (do not mix NBP iterations with PMC iterations). We refer to this version of NBP, as PMC-NBP.

Finally, we propose an optional approximation of the PMC-NBP method. The main computational problem of NBP and variations is the computation of KDE, which requires $O(N_p^2)$ operations. This is especially the problem in PMC-NBP, since it has nested iterations (i.e., within one NBP iteration, there are N_{PMC} iterations). Therefore, instead of using full information (the product of the messages from all the neighbors), we use only information from the anchors. In order to keep the NBP algorithm regular, we just need to keep the full information in the first iteration of the PMC (which corresponds to the standard importance sampling). Since we do not use the information from the target nodes, this method represents a *non-cooperative* PMC. Note that this approach will not only improve the beliefs of the anchors' neighbors. Since NBP is still a cooperative method, in NBP's very next iteration, the improved estimate of the anchors' neighbors will be flooded further into the network.

5.3. Auxiliary variable

Standard importance sampling used in NBP does not take into account most of the available information in the graph. This often causes high variance of the weights, i.e., there will be a lot of particles in the regions of low probability, and very few (or even just one) in the regions of high probability. One solution to this problem is to use the optimal importance density, which includes all the available information, but this is not feasible in most cases [19]. A second solution is PMC from the previous section, which iteratively improves the importance density. The auxiliary particle filtering (APF) [36, 37] is an alternative solution which tries to predict (using an auxiliary variable) which particles will be in regions of high probability.

However, NBP is a generalization of particle filtering for general graphs, so we need to adapt the standard APF method. One framework has been already proposed in [38], in which the authors propose to use the index of the messages as an auxiliary variable in order to predict which message provides better information. This method is not very suitable (especially, for localization) due to the high dimensional auxiliary variable. In contrast to this approach, we will choose a 1D auxiliary variable, which will provide the largest amount of information.

Recall equation (11), written in a more general form:

$$x_{ru}^{j} = X_r^{j_X} + (d_{ru} + v^{j_d})[\cos(\theta^{j_\theta}) \;\; \sin(\theta^{j_\theta})] \quad (30)$$

in which we again removed the NBP iteration index and, in contrast to (11), we use a different index ($j, j_X, j_d, j_\theta \sim 1...N_p$) for each random variable. As we can see, to update the particles of the messages (x_{ru}^{j}), we need to use three random variables: particles of the current position ($X_r^{j_X} \sim M_r(x_r)$), distance samples ($d_{ru} + v^{j_d} \sim p_v$), and angle samples ($\theta^{j_\theta} \sim Unif[0, 2\pi)$). The auxiliary variable could be an index of any of these three random variables. Obviously, the uniformly distributed θ^{j_θ} includes the smallest amount of information (the entropy is maximal) than any other random variable, so we choose j_θ as the auxiliary variable. Then, we can set the other indices to the same value ($j_X = j_d = j$). In other words, instead of drawing samples uniformly in any direction (which will create a lot of particles with small weights), we will draw them in the most likely direction according to the distribution of the auxiliary variable. To achieve this, we first, for each index j_θ, find some likely value associated with the message, e.g., expected value:

$$\mu_{ru}^{j_\theta} = \mu_{X_r} + \mu_{d_{ru}}[\cos(\theta^{j_\theta}) \;\; \sin(\theta^{j_\theta})] \quad (31)$$

where we averaged the left-hand side of (30) over j. The computed set of mean values is further used to compute the *first-stage* (1st) weights:

$$w_{ru}^{j_\theta,1st} \propto p(Y|\mu_{ru}^{j_\theta}) \quad (32)$$

which represent the likelihood function, given all the information (Y) that we want to include. Recall that these weights are for the message from node r to node u, so they represents some information about the position of node u. Thus, we can include the product of all the messages coming to node u, but as for PMC, we again restrict to information from the anchors. For this approach, it is even more critical because the number of messages (equal to twice the number of the edges in the graph) is typically significantly larger than the number of target nodes. Therefore, the likelihood of the information that we want to include is given by:

$$p(Y|\mu_{ru}^{j_\theta}) = \prod\nolimits_{a\in G_a} p(d_{au}|x_a^*, \mu_{ru}^{j_\theta}) \propto \prod\nolimits_{a\in G_a} \psi_{au}(x_a^*, \mu_{ru}^{j_\theta}) \quad (33)$$

where G_a is the set of all the anchor neighbors of node u. In the case of no anchors in the neighborhood, we simply do not apply this approach. First-stage weights provide the information on how likely is the index of the angle j_θ. Therefore, given the multinomial distribution defined by the first-stage weights, we can draw the set of N_p indices $ind(j_\theta)$ ($j_\theta = 1...N_p$). Then, we can compute particles from the message:

$$x_{ru}^{j_\theta} = X_r^{j_\theta} + (d_{ru} + v^{j_\theta})[\sin(\theta^{ind(j_\theta)}) \;\; \cos(\theta^{ind(j_\theta)})] \quad (34)$$

Finally, the whole procedure is still not regular due to the double-counting of the information from anchors (which is regularly used in (14)). Thus, the regular weights, given by (12), should be divided by the weights of the importance density given by the first-stage weights (32). The final weights for the particles from the message, also called the *second-stage* weights [36] are given by:

$$w_{ru}^{j_\theta} = \frac{W_r^{j_\theta}}{m_{ur}(X_r^{j_\theta})} \cdot \frac{1}{p(Y|\mu_{ru}^{ind(j_\theta)})} \quad (35)$$

Given these weights, we can proceed with the standard NBP. We refer to this version as the *auxiliary* NBP (ANBP). It is worth noting that the standard NBP is a special case of ANBP, if no additional information (Y) has been used, i.e., when $p(Y|\mu_{ru}^{j_\theta}) \propto 1$, and $p(\theta^{ind(j_\theta)}) = p(\theta^{j_\theta}) \propto Unif[0, 2\pi)$.

6. Simulation Results

We conducted several simulations using Matlab, in order to analyze the performance of the NBP method in mobile networks, the effect of package approximation and censoring, and the effect of the improved sampling techniques.

6.1. Analysis of mobile positioning based on NBP

In the first set of tests, we assume that there are $N_a = 16$ anchor nodes[8] and $N_t = 5$ target nodes, deployed in a 100m x 100m area. Anchor nodes are deployed in a grid, or semi-random topology. The latter means that the area is divided into N_a square-shaped cells, and each anchor node is deployed randomly within one of them[9]. Target nodes are moving according to the Gaussian-Markov mobility model [39]. This model is using one tuning parameter to vary the degree of randomness of the movement, and can easily ensure that the target is always within the deployment area. For this test, we set: number of particles $N_p = 400$, communication radius $R = 20$m, and standard deviation of the zero-mean Gaussian noise for the measured distance $\sigma_d = 1$m. Other parameters (which will be the same for all the simulations) are summarized in Table 1.

In Figures 3a-3c, we show an example of estimated tracks for the three different NBP estimates. As we can

[8] In order to compare different deployment strategies in this example, we use significantly more anchors than usual.

[9] The purpose of this model is to assume that the anchors cannot be perfectly deployed.

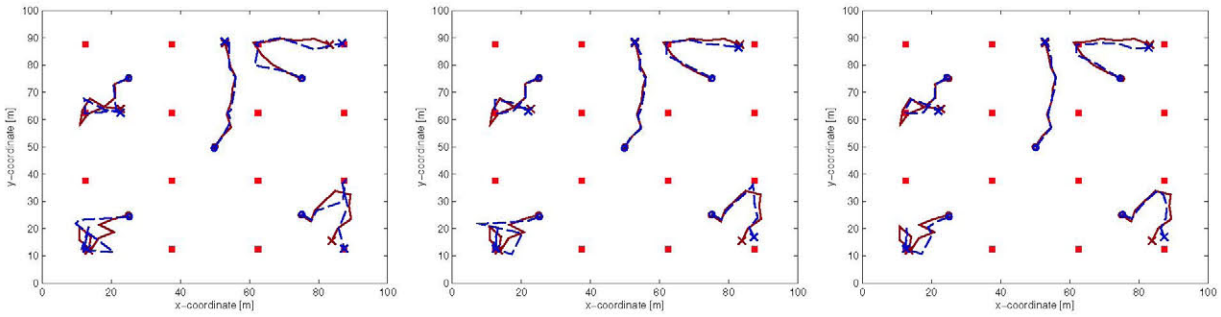


Figure 3: Illustration of 5 tracks and their estimates found using: (a) $M_{n,t}^{S}$, (b) $M_{n,t}^{F}$, and (c) $M_{n,t}^{FS}$. Anchors are marked with squares, true track with lines, and estimated track with dashed lines (starting points of the tracks are marked with circles, and destination points with 'X').

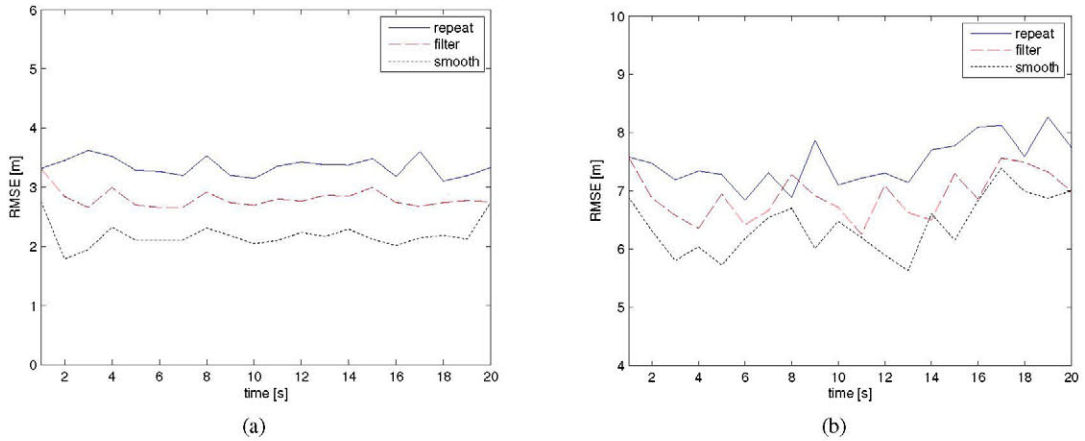


Figure 4: Comparison of the RMSE in the case of: (a) grid, and (b) semi-random topology of the anchors.

Table 1: Parameters for simulations.

Parameter	**value**
sampling interval (T_S)	1s
max. speed (V_{max})	5m/s
tracking period (T_P)	20s
number of NBP iterations (N_{iter})	3
number of Monte Carlo runs (N_{MC})	100

see, all the estimates are similar in the case of a sufficient number of neighbors, but the smoothed estimate provides the best result for the tracks close to the edges. We also compare root mean square errors (RMSE) of all the three methods for three different deployments (Figure 4). As expected, the smoothed estimate consistently performs better than the filtered estimate, which performs better than naive repeating of the static NBP localization method. Note that the smoothed (1-lag) estimate is available 1 second after the filtered estimate, but this delay should not be a problem for most applications. On the other hand, we can see that the deployment of the anchor nodes significantly affects accuracy. Thus, it is advisable to use the grid deployment. However, if it is not possible, this problem can be solved (with an additional cost) either by increasing the number of anchors or by increasing the communication radius.

6.2. Analysis of package approximation and censoring

In contrast to previous tests, we change the number of nodes ($N_a = 5$, $N_t = 10$), and the communication radius ($R = 40$m). The anchors are deterministically placed (4 near the edges, and one in the center). To make the analysis more general, we add an outlier component (25% of the true distance) to the zero-mean Gaussian noise. More precisely, the noise is a two-component Gaussian mixture with the same weights ($w_{d,1} = w_{d,2} = 0.5$) and the same standard deviations ($\sigma_{d,1} = \sigma_{d,2} = 1$m), but different means ($\mu_{d_{ru},1} = 0$, $\mu_{d_{ru},2} = 0.25 d_{ru}$). This noise

Table 2: Number of transmitted packages (N_{pack}) for different protocols and approximations.

Package	N_{pack}
message	6144
belief (Alg. 1)	2400
belief (5-mix approx.)	72
belief (5-mix approx. and censoring)	45

is applicable to the scenario in which there is an obstacle (between two sensors) for 50% of the time. Moreover, taking into account the conclusion from the previous section, we use the smoothed estimate of the NBP.

We start by analyzing the KL divergence (KLD) between the approximated and the particle-based belief w.r.t. the number of mixture components N_m. According to Figure 5, we can see that KLD is decreasing as we increase N_m, as expected. In the case of Gaussian noise, we just need 2 or 3 mixture components, since Gaussian noise does not necessarily lead to Gaussian posteriors due to the nonlinearity (especially emphasized in non-rigid graphs). However, if we add an outlier, we need a few more mixture components. To see how this approximation affects the error, we analyze the cumulative distribution function (CDF) for different approximations of the beliefs (particle-based, and beliefs represented with $N_m = 1$, $N_m = 3$, $N_m = 5$ mixture components). We consider the case with an outlier since it is the most critical case. As we can see in Figure 6, the 5-mixture approximation provides almost the same accuracy as the particle-based approximation. However, the number of mixture components should be a tuning parameter, which will allow the user to make the trade-off between accuracy and cost. Note also that the package censoring proposed in Section 4.3 does not affect accuracy at all, assuming that the KLD threshold (used for measuring the similarity between beliefs) is sufficiently small (less than 0.2, in our case).

Finally, we analyze the communication cost per node within one time instant. According to Table 2, we can conclude that broadcasting the particles from beliefs decrease the cost 2.56 times ($n_d \approx 2.56$, excluding anchors). Moreover, the GMM approximation significantly decreases the communication cost (97%) as expected. Finally, package censoring also decreases the communication (37% in our case), especially because of the savings in the first and the last iteration. In the case of more iterations (e.g., in large-scale networks with small R), the benefit of the first and the last iteration will be less significant. However, additional savings are expected due to the more frequent occurrence of similar beliefs between two consecutive iterations.

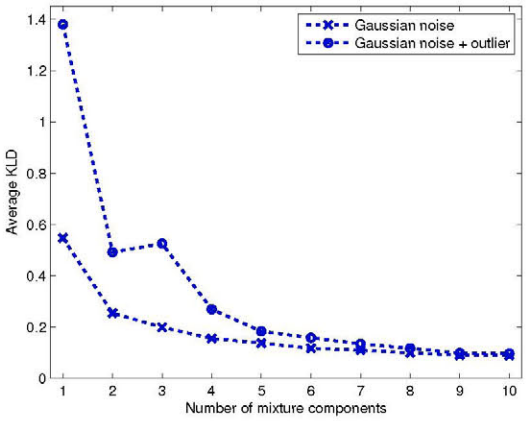


Figure 5: Comparison of KLD between approximated belief and particle-based belief.

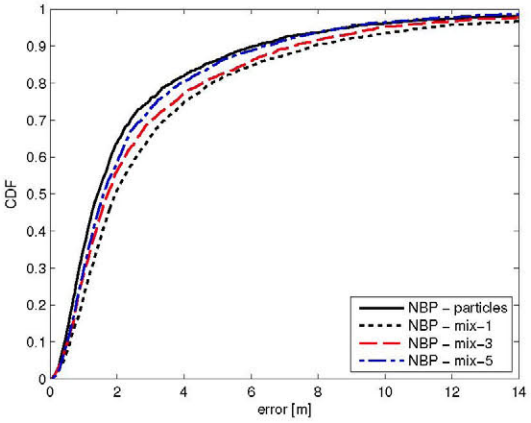


Figure 6: Cumulative distribution function (CDF) of the position error for different approximations.

For the further analysis, we will assume that each belief is approximated by a GMM of 5 components, since it practically has no effect on accuracy.

6.3. Analysis of sampling techniques

We start with the comparison between the MIS and MIS-RP techniques. We consider the same scenario as in Section 6.2 (with outliers). For MIS-RP, we added 20% reference particles, which are uniformly distributed within the deployment area. According to Figure 7, NBP with the MIS-RP technique consistently outperforms NBP with the MIS technique. For example, 90th percentile of MIS-RP is about 20% more accurate than MIS. We also compared RMSE w.r.t. the number of particles. As we can see in Figure 8, the MIS-RP technique provides about 10% more accurate estimates than MIS. More importantly, by applying MIS-RP we can achieve the same error (e.g., 5m) using 15-25% fewer particles. This practically compensates for the previously added

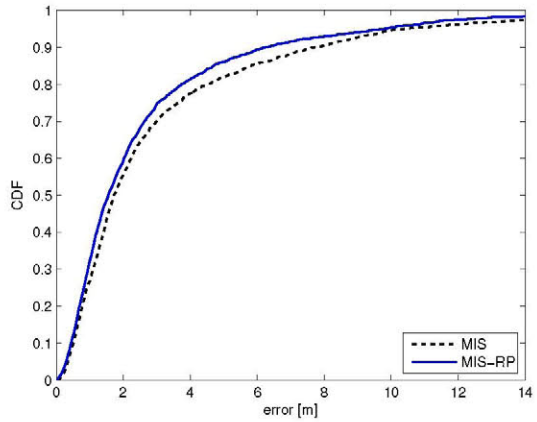


Figure 7: Cumulative distribution function (CDF) of the position error for MIS and MIS-RP (400 particles used).

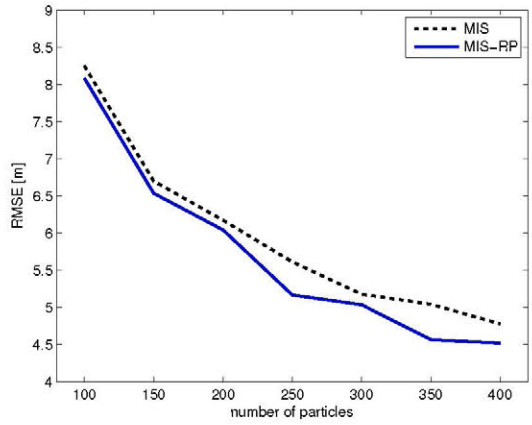


Figure 8: RMSE for MIS and MIS-RP w.r.t. number of particles.

reference particles. It is also worth to mention that MIS-RP performs similarly to MIS if there are no outliers. In any case, it is strongly recommended to use MIS-RP in order to increase the robustness of the NBP method.

We now provide the comparison between the NBP, PMC-NBP, and ANBP methods. We again consider the same scenario as in the previous section, but without outliers. For PMC-NBP, we found that it is sufficient to use 5 PMC iterations. In Figure 9, we compare the CDF for all three techniques. As expected, PMC-NBP, and ANBP provide more accurate estimates. Moreover, both methods (PMC-NBP and ANBP) provide nearly the same estimates. This is expected since we used the same information to improve the importance density (information from the anchors). In Figure 10, we provided a comparison of the RMSE w.r.t. the number of particles. We can see that the benefit of PMC-NBP and ANBP can be about 30% if we use a small number of particles (< 200, in this case). Moreover, ANBP can outperform PMC-NBP for a small number of particles.

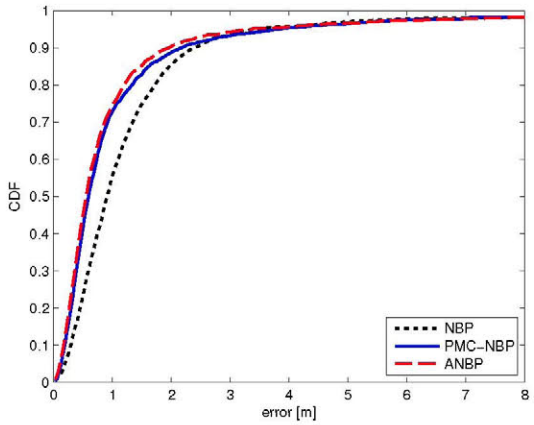


Figure 9: CDF of the position error for NBP, PMC-NBP and ANBP methods (400 particles used).

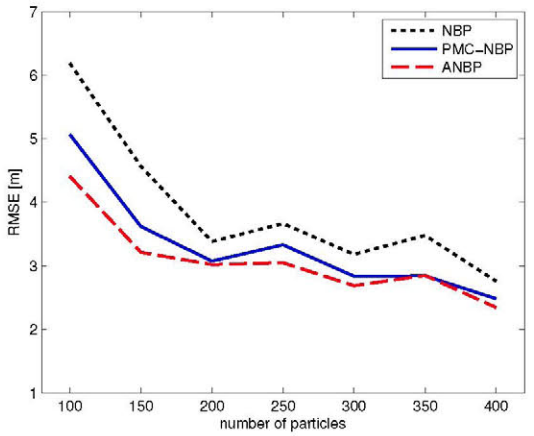


Figure 10: RMSE for NBP, PMC-NBP and ANBP methods, w.r.t. number of particles.

We also note that if we need to achieve a predefined accuracy (e.g., 3m), we need to use significantly fewer particles (15-30%, in our case). That means that PMC-NBP/ANBP are more efficient than standard NBP.

Finally, the main question is which method should be applied (PMC-NBP or ANBP) since both of them provide similar performance. The communication cost of both PMC-NBP and ANBP is the same as the NBP cost, since all modifications can be done locally (see Sections 5.2 and 5.3). However, taking into account that ANBP tries to improve particles from the messages, and PMC-NBP particles from the beliefs, the latter one is less complex (assuming a small number of PMC iterations). Therefore, PMC-NBP should be applied for low-cost applications. Otherwise, ANBP should be used, since it is slightly more accurate than PMC-NBP (see Figure 10).

7. Conclusions and Future Work

As presented in this article, novel variants of NBP, for cooperative localization in mobile networks, outperform the standard NBP method in terms of accuracy and communication. Our main contributions are: i) optional 1-lag smoothing done almost in real-time, ii) novel communication protocol, and iii) improved sampling techniques. One drawback is a higher computational cost, but taking into account the significantly decreased communication cost, the total cost is significantly decreased. According to our results, PMC-NBP is the most promising solution, since it is up to 30% more accurate than NBP. It can be used in low-cost applications because of the very low communication cost. For future research, one can try to analyse the effect of high-frequency tracking (i.e., with very small T_S) when there is no time to execute static NBP within each time instant. Moreover, it is also possible to combine ANBP and PMC-NBP (i.e., to use PMC-ANBP) and try to decrease their complexity. Finally, this method can be used for some real application, e.g., for search-and-rescue or cooperative vehicle tracking.

Acknowledgments

Vladimir Savic is supported by the FPU fellowship from Spanish Ministry of Science and Innovation; This work was supported in part by the Spanish Ministry of Science and Innovation under the grants TEC2009-14219-C03-01 and TEC2010-21217-C02-02-CR4HFDVL; program CONSOLIDER-INGENIO 2010 under the grant CSD2008-00010 COMONSENS; and the European Commission under the grant FP7-ICT-2009-4-248894-WHERE-2; The authors would also like to thank Pavle Belanovic for many useful comments.

References

[1] N. Patwari, J.N. Ash, S. Kyperountas, A.O. Hero III, R.L. Moses and N.S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *Signal Processing Magazine, IEEE*, vol. 22, issue 4, pp. 54-69, July 2005.

[2] C.-Y. Chong and S.P. Kumar, "Sensor networks: evolution, opportunities, and challenges", *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1247- 1256, August 2003.

[3] A. H. Sayed, A. Tarighat, and N. Khajehnouri, "Network-based wireless location: challenges faced in developing techniques for accurate wireless location information", *Signal Processing Magazine, IEEE*, vol. 22, no. 4, pp. 24-40, July 2005.

[4] N.I. Dopico, B. Bejar, S.V. Macua, P. Belanovic, and S. Zazo, "Improved animal tracking algorithms using distributed Kalman-based filters," in *IEEE Proc. of 11th European Wireless Conf.*, pp.1-8, April 2011.

[5] B. Parkinson, and J.J. Spilker Jr., *Global Positioning System: Theory and Application.* Volume I, Progress in Astronautics and Aeronautics, Volume 163, 1996.

[6] Y. Shang, W. Ruml, and M. Fromherz, "Positioning using local maps," in *Ad-Hoc Networks - Elsevier*, vol. 4, issue 2, pp. 240-253, March 2006.

[7] A. Savvides, H. Park, and M. B. Srivastava, "The bits and flops of the n-hop multilateration primitive for node localization problems," in *Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Application*, pp. 112-121, September 2002.

[8] V. Vivekanandan and V.W.S. Wong, "Concentric anchor beacon localization algorithm for wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 56, issue 5, pp. 2733 - 2744, September 2007.

[9] A.T. Ihler, J. W. Fisher III, R. L. Moses and A. S. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE Journal On Selected Areas In Communications*, vol. 23, issue 4, pp. 809-819, April 2005.

[10] R. Peng, M.L. Sichitiu, "Robust, probabilistic, constraint-based localization for wireless sensor networks," in *IEEE Proc. of Sensor and Ad Hoc Communications and Networks - SECON*, pp. 541-550, 26-29 Sept. 2005.

[11] G. Wang and K. Yang, "A new approach to sensor node localization using RSS measurements in wireless sensor networks," in *IEEE Trans. of Wireless Communications*, vol. 10, no. 5, pp. 1389-1395, May 2011.

[12] M. Huang, S. Chen, Y. Wang, "Minimum cost localization problem in wireless sensor networks," in *Ad-Hoc Networks - Elsevier*, vol. 9, issue 3, pp. 387-399, May 2011.

[13] H.J. Rad, A. Amar, and G. Leus, "Cooperative mobile network localization via subspace tracking," in *IEEE Proc. of ICASSP*, pp. 2612-2615, May 2011.

[14] J. Yi , S. Yang and H. Cha, "Multi-hop-based Monte Carlo localization for mobile sensor networks," in *IEEE Proc. of the Sensor and Ad Hoc Communications and Networks - SECON*, pp. 162-171, June 2007.

[15] A. Baggio and K. Langendoen, "Monte Carlo localization for mobile wireless sensor networks," in *Ad-Hoc Networks - Elsevier*, vol. 6, issue 5, pp. 718-733, July 2008.

[16] H. Wymeersch, J. Lien and M.Z. Win, "Cooperative localization in wireless networks," in *Proc. of IEEE*, vol. 97, issue 2, pp. 427-450, Feb. 2009.

[17] J. Schiff, E.B. Sudderth, K. Goldberg, "Nonparametric belief propagation for distributed tracking of robot networks with noisy inter-distance measurements," in *IEEE Proc. of International Conference on Intelligent Robots and Systems*, pp. 1369-1376, Oct. 2009.

[18] J. Pearl, *Probabilistic Reasoning in Intelligent Systems. Networks of plausible inference.* Morgan Kaufmann, 1988.

[19] M. S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, "A Tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking", *IEEE Transactions on Signal Processing*, vol. 50, issue 2, pp. 174-188, February 2002.

[20] P.M. Djuric, J.H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M.F. Bugallo and J. Miguez, "Particle filtering," *IEEE Signal Processing Magazine*, vol. 20, issue 5, pp. 19-38, Sept. 2003.

[21] V. Savic, A. Poblacion, S. Zazo, and M. Garcia, "Indoor positioning using nonparametric belief propagation based on spanning trees," *EURASIP Journal on Wireless Communications and Networking*, 2010.

[22] M. A. Caceres, F. Penna, H. Wymeersch, and R. Garello, "Hybrid cooperative positioning based on distributed belief propagation," *IEEE Journal on Selected Areas in Communications*, pp. 1948-1958, 2011.

[23] A.T. Ihler, *Inference in Sensor Networks: Graphical Models and Particle Methods*, Thesis, MIT, Department of Electrical Engineering and Computer Science, June 2005.

[24] V. Savic and S. Zazo, "Pseudo-junction tree method for cooperative localization in wireless sensor networks", in *IEEE Proc. of Information Fusion*, July 2010.

[25] V. Savic , H. Wymeersch , F. Penna and S. Zazo, "Optimized edge appearance probability for cooperative localization based on tree-reweighted nonparametric belief propagation," in *IEEE*

Proc. of ICASSP, pp. 3028-3031, May 2011.

[26] B.W. Silverman, *Density Estimation for Statistics and Data Analysis*, Chapman and Hall, New York, 1986.

[27] Z.I. Botev, "A Novel nonparametric density estimator," Technical Report, The University of Queensland, Australia, 2006.

[28] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.J. Nordlund, "Particle filters for positioning, navigation and tracking," in *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425-437, Feb. 2002.

[29] K. Krishna and M. N. Murty, "Genetic K-means algorithm," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 29, no. 3, pp. 433-439, June 1999.

[30] C.M. Bishop, *Pattern Recognition and Machine Learning* , Springer, 2006.

[31] K. Das and H. Wymeersch, "Censored cooperative positioning for dense wireless networks," in *IEEE Proc. of International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 262-266, Sept. 2010.

[32] A.T. Ihler, E.B. Sudderth, W.T. Freeman and A. S. Willsky, "Efficient multiscale sampling from products of Gaussian mixtures," in *Proc. of Neural Information Processing Systems (NIPS)*, 2003.

[33] D. Fox, W. Burgardy, F. Dellaert and S. Thrun, "Monte Carlo localization: Efficient position estimation for mobile robots," in *IEEE Proc. of Sixteenth National Conference on Artificial Intelligence (AAAI)*, pp. 343-349, 1999.

[34] O. Cappe, A. Guillin, J.M. Marin, and C.P. Robert, "Population Monte Carlo," in *Journal of Computational and Graphical Statistics*, vol. 13, pp. 907929, 2004.

[35] M.F. Bugallo, M. Hong, and P.M. Djuric, "Marginalized population Monte Carlo," in *IEEE Proc. of ICASSP*, pp. 2925-2928, 2009.

[36] M.K. Pitt, and N. Shepard, "Filtering via simulations: Auxiliary particle filters," *Journal of the American Statistical Association*, Vol. 94, No. 446., pp. 590-599, 1999.

[37] A. Doucet and A.M. Johansen, "A tutorial on particle filtering and smoothing: fifteen years later," in *The Oxford Handbook of Nonlinear Filtering*, Oxford University Press, 2011.

[38] M. Briers, A. Doucet and S.S. Singh, "Sequential auxiliary particle belief propagation," in *IEEE Proc. of 8th International Conference on Information Fusion*, pp. 705-711, 2005.

[39] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," in *Wireless Communication and Mobile Computing Journal*, vol. 2, no. 5, pp. 483-502, 2002.