2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

# Distributed Nonlinear Estimation for Robot Localization using Weighted Consensus

Andrea Simonetto, Tamás Keviczky and Robert Babuška

*Abstract*— **Distributed linear estimation theory has received increased attention in recent years due to several promising industrial applications. Distributed nonlinear estimation, however is still a relatively unexplored field despite the need in numerous practical situations for techniques that can handle nonlinearities. This paper presents a unified way of describing distributed implementations of three commonly used nonlinear estimators: the Extended Kalman Filter, the Unscented Kalman Filter and the Particle Filter. Leveraging on the presented framework, we propose new distributed versions of these methods, in which the nonlinearities are locally managed by the various sensors whereas the different estimates are merged based on a weighted average consensus process. The proposed versions are shown to outperform the few published ones in two robot localization test cases.**

## I. Introduction

Driven by the numerous potential applications in sensor networks, distributed estimation techniques have been studied with growing interest in the past few years. So far, the research has been mainly focused on state estimation of linear dynamical systems. References [1], [2], [3] and [4] give a comprehensive overview of the field.

There are many situations in which such a framework cannot be applied, due to the nonlinearities in the dynamical system or in the sensing equation or both. An important example of these scenarios is the localization of a moving object via range-only measurements. This particular problem arises in applications such as indoor robot localization [5], [6], [7], underwater sensor networks [8], [9], [10] and space exploration [11] among others.

Although there are a few cases in the literature in which distributed nonlinear estimation has been addressed, a clear performance evaluation is still missing. For instance, in [3] a distributed Extended Kalman Filter is suggested, but not implemented, and in [12] a distributed Particle Filter is proposed without extensive analysis.

The aim of this paper is to propose effective distributed algorithms for nonlinear estimation. To achieve this, we introduce a unified framework, in which we design versions of both distributed Extended Kalman Filters and distributed Particle Filters that show better performance in simulation with respect to the aforementioned literature. Moreover, we propose an algorithm for the distributed Unscented Kalman filter. The core of the framework is a merging mechanism based on a weighted consensus procedure similar to the ones developed in [13].

The authors are with the Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands {a.simonetto, t.keviczky, r.babuska}@tudelft.nl

The paper is organized as follows: Section II defines the problem, in Sections III and IV the merging mechanism and the estimation algorithms are proposed, Section V presents two test cases and in Section VI the conclusions are drawn.

## II. Problem Formulation

### A. Notation

The state variable is denoted by $x \in \mathbb{R}^n$, $u \in \mathbb{R}^{n_u}$ is the control, $z \in \mathbb{R}^{n_z}$ the measurement, $w \in \mathbb{R}^{n_w}$ and $v \in \mathbb{R}^{n_v}$ two independent zero mean Gaussian noise terms. $\hat{x}$ represents the estimate of $x$. $x(k)$ is the state at time instant $k$, and $u(k)$ is the control at the same time instant; $z_i(k)$ is the measurement vector of sensor $i$, at time $k$. For all the other variables the same notation holds. If $a_i(k)$ is a generic vector variable associated with sensor $i$, at time $k$, and $i = 1, \ldots, N$, then $\mathbf{a}(k) = (a_1^T(k), \ldots, a_N^T(k))^T$ is a stacked vector of all the sensor variables.

### B. Distributed estimation

Let the Nonlinear Time-Invariant dynamical model of the system with state $x$ be:

$$x(k+1) \quad = \quad f(x(k), u(k), w(k)) \tag{1}$$

where $f$ is a nonlinear function. Let the process be observed by $N$ sensors each with some processing and communication capability. The sensors are labeled $i = 1, \ldots, N$ and form the set $\mathcal{V}$. The communication topology is modeled as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where an edge $(i, j)$ is in $\mathcal{E}$ if and only if node $i$ and node $j$ can exchange messages. Let the graph be connected, let the sensor clocks be synchronized and assume perfect communication (no delays or packet losses). The nodes with which node $i$ communicates are called neighbors and are contained in the set $\mathcal{N}_i$. Note that node $i$ is not included in the set $\mathcal{N}_i$. We define $\mathcal{J}_i = \mathcal{N}_i \cup \{i\}$ and $N_i = |\mathcal{J}_i|$. Let the measurement equation for each sensor $z_i(k)$ be:

$$z_i(k) = g(x(k)) + v_i(k), \qquad i = 1, \ldots, N \tag{2}$$

where $v_i(k)$ is a Gaussian noise term and $g$ a nonlinear function. In the distributed setting each sensor estimates the state, and $\hat{x}_i(k)$ denotes the estimate of sensor $i$ at time $k$. Therefore, the distributed estimation problem can be formulated as follows. Allowing communication only within the neighborhood, estimate $N$ different copies of the state $\hat{x}_i(k)$ so that the following requirements are satisfied:

R1) each $\hat{x}_i(k)$ is an unbiased estimate of $x(k)$ at each time step $k$;
R2) for $k \to \infty$, all the $\hat{x}_i(k)$'s have the same value.

In this paper, we will restrict ourselves to algorithms that do not exchange raw measurements among the neighbors; since nonlinear filters require more computational effort than linear ones. This restriction is a positive feature because it decreases the communication time and gives more time for computation. This also implies that the structure of our algorithms will consist of $N$ local filters and a distributed merging mechanism which aggregates the different estimates.

*Remark 1:* The graph $\mathcal{G}$ is assumed to be fixed in time. However, under weak technical conditions, we could extend the algorithms to situations in which the graph $\mathcal{G}$ is time-varying, see for instance [13].

### C. Distributed localization

Distributed localization via range-only measurements is one example of a distributed nonlinear estimation problem where the process is locally unobservable by the individual sensors. Let $x_p$ be the position of a generic object, a robot for instance. Let $b_i$, $i = 1, \ldots, N$ be the position of the fixed range sensors. The measurement equation can be written as:

$$z_i(k) = ||x_p(k) - b_i|| + v_i(k), \qquad i = 1, \ldots, N \tag{3}$$

Since the state is locally unobservable, the sensors have to communicate with each other. In the following section we will present some possibilities for combining their estimates.

## III. Consensus Algorithms

If requirement R1) is satisfied by the choice of a suitable local nonlinear filter, requirement R2) has to be imposed using a consensus algorithm which merges the different estimates.

Consensus algorithms make use of linear maps between some local variables, for example the estimates $\hat{x}_j(k)$ with $j \in \mathcal{J}_i$, and their weighted nodal average $\bar{x}_i(k)$ as:

$$\bar{x}_i(k) = \sum_{j \in \mathcal{J}_i} W_{ij} \hat{x}_j(k)$$

or, in matrix-vector notation

$$\bar{\mathbf{x}}(k) = (W \otimes I_n)\hat{\mathbf{x}}(k) \tag{4}$$

where $\otimes$ represents the Kronecker product. Equation (4) constitutes a consensus iteration. In usual consensus algorithms this iteration is repeated $\tau$ times, from $t = 1$ to $\tau$ as:

$$\begin{cases} \bar{\mathbf{x}}(k) = (W \otimes I_n)\hat{\mathbf{x}}_{t-1}(k) \\ \hat{\mathbf{x}}_t(k) = \bar{\mathbf{x}}(k) \end{cases}$$

with $\hat{\mathbf{x}}_0(k) = \hat{\mathbf{x}}(k)$. It is straightforward to note that the $\tau$-th iteration is computed as $(W^\tau \otimes I_n)\hat{\mathbf{x}}(k)$. The matrix $W$ is required to satisfy:

$$\lim_{\tau \to \infty} W^\tau = \frac{1}{N}\mathbf{1}\mathbf{1}^T \tag{5}$$

so that the consensus iterations not only converge but they also give the mean of the initial values (which gives the name average-consensus as a particular instance of the more general $\chi$-consensus [14]). This property has been used in the last few years [15] as a means of averaging the different estimates $\hat{x}_i(k)$ without the need to know $N$, see also [3] and [12]. A typical form for $W$ is $W = I_N - \epsilon\mathcal{L}$, where $\mathcal{L}$ is the weighted Laplacian associated with the graph $\mathcal{G}$, $\epsilon$ is a positive constant which has to be less than one to ensure convergence. In its typical implementation, a consensus algorithm merges the different $\hat{x}_i(k)$ as

$$\bar{x}_i = \hat{x}_i + \epsilon/(N_i - 1) \sum_{j \in \mathcal{N}_i} (\hat{x}_j - \hat{x}_i) \tag{6}$$

delivering for $\tau \to \infty$, $\forall i$:

$$\bar{x} = \bar{x}_i = \frac{1}{N} \sum_{j \in \mathcal{V}} \hat{x}_j$$

Quite often, the number of consensus iterations is finite ($\tau \ll \infty$), or even 1, such as for instance in [3]. This reduces drastically the communication among the sensors, losing however the convergence property. Detailed analysis of such interleaved schemes is still an open problem; see [16] for a stability/convergence proof applied to a particular case.

We propose to use a weighted version of the typical algorithm (6), similar to [13]. First we cite the following lemma:

*Lemma 1* [13]: Given a set of independent and unbiased estimates, $\hat{x}_i$, with associated covariance matrices, $P_i$, where $i \in \mathcal{V}$, the following weighted averaging:

$$\begin{aligned} \hat{x} &= \left( \sum_{j \in \mathcal{V}} P_j^{-1} \right)^{-1} \sum_{j \in \mathcal{V}} P_j^{-1} \hat{x}_j \\ P^{-1} &= \sum_{j \in \mathcal{V}} P_j^{-1} \end{aligned}$$

gives the linear minimum-variance unbiased estimate of $x$.

Second, the previous weighted averaging can be seen as

$$\begin{aligned} \hat{x} &= Y^{-1} Z \\ P^{-1} &= NY \end{aligned}$$

where $Z = (1/N)\sum_{j \in \mathcal{V}} Z_j$, $Y = (1/N)\sum_{j \in \mathcal{V}} Y_j$, $Z_i = P_i^{-1}\hat{x}_i$, $Y_i = P_i^{-1}$. Therefore $Z$ and $Y$ can be calculated as standard averaging. Hence we can implement a consensus iteration in the form:

$$\begin{aligned} \bar{Z}_i &= (1/N_i) \sum_{j \in \mathcal{J}_i} Z_j \\ \bar{Y}_i &= (1/N_i) \sum_{j \in \mathcal{J}_i} Y_j \end{aligned} \tag{7}$$

which has iteration matrix $W$, where $W_{ij} = 1/N_i$ if and only if $j \in \mathcal{J}_i$, and $W_{ij} = 0$ otherwise. $W$ can be proven to satisfy (5).

Note that in general the different $\bar{x}_i(k)$'s will not be independent but they will be correlated since the sensors are observing the same model. This implies that the merging mechanism, even though it can be proven to deliver unbiased estimates, will not provide optimal solutions. However, computing the correlation among the estimates would lead to adopting a solution that is closer to the centralized one.

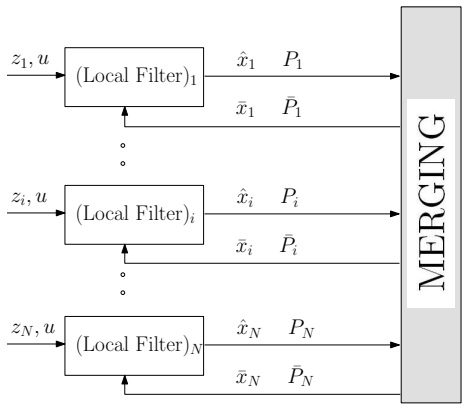


Fig. 1. A unified setting for distributed nonlinear estimation

The test cases will show how this merging is more accurate than typical consensus algorithms, even if not optimal. Furthermore, since we will adopt $\tau = 1$, note that a detailed analysis of requirement R2), i.e., convergence for $k \to \infty$, is still an open issue, as well as the consistency of the filters. However, from a practical point of view one could validate consistency using standard tests on recorded data, see [17] and [18]. Algorithm 1 shows the method for each sensor $i$.

**Algorithm 1** MERGE algorithm

1: Input: $\hat{x}_j(k)$, $P_j(k)$, $j \in \mathcal{J}_i$
2: Merge:

$$\begin{aligned} \bar{x}_i(k) &= \left(\sum_{j\in\mathcal{J}_i} P_j^{-1}(k)\right)^{-1} \sum_{j\in\mathcal{J}_i} P_j^{-1}(k)\,\hat{x}_j(k) \\ \bar{P}_i^{-1}(k) &= \sum_{j\in\mathcal{J}_i} P_j^{-1}(k) \end{aligned}$$

3: Output: $\bar{x}_i(k)$, $\bar{P}_i(k)$

## IV. DISTRIBUTED NONLINEAR ESTIMATION

Given the premises of the previous sections, our general distributed nonlinear estimation framework will consist of $N$ different local filters, connected to the merge/consensus algorithm as depicted in Figure 1. This setting unifies and simplifies the distributed implementation of three of the most used nonlinear filters, namely Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF) and Particle Filter (PF), which will be presented in the next sections. First, the distributed EKF formulation of [3] will be reviewed, pointing out the differences from the new proposed solution, then a distributed UKF algorithm will be derived and, finally, a novel distributed Particle Filter scheme will be shown and compared with [12].

### A. Distributed Extended Kalman Filters

Let $F_i$, $G_i$ and $H_i$ be respectively:

$$\begin{aligned} F_i &= \left.\frac{\partial f(x(k),u(k),w(k))}{\partial x(k)}\right|_{(\bar{x}_i(k),u(k),0)} \\ G_i &= \left.\frac{\partial f(x(k),u(k),w(k))}{\partial w(k)}\right|_{(\bar{x}_i(k),u(k),0)} \\ H_i &= \left.\frac{\partial g(x(k))}{\partial x(k)}\right|_{(\bar{x}_i(k))} \end{aligned}$$

Let $Q = \mathbb{E}[w(k)w^T(k)]$, $R_i = \mathbb{E}[v_i(k)v_i^T(k)]$. Define the weighted predicted observation vector $O_i$, the weighted true observation vector $y_i$ and their nodal aggregates:

$$\begin{aligned} O_i &\triangleq H_i^T R_i^{-1} g(F_i\bar{x}_i(k)), \quad \overline{O}_i \triangleq \sum_{j\in\mathcal{J}_i} O_j \\ y_i &\triangleq H_i^T R_i^{-1} z_i(k+1), \quad \overline{y}_i \triangleq \sum_{j\in\mathcal{J}_i} y_j \end{aligned}$$

whose differences $(O_i - y_i)$ and $(\overline{O}_i - \overline{y}_i)$ represent the mismatch between the prediction and the measurements. Define the inverse-covariance matrix $S_i$ and its nodal aggregate:

$$S_i \triangleq H_i^T R_i^{-1} H_i, \quad \overline{S}_i \triangleq \sum_{j\in\mathcal{J}_i} S_j$$

The distributed Extended Kalman Filter algorithm of [3] implements a typical consensus iteration. For each sensor it contains the following two steps:
Step 1. (Prediction):

$$\begin{aligned} \check{x}_i(k+1) &= F_i\bar{x}_i(k) \\ \check{P}_i(k+1) &= F_i\bar{P}_i(k)F_i^T + G_iQG_i^T \end{aligned} \tag{8}$$

Step 2. (Update):

$$\begin{aligned} \left(\bar{P}_i(k+1)\right)^{-1} &= \left(\check{P}_i(k+1)\right)^{-1} + \overline{S}_i \\ \bar{x}_i(k+1) &= \check{x}_i(k+1) + \\ &\quad \bar{P}_i(k+1)\left(\overline{y}_i - \overline{O}_i\right) + \epsilon\bar{P}_i(k+1) \times \\ &\quad \sum_{j\in\mathcal{N}_i}\left(\check{x}_j(k+1) - \check{x}_i(k+1)\right) \end{aligned} \tag{9}$$

The proposed version has a different update step and makes use of Algorithm 1:
Step 1. (Prediction): same as in (8)
Step 2. (Modified Update):

$$\begin{aligned} \left(P_i(k+1)\right)^{-1} &= \left(\check{P}_i(k+1)\right)^{-1} + S_i \\ \hat{x}_i(k+1) &= \check{x}_i(k+1) + \\ &\quad P_i(k+1)\left(y_i - O_i\right) \end{aligned} \tag{10}$$

Step 3. (Merge):

$$\left(\bar{x}_i(k+1), \bar{P}_i(k+1)\right) = \text{MERGE}(\hat{x}_j(k+1), P_j(k+1)),\ j \in \mathcal{J}_i$$

Note that the consensus algorithm in (9) has a similar form to (6). In addition, $y_i$, $O_i$ and $S_i$ are used in the modified update step rather than $\overline{y}_i$, $\overline{O}_i$ and $\overline{S}_i$. The reasons for this is the aim to reduce the amount of data that the sensors send to each other. Hence, the final new algorithm consists of a modified local prediction-update step, followed by a merge step.

### B. Distributed Unscented Kalman Filters

Let

$$(\hat{x}_i(k+1), P_i(k+1)) = \text{UKF}_i(\bar{x}_i(k), \bar{P}_i(k), z_i(k+1))$$

be the local Unscented Kalman Filter, whose formulation can be found for example in [19]. Then, the Distributed UKF algorithm is shown in Algorithm 2.

**Algorithm 2** DUKF algorithm

1: $\bar{x}_i(0)$, $\bar{P}_i(0)$
2: **while** new data exists **do**
3: Local UKF:

$$(\hat{x}_i(k+1), P_i(k+1)) = \text{UKF}_i(\bar{x}_i(k), \bar{P}_i(k), z_i(k+1))$$

4: Merge:

$$(\bar{x}_i(k+1), \bar{P}_i(k+1)) = \text{MERGE}(\hat{x}_j(k+1), P_j(k+1)), \quad j \in \mathcal{J}_i$$

5: **end while**

As it can be seen, the structure fits in the same framework as before, using a local UKF filter and a merge step which processes the different estimates and their covariances.

### C. Distributed Particle Filters

Distributed Particle Filters are a rather unexplored research field. The main idea behind them is that local particle filters agree upon a common proposal distribution, from which to draw the particles. A comprehensive overview of standard Particle Filters can be found in [20], while a survey of recent developments in distributed Particle Filters is given in [21], including references to applications in target tracking, [22] and [23]. For our purposes the formulation of [12] can be considered as the state-of-the-art in this field.

The key concept is to use a Gaussian proposal distribution such as:

$$q(x(k+1)|x(k), z(k+1)) = \mathcal{N}(\mu(k), \Sigma(k))$$

where $\mathcal{N}$ represents a normal distribution with mean $\mu(k)$ and covariance $\Sigma(k)$. The pair $(\mu(k), \Sigma(k))$ is calculated by propagating $(\hat{x}(k), P(k))$ via an Unscented Transformation, UT, as in [12]:

$$(\mu(k), \Sigma(k)) = \text{UT}(\hat{x}(k), P(k))$$

whereas the couple $(\hat{x}(k), P(k))$ can be approximated via consensus using the local couples $(\hat{x}_i(k), P_i(k))$. These can be computed by each local Particle Filter (at the preceding time instant) in the following way:

$$\hat{x}_i(k) = \sum_{j=1}^{m} \omega_{i,k}{}^j x_i(k)^j \tag{11}$$

$$P_i(k) = \sum_{j=1}^{m} \omega_{i,k}{}^j (x_i(k)^j - \hat{x}_i(k))(x_i(k)^j - \hat{x}_i(k))^T \tag{12}$$

where $j$ represents the particle index, $m$ the number of particles, $x_i(k)^j$ and $\omega_{i,k}{}^j$ respectively the state and the weight of the $j$-th particle for the $i$-th sensor at time $k$. In the formulation of [12], the global couple $(\hat{x}(k), P(k))$ is approximated via a consensus algorithm in the form of (4):

$$\bar{x}_i(k) = \hat{x}_i(k) + \epsilon/(N_i - 1) \sum_{j \in \mathcal{N}_i} (\hat{x}_j(k) - \hat{x}_i(k))$$

$$\bar{P}_i(k) = P_i(k) + \epsilon/(N_i - 1) \sum_{j \in \mathcal{N}_i} (P_j(k) - P_i(k))$$

Therefore, after the consensus iteration each local PF has the new proposal distribution:

$$q(x(k+1)|x(k), z(k+1)) = \mathcal{N}(\mu_i(k), \Sigma_i(k))$$

with

$$(\mu_i(k), \Sigma_i(k)) = \text{UT}(\bar{x}_i(k), \bar{P}_i(k))$$

In our formulation, we will use the MERGE algorithm instead of the standard consensus algorithm, thus:

$$(\bar{x}_i(k), \bar{P}_i(k)) = \text{MERGE}(\hat{x}_j(k), P_j(k)), \quad j \in \mathcal{J}_i$$

Algorithm 3 presents the proposed modified method. Note that PF indicates the local Particle filter.

**Algorithm 3** DPF algorithm

1: $\bar{x}_i(0)$, $\bar{P}_i(0)$
2: **while** new data exists **do**
3: LOCAL FILTER{
a. Propagation of $(\bar{x}_i(k), \bar{P}_i(k))$:

$$(\mu_i(k), \Sigma_i(k)) = \text{UT}(\bar{x}_i(k), \bar{P}_i(k))$$

b. Local PF with proposal distribution $\mathcal{N}(\mu_i(k), \Sigma_i(k))$:

$$(x_i(k+1)^j, \omega_{i,k+1}{}^j) = \text{PF}_i(\mu_i(k), \Sigma_i(k), z_i(k+1))$$

c. Compute $(\hat{x}_i(k+1), P_i(k+1))$ via (11) - (12)
d. Merge:

$$(\bar{x}_i(k+1), \bar{P}_i(k+1)) = \text{MERGE}(\hat{x}_j(k+1), P_j(k+1)), \quad j \in \mathcal{J}_i$$

}
4: **end while**

*Remark 2:* Note that in [12] the choice of $\tau$ for the consensus algorithm is left to the user as a parameter. We will assume $\tau = 1$ to compare it with our scheme.

## V. TEST CASES

In this section we present two different test cases to analyze the proposed algorithms. First, we consider the localization problem of a unicycle robot in a 2D environment. This is representative of an experimental setup currently under development by the authors. Second, we estimate the state of an autonomous underwater vehicle, which can represent a scaled model of many existing underwater robotic platforms, see for example [9]. In both cases we define the error of sensor $i$ at time $k$, $e_i(k)$, as the distance between the true position at that time and the one estimated by the sensor $i$. Let the mean error $e_m$ be defined as:

$$e_m = \frac{1}{NT} \sum_{i=1}^{N} \sum_{k=0}^{T} e_i(k)$$

where $T$ is the final time of simulation. Let the average error be the mean error averaged on a number of different simulations.

### A. Unicycle robot

The state of the unicycle robot is chosen as $x = (x_p^T, \theta)^T = (\xi, \eta, \theta)^T$, where $x_p \in \mathbb{R}^2$ is the position and $\theta$ is the orientation. The model equations are:

$$\begin{aligned}
\xi(k+1) &= \xi(k) + \frac{\hat{s}}{\hat{r}}\left(\sin(\theta(k) + \hat{r}\Delta t) - \sin\theta(k)\right) \\
\eta(k+1) &= \eta(k) - \frac{\hat{s}}{\hat{r}}\left(\cos(\theta(k) + \hat{r}\Delta t) - \cos\theta(k)\right) \\
\theta(k+1) &= \theta(k) + \hat{r}\Delta t + \gamma\Delta t \\
\hat{s} &= s + n_s \\
\hat{r} &= r + n_r
\end{aligned}$$

where $s$ and $r$ are the velocity and the angular velocity control inputs, respectively, and $n_s$, $n_r$ and $\gamma$ are noise terms. More details about the model can be found in [24]. We assume to have $N = 15$ sensors sparsely distributed in the environment. The simulation parameters are $\Delta t = 1$ s, $T = 130$ s, mean$(s) = 30$ cm/s, std$(n_s) = 5$ cm/s, std$(n_\omega) = 0.01$ rad/s and std$(\gamma) = 0.01$ rad, where mean$(\cdot)$ is the mean operator and std$(\cdot)$ is the standard deviation. We assume that the measurement error in equation (2) is std$(v_i) = 0.01$ m, for all the sensors.

We consider 500 particles for the DPF. We run 200 different simulations, varying randomly the position of the sensors. For the chosen communication range, in mean, each sensor has 4 neighbors. In Figure 2 an example of the simulation results is shown. In Table I the comparison between the proposed algorithms and the ones in the literature is reported. The average error is based on the 200 different simulations.

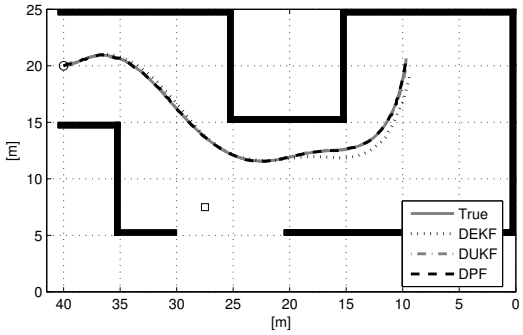


Fig. 2. Example of the simulation results for the proposed DEKF, DUKF and DPF algorithms in comparison with the true trajectory for the sensor located at the □ position. The robot starts at the circle at the top.

The results show that the proposed algorithms outperform those in the literature. Furthermore, in this situation, the proposed DUKF is more accurate than the DPF of [12]. This can be exploited in cases where we need both high sampling frequency and high accuracy.

TABLE I

COMPARISON BETWEEN THE PROPOSED ALGORITHMS AND THOSE IN THE LITERATURE, [3] AND [12]. THERE IS CURRENTLY NO PUBLISHED DUKF METHOD TO THE BEST OF OUR KNOWLEDGE.

| | Average Error [m] | |
|---|---|---|
| | Literature Results | Proposed Formulation |
| DEKF | $3.371 \pm 1.535$ | $1.109 \pm 0.916$ |
| DUKF | – | $0.104 \pm 0.035$ |
| DPF | $0.388 \pm 0.130$ | $0.071 \pm 0.040$ |

### B. Autonomous underwater vehicle

As a second test case we simulate the localization problem of an autonomous underwater vehicle (AUV). The state of the AUV is chosen as $x = (x_p^T, s_p^T)^T$, where $x_p \in \mathbb{R}^3$ is the position and $s_p \in \mathbb{R}^3$ is the velocity. The dynamical equations are:

$$\begin{aligned}
x_p(k+1) &= x_p(k) + s_p(k)\Delta t \\
s_p(k+1) &= s_p(k) + \frac{\Delta t}{m}\left(\hat{u} - \alpha \left\|s_p(k)\right\| s_p(k)\right) \\
\hat{u} &= u + n_u
\end{aligned}$$

where $m$ is the mass of the vehicle, $\alpha$ is a drag coefficient, and $n_u$ is a noise term. We assume to have $N = 25$ sensors sparsely distributed at varying heights from a plane surface. The different heights simulate an uneven seafloor. We take $\Delta t = 1$ s, $T = 130$ s, $m = 1$ kg, $\alpha = 1$ kg/s, $||\text{mean}(u)|| \sim 0.5$ N and std$(n_u) = (0.05, 0.05, 0.025)^T$ N. We assume that the measurement error in equation (2) is std$(v_i) = 0.1$ m, for all the sensors. We consider 500 particles for the DPF. We run 2500 different simulations, varying randomly the position and the communication range of the sensors. Figure 3 depicts the average error of the proposed algorithms versus the second smallest eigenvalue of the communication graph Laplacian (also called the algebraic connectivity). The average error is based on 2500 simulations. The second smallest eigenvalue, denoted as $\lambda_2$, or its normalized counterpart, $\lambda_2/\lambda_{2,\max}$, dictate the convergence rate of the consensus algorithm, [25]. Values of $\lambda_2/\lambda_{2,\max}$ close to 0 represent graphs which are not highly connected, leading to more distributed estimation problems. Values of $\lambda_2/\lambda_{2,\max}$ near 1 reveal highly connected graphs, thus estimation problems close to the centralized case. Here $\lambda_{2,\max}$ is the maximum over the graphs from the 2500 simulations. In Figure 3 a dot at the coordinate $(\phi, \psi)$, represents that the graphs with $\lambda_2/\lambda_{2,\max} \in (\phi - 0.05, \phi + 0.05)$ have an average error of $\psi$. The shaded areas show the standard deviation of these errors. Note that the DEKF estimations are not depicted here because they do not converge. The DUKF is shown without the standard deviation to make the graph more readable, its value is on the order of 0.3 m.

The results show that the proposed DPF outperforms the DPF found in the literature. The reason of this result, as well as the one of Section V.A, is the MERGE algorithm. In fact, this algorithm delivers estimates closer to the minimum-variance one than the literature, e.g. [12], where simple averaging algorithms are implemented. This also means that, since the trace of our covariance is smaller than the one of [12], then given a set of particles, they will characterize better

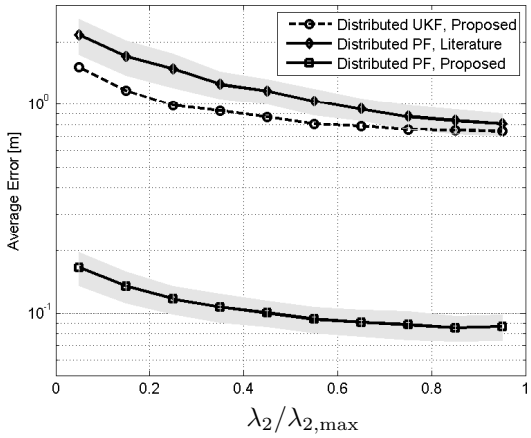


Fig. 3. Comparison between the proposed algorithms and the one in the literature with respect to the normalized $\lambda_2$ value. The Average Error is computed as the mean error of the sensor averaging 2500 different simulations. The shaded areas are the standard deviations of the bold lines. The DUKF's standard deviation is not depicted.

the a posteriori distribution. A limit of our procedure is that this 'small covariance' could cause an impoverishment of the particle diversity, which may lead to a loss in accuracy. This has not been detected in our simulation but it is a topic of further investigations.

Our results show also that in this simulation study, the DUKF has similar average error as the DPF reported in the literature. This is important because DUKF is less computationally expensive than DPF which is crucial in the context of designing fast yet accurate algorithms.

## VI. CONCLUSIONS

We proposed an effective scheme to distribute the nonlinear estimation problem among different sensing units. We applied the method to localization with range-only measurements, designing distributed Extended Kalman Filters, distributed Unscented Kalman Filters and distributed Particle Filters. The proposed algorithms outperform the ones found in the literature using a simulated benchmark.

As future work we plan to implement the scheme in a real robotic testbed which is currently under development. Moreover, we will extend the formulation to multi-robot settings, in which some of the sensors are moving with the robots themselves. Another promising extension that can be investigated is the case in which the local filters are different: some of them can be UKFs and others PFs. This is especially practical when the sensors have different computational capabilities.

## REFERENCES


[1] P. Alriksson and A. Rantzer, "Model Based Information Fusion in Sensor Networks," in *Proceedings of the 17th IFAC World Congress, Seoul, Korea*, July 2008.

[2] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri, "Distributed Kalman Filtering Using Consensus Strategies," in *Proceedings of the 46th Conference on Decision and Control, New Orleans, LA, USA, December*, 2007.

[3] R. Olfati-Saber, "Distributed Kalman Filtering for Sensor Networks," in *Proceedings of the 46th IEEE Conference on Decision and Control 2007, New Orleans, LA, USA, December*, December 2007.

[4] A. Speranzon, C. Fischione, and K. H. Johansson, "Distributed and Collaborative Estimation over Wireless Sensor Networks," in *Proceedings of the 45th IEEE Conference on Decision and Control, Manchester Grand Hyatt Hotel, San Diego, CA, USA, December 13-15*, 2006.

[5] A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha, "Tracking Moving Devices with the Cricket Location System," in *In Proceedings of MobiSYS'04, Boston, USA*, June 2004.

[6] A. Hossain, H. N. Van, Y. Jin, and W. Soh, "Indoor Localization Using Multiple Wireless Technologies," in *IEEE International Conference on Mobile Adhoc and Sensor Systems*, Los Alamitos, CA, USA, 2007.

[7] B. K. Kim, H. M. Jung, J.-B. Yoo, W. Y. Lee, C. Y. Park, and Y. W. Ko, "Design and Implementation of Cricket-based Location Tracking System," *Proceedings of World Academy of Science, Engineering and Technology*, vol. 28, pp. 96 – 100, April 2008.

[8] V. Chandrasekhar, W. K. Seah, Y. S. Choo, and H. V. Ee, "Localization in Underwater Sensor Networks – Survey and Challenges," in *Proceedings of the 1st ACM International Workshop on Underwater Networks, September 25, Los Angeles, CA, USA*, 2006.

[9] P. Corke, C. Detweiler, M. Dunbabin, M. Hamilton, D. Rus, and I. Vasilescu, "Experiments with Underwater Robot Localization and Tracking," in *Proceeding of the IEEE International Conference on Robotics and Automation, Roma, Italy, April 10-14*, 2007.

[10] Y. Huang, W. Liang, H. Yu, and Y. Xiao, "Target tracking based on a distributed particle filter in underwater sensor networks," *Wireless Communication and Mobile Computing*, vol. 8, pp. 1023 – 1033, 2008.

[11] I. Rekleitis, J. Bedwani, and E. Dupuis, "Autonomous Planetary Exploration using LIDAR data," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, May 12-17*, 2009.

[12] D. Gu, J. Sun, Z. Hu, and H. Li, "Consensus Based Distributed Particle Filter in Sensor Network," in *Proceedings of the 2008 IEEE International Conference on Information and Automation, June 20-23, Zhangjiajie, China*, 2008.

[13] L. Xiao, S. Boyd, and S. Lall, "A Scheme for Robust Distributed Sensor Fusion Based on Average Consensus," in *Proceedings of the International Conference on Information Processing in Sensor Networks, Los Angeles*, April 2005.

[14] J. Cortés, "Distributed algorithms for reaching consensus on general functions," *Automatica*, vol. 44, pp. 726 – 737, 2008.

[15] W. Ren and R. Beard, *Distributed consensus in multi-vehicle cooperative control: theory and applications*, ser. Communications and Control Engineering Series. Springer-Verlag London, 2008.

[16] M. Kamgarpour and C. Tomlin, "Convergence Properties of a Decentralized Kalman Filter," in *Proceedings of the 47th IEEE Conference on Decision and Control, Cancun, Mexico, December 9-11*, 2008.

[17] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. Wiley Inter-Science, 2001.

[18] F. van der Heijden, "Consistency Checks for Particle Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, pp. 140 – 145, 2006.

[19] S. J. Julier and J. K. Uhlmann, "Unscented Filtering and Nonlinear Estimation," *Proceedings of IEEE*, vol. 93, no. 3, pp. 401 – 422, 2004.

[20] M. 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, no. 2, pp. 174 – 188, February 2002.

[21] A. Simonetto and T. Keviczky, "Recent Developments in Distributed Particle Filters: Towards Fast and Accurate Algorithms," in *Proceedings of the 1st IFAC Workshop on Estimation and Control of Networked Systems, September 24-26, Venice, Italy*, 2009.

[22] M. Coates, "Distributed Particle Filters for Sensor Networks," in *Proceedings of the Third International Symposium on Information Processing in Sensor Networks*, 2004, pp. 99 – 107.

[23] X. Sheng, Y. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network," in *Fourth International Symposium on Information Processing in Sensor Networks*, 2005.

[24] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[25] R. Olfati-Saber, , and R. Murray, "Consensus Problems in Networks of Agents with Switching Topology and Time-Delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520 – 1533, September 2004.