Cooperative Localization Using Posterior Linearization Belief Propagation

Ángel F. García-Fernández, Lennart Svensson, and Simo Särkkä

***Abstract*—This paper presents the posterior linearization belief propagation (PLBP) algorithm for cooperative localization in wireless sensor networks with nonlinear measurements. PLBP performs two steps iteratively: linearization and belief propagation. At the linearization step, the nonlinear functions are linearized using statistical linear regression with respect to the current beliefs. This SLR is performed in practice by using sigma-points drawn from the beliefs. In the second step, belief propagation is run on the linearized model. We show by numerical simulations how PLBP can outperform other algorithms in the literature.**

***Index Terms*—Belief propagation, cooperative localization, Gaussian message passing, posterior linearization, sigma points.**

I. Introduction

Cooperative localisation is an important problem with applications in many different fields, for example, wireless communications, robotics and vehicular networks [1]–[3]. In cooperative localisation, there is a small number of anchor nodes, whose positions are known with high accuracy, and the rest of the nodes aim to infer their own location based on inter-sensor measurements and wireless messages exchanged with other nodes. This inference problem can be posed in the Bayesian framework and represented using a probabilistic graphical model. Then, the nodes can infer their positions cooperatively using message passing algorithms [1]. In this paper, we focus on the message passing algorithm called belief propagation (BP) [4]. BP calculates approximate marginal distributions, which are called beliefs, of the nodes in an efficient way suitable for cooperative networks [1].

In cooperative localisation, the measurement model is usually nonlinear so BP implementations require certain approximations. A commonly used family of algorithms is non-parametric belief propagation (NBP), which approximates the BP messages using particles [5], [6]. For a limited number of particles, satisfactory performance is not guaranteed [5]. Nevertheless, by increasing the number of particles, messages are approximated more accurately so NBP works better at the expense of a rise in the computational burden. Therefore, it is also of interest to design computationally efficient alternatives based on parametric BP, which characterises the messages by some parameters [7].

Manuscript received January 30, 2017; revised April 11, 2017 and June 15, 2017; accepted July 24, 2017. Date of publication August 1, 2017; date of current version January 15, 2018. The review of this paper was coordinated by Dr. Jie Wang. *(Corresponding author: Ángel F. García-Fernández.)*

Á. F. García-Fernández and S. Särkkä are with the Department of Electrical Engineering and Automation, Aalto University, 02150 Espoo, Finland (e-mail: angel.garciafernandez@aalto.fi; simo.sarkka@aalto.fi).

L. Svensson is with the Department of Electrical Engineering, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden (e-mail: lennart.svensson@chalmers.se).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2017.2734683

In cooperative localisation, beliefs are unimodal when there is a sufficient number of anchor nodes and/or sufficiently accurate prior information about node locations. In this case, we can satisfactorily use Gaussian parametric BP, a common form of parametric BP which considers Gaussian distributions [8], [9]. Now, the main difficulty is to deal with nonlinear measurement models. One approach to do this, is to use Monte Carlo sampling to calculate the beliefs [9], though this implies a high computational burden. In order to lower the computational complexity, one possibility is to linearise the nonlinear functions before applying BP using analytical linearisation at the prior mean [10]. Nevertheless, sigma-point methods can deal with nonlinearities better than analytical linearisation [11] so it is convenient to use sigma-points, as in sigma-point BP (SPBP) [12] and in [13]. Sigma-point methods, such as the unscented transform, are efficient and widely used techniques to approximate the moments of a Gaussian distribution that undergoes a nonlinear transformation [11]. This procedure implicitly linearises the nonlinear functions by an approximate statistical linear regression (SLR) [14]. Consequently, the algorithms in [12], [13] perform SLR of the nonlinear functions for each message. Another relevant characteristic of SPBP is that it performs integration in possibly high-dimensional spaces that define a node and all its neighbours simultaneously.

In this paper, we propose a different use of sigma-point methods in BP inspired by the posterior linearisation filter (PLF) [15] and smoother [16]. In these papers, it is pointed out that the most commonly used Gaussian filters and smoothers approximate nonlinear functions as affine functions with additive Gaussian noise. The best approximation of the nonlinear functions, in the mean square error sense, is then obtained by SLR with respect to the posterior probability density function (PDF), which is the PDF of the states given the measurements. In practice, this idea can be implemented via an iterated procedure: linearise the model with respect to the current posterior approximation, obtain a new posterior approximation based on the linearised model and repeat until convergence. We extend this framework to BP and refer to it as posterior linearisation BP (PLBP). Therefore, in PLBP, we linearise the models with respect to the current beliefs, we run BP on the linearised model and repeat the procedure. The most important difference with SPBP is that, PLBP includes a double loop in which, after each linearisation, we obtain the new beliefs running BP on the linearised model. Another important characteristic of PLBP is that sigma-points are always used in low dimensional spaces that only consider two nodes simultaneously. The proposed double loop scheme, with low dimensional integrations, can significantly reduce the localisation error, as will be demonstrated by simulations.

II. Problem Formulation

In cooperative localisation, we have some anchor nodes, whose positions are known accurately, a number of nodes with some prior knowledge about their locations, and measurements that depend on the states of pairs of nodes. The objective is then to infer the positions of the nodes in a cooperative fashion. We proceed to pose this problem in the Bayesian framework.

A graph $G = (V, E)$ is formed by a collection of vertices/nodes $V = \{1, ..., m\}$, where m is the number of nodes, and a collection of edges $E \subset V \times V$. Each edge consists of a pair of nodes $(i, j) \in E$. The state of node i is represented by $x_i \in \mathbb{R}^{n_x}$ and we assume it has a Gaussian prior PDF

$$p_i\left(x_i\right) = \mathcal{N}\left(x_i; \overline{x}_i, P_i\right),$$

where $\mathcal{N}\left(x_i; \overline{x}_i, P_i\right)$ denotes a Gaussian PDF with mean $\overline{x}_i$ and covariance matrix P_i evaluated at x_i. Vector $z_{i,j} \in \mathbb{R}^{n_z}$, $(i, j) \in E$, represents the measurement that depends on nodes i and j. We assume that $z_{i,j}$ is a function of x_i, x_j with additive Gaussian noise

$$z_{i,j} = h_{i,j}\left(x_i, x_j\right) + \eta_{i,j} \tag{1}$$

where $\eta_{i,j}$ is zero-mean Gaussian noise with covariance matrix $R_{i,j}$. Thus, the likelihood for measurement $z_{i,j}$ is $l_{i,j}\left(z_{i,j}|x_i, x_j\right) = \mathcal{N}\left(z_{i,j}; h_{i,j}\left(x_i, x_j\right), R_{i,j}\right)$. This model can accommodate line-of-sight and non-light-of-sight scenarios [17]. Without loss of generality, we assume that if $(i, j) \in E$ then $(j, i) \notin E$, as (1) is already general enough to model all measurements between nodes i and j. For describing BP in Section III-A, we also introduce the set

$$\overline{E} = \{(i, j) : (i, j) \in E \text{ or } (j, i) \in E\}.$$

We denote $x = \left[x_1^T, ..., x_m^T\right]^T$, where superscript T denotes transpose, and z is the vector that contains all $z_{i,j}$, $(i, j) \in E$ with (i, j) arranged in any established order. In the Bayesian framework, all information of interest about x after observing z is given by the posterior PDF

$$p\left(x|z\right) \propto \prod_{i \in V} p_i\left(x_i\right) \prod_{(i,j) \in E} l_{i,j}\left(z_{i,j}|x_i, x_j\right) \tag{2}$$

where $\propto$ stands for proportionality. Even though only the edges $(i, j) \in E$ appear in the above likelihood function, we assume that all functions and variables in this paper take the same value if we interchange indices i and j, for instance, $z_{i,j} = z_{j,i}$, $h_{i,j}\left(x_i, x_j\right) = h_{j,i}\left(x_j, x_i\right)$.

Our objective is to compute/approximate the marginal PDFs of (2) for all x_i, $i \in V$, in a cooperative way. That is, each node makes part of the processing, transmits and receives messages from neighbouring nodes. With the marginal PDFs, we can estimate the node localisations and their estimation errors. However, there are two difficulties:

- D1: The nonlinear measurement models.
- D2: A cooperative calculation of the beliefs.

We proceed to describe how we tackle both difficulties.

Due to the nonlinear measurements, the beliefs cannot generally be computed in closed-form, so we need approximations. As in popular nonlinear Gaussian filters and smoothers [15], [16], nonlinear functions, see (1), are dealt with by performing an enabling approximation in which they are approximated as affine functions with additive Gaussian noise

$$h_{i,j}\left(x_i, x_j\right) \approx A_{i,j}^1 x_i + A_{i,j}^2 x_j + b_{i,j} + e_{i,j} \tag{3}$$

where $A_{i,j}^1, A_{i,j}^2, \in \mathbb{R}^{n_z \times n_x}$, $b_{i,j} \in \mathbb{R}^{n_z}$ and $e_{i,j} \in \mathbb{R}^{n_z}$ is a zero-mean Gaussian distributed random variable with covariance matrix $\Omega_{i,j}$. Under approximation (3), $p\left(x|z\right)$ is Gaussian, which implies that the marginal PDFs are Gaussian. As in many Gaussian filters/smoothers, the accuracy of the marginal PDF approximations only depends on the choice of (3) so it is of utmost importance to select it properly [15], [16]. Under approximation (3), calculating the marginal PDFs directly from $p\left(x|z\right)$ is theoretically simple as we can just integrate out the other states. However, this procedure is intractable for large networks as the complexity to compute $p\left(x|z\right)$ grows exponentially with the number of variables.

BP is an efficient algorithm suitable for cooperation in wireless networks to calculate/approximate these marginals. If the graph has no loops (it is a tree), under approximation (3), BP calculates the marginal PDFs of $p\left(x|z\right)$ in closed-form in a cooperative way [1]. Usually, the graph has loops but we can still apply BP to get approximations of the marginal PDFs [4]. The aim of this paper is to solve the joint problem of computing the marginal PDFs using BP and selecting the best possible approximation (3), to address D1 and D2.

III. Posterior Linearisation Belief Propagation

In this section, we propose the PLBP algorithm. The PLBP algorithm is iterative and has two phases: selection of the approximation (3), explained in Section III-B, and use of BP on an affine model, explained in Section III-A. We also indicate some properties on its convergence in Section III-C.

A. BP With Affine Measurement Functions

We apply BP to approximate the marginal PDFs under approximation (3), which implies that we are using BP on a Gaussian graphical model [18]. In BP, messages are transmitted between neighbouring nodes in the graph. The message $\mu_{i \to j}$ from node i to j, with $(i, j) \in \overline{E}$, is [19]

$$\mu_{i \to j}\left(x_j\right) \propto \int l_{i,j}\left(z_{i,j}|x_i, x_j\right) \mathcal{N}\left(x_i; \overline{x}_i, P_i\right) \times \prod_{p \in n(i) \setminus \{j\}} \mu_{p \to i}\left(x_i\right) dx_i \tag{4}$$

where $n\left(i\right) \subseteq V$ denotes the set of neighbouring nodes of node i according to E.

Proposition 1: Under approximation (3), the message $\mu_{i \to j}$ from node i to j is

$$\mu_{i \to j}\left(x_j\right) \propto \mathcal{N}\left(\alpha_{i \to j}; H_{i \to j} x_j, \Gamma_{i \to j}\right) \tag{5}$$

$$\alpha_{i \to j} = z_{i,j} - A_{i,j}^1 \overline{x}_{i \to j} - b_{i,j} \tag{6}$$

$$H_{i \to j} = A_{i,j}^2 \tag{7}$$

$$\Gamma_{i \to j} = R_{i,j} + \Omega_{i,j} + A_{i,j}^1 P_{i \to j} \left(A_{i,j}^1\right)^T \tag{8}$$

where $\overline{x}_{i \to j}$ and $P_{i \to j}$ are given by

$$\mathcal{N}\left(x_i; \overline{x}_{i \to j}, P_{i \to j}\right) \propto \prod_{p \in n(i) \setminus \{j\}} \mathcal{N}\left(\alpha_{p \to i}; H_{p \to i} x_i, \Gamma_{p \to i}\right) \times \mathcal{N}\left(x_i; \overline{x}_i, P_i\right). \tag{9}$$

Messages are characterised by $\alpha_{i \to j} \in \mathbb{R}^{n_z}$, $H_{i \to j} \in \mathbb{R}^{n_z \times n_x}$ and $\Gamma_{i \to j} \in \mathbb{R}^{n_z \times n_z}$ and represent likelihoods resulting from linear observations with Gaussian noise. We can calculate $\overline{x}_{i \to j}$ and $P_{i \to j}$ in (9) by performing $|n\left(i\right) \setminus \{j\}|$ Kalman filter updates [20] on $\mathcal{N}\left(\cdot; \overline{x}_i, P_i\right)$. The pseudocode for calculating a message is given in Algorithm 1.

Algorithm 1: Calculation of BP Message from node i to j.

```
Input: Incoming messages $\alpha_{p\to i}, H_{p\to i}, \Gamma_{p\to i}$ for
$p \in n(i) \setminus \{j\}$, prior moments $\overline{x}_i, P_i$, measurement $z_{i,j}$ and
linearisation $A^1_{i,j}, A^2_{i,j}, b_{i,j}, \Omega_{i,j}$.
Output: Output message $\alpha_{i\to j}, H_{i\to j}, \Gamma_{i\to j}$.
  - Set $\overline{x}_{i\to j} = \overline{x}_i$ and $P_{i\to j} = P_i$.   ▷ Kalman filter updates
  for $p \in n(i) \setminus \{j\}$ do
      - $\overline{z} = H_{p\to i}\overline{x}_{i\to j}$, $S = H_{p\to i}P_{i\to j}H^T_{p\to i} + \Gamma_{p\to i}$.
      - $\Psi = P_{i\to j}H^T_{p\to i}$, $\overline{a} = \overline{x}_{i\to j} + \Psi S^{-1}(\alpha_{p\to i} - \overline{z})$.
      - $A = P_{i\to j} - \Psi S^{-1}\Psi^T$.
      - Set $\overline{x}_{i\to j} = \overline{a}$, $P_{i\to j} = A$.
  end for
  - Compute $\alpha_{i\to j}, H_{i\to j}, \Gamma_{i\to j}$ using (6)-(8).
```

We consider a BP implementation in which messages are initiated to 1 (uniform function) and all nodes transmit all their messages at each iteration. When this iteration converges, the marginal PDF of node i calculated by BP is

$$\begin{aligned} q(x_i|z) &= \mathcal{N}(x_i; \overline{u}_i, W_i) \\ &\propto \mathcal{N}(x_i; \overline{x}_i, P_i) \prod_{p\in n(i)} \mu_{p\to i}(x_i) \end{aligned} \tag{10}$$

where $\overline{u}_i$ and W_i are the mean and covariance matrix, which can be calculated by performing $|n(i)|$ Kalman filter updates on $\mathcal{N}(x_i; \overline{x}_i, P_i)$. A similar procedure is detailed in Algorithm 1 to compute (9).

Even though our objective is to compute the marginal PDFs, due to the dependence of the measurement function on two nodes, see (1), it will be useful to calculate the joint posterior PDF over two neighbouring nodes. The BP messages can also be used to calculate the joint posterior PDF of nodes $(i,j) \in E$ as [19, Eq. (4)]

$$\begin{aligned} q(x_i, x_j|z) &= \mathcal{N}\left(\left[x_i^T, x_j^T\right]^T; \overline{u}_{i,j}, W_{i,j}\right) \\ &\propto \mathcal{N}(x_i; \overline{x}_i, P_i)\mathcal{N}(x_j; \overline{x}_j, P_j)\, l_{i,j}(z_{i,j}|x_i, x_j) \\ &\quad \times \left[\prod_{p\in n(i)\setminus\{j\}} \mu_{p\to i}(x_i)\right]\left[\prod_{p\in n(j)\setminus\{i\}} \mu_{p\to j}(x_j)\right] \end{aligned} \tag{11}$$

where $\overline{u}_{i,j}$ and $W_{i,j}$ can be computed using Kalman filter updates as in (9). If the graph is a tree, the BP iteration is ensured to converge and the BP outputs (10) and (11) are exact marginals of (2). If the graph is not a tree, the iteration is not ensured to converge but if it does, the BP outputs (10) and (11) can be quite accurate approximations.

B. Selection of the Linearisation

This section describes the selection of the approximation (3) using posterior linearisation [15], [16]. The basic idea behind posterior linearisation is that we want the approximation in (3) to be accurate in the area of interest, namely, where the posterior distribution has its mass. More specifically, given the measurements, the selected $A^1_{i,j}$, $A^2_{i,j}$ and $b_{i,j}$ are chosen to minimise the mean square error between the measurement function $h_{i,j}(\cdot)$ and its approximation, and $\Omega_{i,j}$ represents the mean square error matrix of the approximation [15]:

$$\left(A^+_{i,j}, b^+_{i,j}\right) = \underset{(A_{i,j}, b_{i,j})}{\arg\min}\ \mathrm{E}\left[(\cdot)^T\left(h(x) - A_{i,j}x - b_{i,j}\right)|z\right]$$

$$\Omega^+_{i,j} = \mathrm{E}\left[\left(h(x) - A^+_{i,j}x - b^+_{i,j}\right)(\cdot)^T|z\right]$$

where $x = \left[x_i^T, x_j^T\right]^T$, $A_{i,j} = \left[A^1_{i,j}, A^2_{i,j}\right]$, the expected value is taken with respect to $p(x_i, x_j|z)$ and $(\cdot)^T a$ and $a(\cdot)^T$ represent $a^T a$ and aa^T, respectively, The solution to this problem is given by selecting $A^1_{i,j}$, $A^2_{i,j}$, $b_{i,j}$ and $\Omega_{i,j}$ using statistical linear regression (SLR) with respect to the joint posterior over the two nodes [15, Sec. II]. In practice, the SLR parameters can be efficiently approximated using sigma-point methods [11], [20], see Algorithm 2.

A challenge with posterior linearisation is that we need the joint posterior to perform posterior linearisation and vice versa. The solution proposed in posterior linearisation algorithms is to solve the problem in an iterated fashion. That is, we perform SLR of the nonlinear functions with respect to the best available approximation of the posterior. After applying BP on the linearised model, we expect to obtain improved approximations of the joint posterior PDFs, which are used to obtain an even better SLR that we compute at the next iteration. The steps of the iteration, which is performed J times, are provided in Algorithm 3. It should be noted that posterior linearisation iterations only change the linearisations, not the prior. If the measurement function is already linear, PLBP corresponds to Gaussian BP [18], [21].

Algorithm 2: SLR of $h(\cdot)$ w.r.t. $p(\cdot)$ using sigma-points

```
Input: Function $h(\cdot)$ and first two moments $\overline{x}$, $P$ of $p(\cdot)$.
Output: SLR parameters $(A^+, b^+, \Omega^+)$.
    - Select $m$ sigma-points $\mathcal{X}_1, \ldots, \mathcal{X}_m$ and weights $\omega_1, \ldots, \omega_m$
      according to $\overline{x}$ and $P$ [20].
    - Transform the sigma-points $\mathcal{Z}_j = h(\mathcal{X}_j)$   $j = 1, \ldots, m$.
    - Compute
      $\overline{z} = \sum_{j=1}^m \omega_j \mathcal{Z}_j$, $\Psi = \sum_{j=1}^m \omega_j (\mathcal{X}_j - \overline{x})(\mathcal{Z}_j - \overline{z})^T$
      $\Phi = \sum_{j=1}^m \omega_j (\mathcal{Z}_j - \overline{z})(\mathcal{Z}_j - \overline{z})^T$
    - $A^+ = \Psi^T P^{-1}$, $b^+ = \overline{z} - A^+\overline{x}$,
      $\Omega^+ = \Phi - A^+ P (A^+)^T$.
```

Algorithm 3: PLBP in cooperative localisation.

```
- Set $\overline{u}^0_{i,j} = \left[\overline{x}_i^T, \overline{x}_j^T\right]^T$, $W^0_{i,j} = \mathrm{diag}(P_i, P_j)$ for $(i,j) \in E$.
for $k = 1$ to $J$ do
    - Obtain $A^1_{i,j}, A^2_{i,j}, b_{i,j}, \Omega_{i,j}$ for all $(i,j) \in E$:
      ○ Run Algorithm 2 with $h_{i,j}(\cdot)$ and $\overline{u}^{k-1}_{i,j}, W^{k-1}_{i,j}$.
    - Run BP with linearised model, see Sec. III-A.
    - Calculate $\overline{u}^k_{i,j}, W^k_{i,j}$ using (11).
end for
- Get $\overline{u}^J_i, W^J_i$ for $i \in V$ from $\overline{u}^J_{i,j}, W^J_{i,j}$ by marginalisation.
```

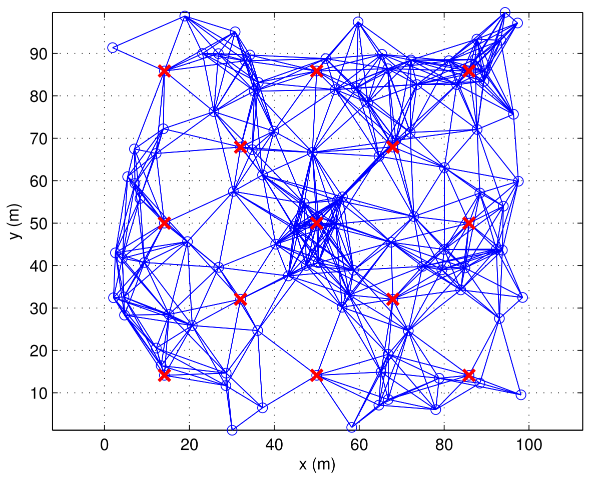

Fig. 1. Scenario of the simulations. Red crosses indicate the positions of 13 anchor nodes, blue circles the positions of the other 100 nodes and blue lines the edges of the graph. Communication radius is 20 m.

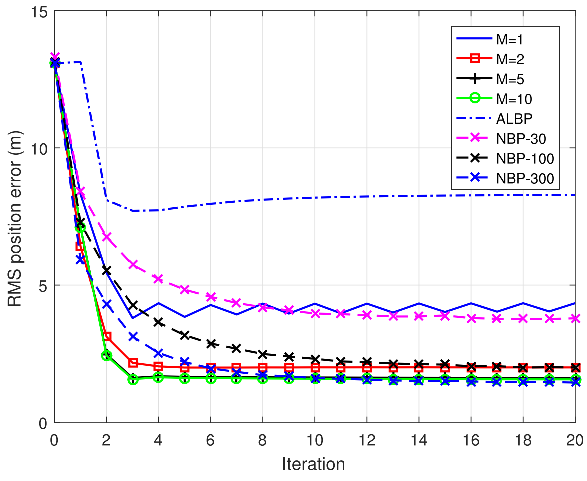

Fig. 2. RMS error against number of iterations. Performance improves with M, number of BP iterations per linearisation.

C. Convergence

If the graph is a tree, BP provides the exact joint marginals under approximation (3) so PLBP can be seen as an efficient application of the iterated PLF [15] to obtain (2). As a result, the local convergence theorem of the iterated PLF [15, Sec. IV.E and App. B], which resembles the Gauss-Newton algorithm convergence theorem, is also valid for the PLBP algorithm for trees, but not for general graphs. On the contrary, for SPBP, there is no available convergence theorem even for trees in the literature.

For graphs with loops, BP is not ensured to converge though it does in many practical cases. If it does not, damping helps improve the convergence of BP [22], and therefore of PLBP.

IV. Simulation Results

We compare PLBP (M BP iterations per linearisation) with the analytical linearisation BP (ALBP) [10], in which the nonlinear functions are linearised once using analytical linearisation at the prior mean and SPBP [12], implemented using BP, so that simulation results reflect the differences on nonlinear Gaussian BP algorithms. We have also implemented NBP, according to Algorithm 1 and 2 with $k = 1$ in [5] (k as defined in [5]), using 30, 100 and 300 particles. Besides, we have computed the posterior Crámer-Rao lower bound (PCRLB) [23], which is a bound on performance on a centralised solution to the problem, using 40000 samples from the prior.

We evaluate the scenario in Fig. 1, which was used in [1]. The state $x_i = (p_{x,i}, p_{y,i}) \in \mathbb{R}^2$ of node i consists of its xy-position. For normal nodes, we use $P_i = \text{diag}\left(\sigma^2, \sigma^2\right)$ with $\sigma = 10$ m, and $\overline{x}_i$ is drawn from a Gaussian PDF whose mean is the true position of node i and covariance P_i. For anchor nodes, we use the same prior but with $P_i = \text{diag}\left(\sigma_a^2, \sigma_a^2\right), \sigma_a = 0.1$ m. Note that we assume some prior knowledge on the locations of the nodes, represented by P_i, so that beliefs are mostly unimodal and Gaussian BP can work well. If beliefs are not unimodal, we should use NBP rather than Gaussian BP, at the expense of a higher computational burden. We use range measurements with $R = 1$ m^2 and

$$h_{i,j}\left(x_i, x_j\right) = \sqrt{\left(p_{x,i} - p_{x,j}\right)^2 + \left(p_{y,i} - p_{y,j}\right)^2}. \qquad (12)$$

SPBP and the SLRs in Algorithm 2 have been implemented using the unscented transform [11] with a weight 1/3 for the sigma-point at the mean. As (x_i, x_j) has dimension 4, 9 sigma-points are used in PLBP. We evaluate the algorithms using Monte Carlo simulation with 200 runs of random measurements.

The SPBP algorithm does not work properly in this scenario and the error increases with the BP iterations. The reason why this happens is mainly due to how SPBP makes use of sigma-points, which belong to a high-dimensional space that considers a node and its neighbours simultaneously. In this example, the maximum number of neighbours of a node is 20, which yields a space of $(20 + 1) \times 2 = 42$ dimensions. An important consequence of this, is that the sigma-points are placed outside the region of interest and the resulting performance is low.

The RMS error against the linearisation iteration, k in Algorithm 3 is shown in Fig. 2. It is important to notice the improvement of PLBP as M increases, up to a certain point. This implies that after a linearisation, we obtain a better performance if we let BP converge so that we obtain the most accurate posterior approximation that is possible with the current linearisation. For $M > 1$, all PLBP algorithms converge after a few iterations. Therefore, the improvement of PLBP with respect to SPBP is due to the use of sigma-point integration in a low dimensional state space and the use of a double loop. NBP with 300 particles (NBP-300) works very well and achieves lowest errors. NBP-100 works well but it does not attain the performance of PLBP with $M = 5$ or $M = 10$. However, NBP has a much higher computational burden than PLBP due to the need of a much higher number of points for good performance. ALBP has worse performance than PLBP and NBP. The running times in seconds of our Matlab implementations with 20 iterations are: 5 ($M = 1$), 8 ($M = 2$), 16 ($M = 5$), 31 ($M = 10$), 56 (NBP-30), 155 (NBP-100), 513 (NBP-300), 3 (ALBP) and 10 (SPBP).

We also show the RMS error after 20 iterations for different values of σ in Table I, where we additionally show the results for a PLBP version that approximates the SLRs using Taylor linearisation at the mean, which implies $\Omega_{i,j} = 0$, rather than sigma-points. Note that PLPB Taylor (M=1) corresponds to the BP algorithm in [8] in this scenario. In this table, the entry NW means that the method did not converge to a value that is lower than the initial error so it is not working properly. On the whole, PLBP performs better than (Taylor) PLBP, as

TABLE I
FINAL RMS ERROR (M) FOR DIFFERENT VALUES OF σ (UNDERLINED THE BEST GAUSSIAN BP ALGORITHM)

	M PLBP (sigma-points)				M PLBP (Taylor)						NBP (particles)			
σ	1	2	5	10	1	2	5	10	SPBP	ALBP	30	100	300	PCRLB
15	7.53	3.61	2.74	<u>2.57</u>	9.71	6.52	4.60	4.32	NW	15.34	NW	2.47	1.85	0.70
10	4.34	2.00	1.61	<u>1.56</u>	8.03	4.48	2.57	2.77	NW	8.29	3.78	2.00	1.44	0.70
5	2.06	1.17	0.90	<u>0.84</u>	2.22	1.18	0.90	<u>0.84</u>	NW	3.38	3.22	1.61	1.13	0.70
1	0.70	0.64	0.62	<u>0.61</u>	0.70	0.64	0.61	<u>0.61</u>	0.62	0.62	1.57	0.92	0.73	0.61

the SLR approximation is more accurate and $\Omega_{i,j}$ is not set to zero. SPBP does not work well except for $\sigma = 1$. In this case, SPBP is slightly outperformed by PLBP ($M = 10$). PLBP methods perform best if M is high enough so that each BP iteration per linearisation converges. NBP-30 and NBP-100 always provide a higher error than PLBP ($M = 10$), except for $\sigma = 15$. In this case, NBP-30 simulation does not work well due to the low number of particles and high uncertainty and NBP-100 outperforms PLBP. NBP-300 provides a higher error than PLBP ($M = 10$) except for $\sigma \in \{10, 15\}$. One reason why this happens is that some multimodality starts to appear for $\sigma \geq 5$, but even in this case, PLBP manages to significantly lower the error. All distributed solutions are far from the PCRLB, which is a bound on the performance of a centralised algorithm, except PLBP, ALBP and SPBP for $\sigma = 1$. In short, PLBP clearly outperforms the other sigma point implementation of BP in the literature, the SPBP. It also performs very well compared to NBP for low prior uncertainty, with a much lower computational burden, and attains the performance of an optimal centralised solution for $\sigma = 1$.

We would finally like to clarify that PLBP can be used for mobile networks, which contain nodes that can move, using Gaussian filtering [20]: each node predicts its belief at the next time step using the dynamic model and PLBP is used on the predicted beliefs to update them.

V. CONCLUSIONS

We have proposed the PLBP algorithm to address the cooperative localisation problem. PLBP carries out an iterated procedure in which the nonlinear functions are linearised using SLR with respect to the current posterior approximation. Then, BP is used on the linearised model to obtain a new, more accurate posterior approximation, which will be used again to linearise the nonlinear functions. The high performance of PLBP in comparison to other Gaussian BP algorithms has been demonstrated via numerical simulations.

REFERENCES

[1] H. Wymeersch, J. Lien, and M. Win, "Cooperative localization in wireless networks," *Proc. IEEE*, vol. 97, no. 2, pp. 427–450, Feb. 2009.

[2] S. Kianoush, A. Vizziello, and P. Gamba, "Energy-efficient and mobile-aided cooperative localization in cognitive radio networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 5, pp. 3450–3461, May 2016.

[3] T. V. Nguyen, Y. Jeong, H. Shin, and M. Z. Win, "Least square cooperative localization," *IEEE Trans. Veh. Technol.*, vol. 64, no. 4, pp. 1318–1330, Apr. 2015.

[4] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA, USA: The MIT Press, 2009.

[5] A. Ihler, J. Fisher, R. Moses, and A. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 809–819, Apr. 2005.

[6] V. Savic and S. Zazo, "Cooperative localization in mobile networks using nonparametric variants of belief propagation," *Ad Hoc Netw.*, vol. 11, no. 1, pp. 138–150, 2013.

[7] J. Lien, U. J. Ferner, W. Srichavengsup, H. Wymeersch, and M. Z. Win, "Comparison of parametric and sample-based message representation in cooperative localization," *Int. J. Navigat. Observ.*, vol. 2012, pp. 1–10, 2012.

[8] W. Yuan, N. Wu, B. Etzlinger, H. Wang, and J. Kuang, "Cooperative joint localization and clock synchronization based on Gaussian message passing in asynchronous wireless networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 9, pp. 7258–7273, Sep. 2016.

[9] S. Li, M. Hedley, and I. B. Collings, "New efficient indoor cooperative localization algorithm with empirical ranging error model," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 7, pp. 1407–1417, Jul. 2015.

[10] B. Li, N. Wu, H. Wang, P.-H. Tseng, and J. Kuang, "Gaussian message passing-based cooperative localization on factor graph in wireless networks," *Signal Process.*, vol. 111, pp. 1–12, Jun. 2015.

[11] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," in *Proc. IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004.

[12] F. Meyer, O. Hlinka, and F. Hlawatsch, "Sigma point belief propagation," *IEEE Signal Process. Lett.*, vol. 21, no. 2, pp. 145–149, Feb. 2014.

[13] W. Sun and K.-C. Chang, "Unscented message passing for arbitrary continuous variables in Bayesian networks," in *Proc. 22nd Nat. Conf. Artif. Intell.*, 2007, pp. 1902–1903.

[14] I. Arasaratnam, S. Haykin, and R. Elliott, "Discrete-time nonlinear filtering algorithms using Gauss–Hermite quadrature," *Proc. IEEE*, vol. 95, no. 5, pp. 953–977, May 2007.

[15] A. F. García-Fernández, L. Svensson, M. R. Morelande, and S. Särkkä, "Posterior linearization filter: Principles and implementation using sigma points," *IEEE Trans. Signal Process.*, vol. 63, no. 20, pp. 5561–5573, Oct. 2015.

[16] A. F. García-Fernández, L. Svensson, and S. Särkkä, "Iterated posterior linearization smoother," *IEEE Trans. Autom. Control*, vol. 62, no. 4, pp. 2056–2063, Apr. 2017.

[17] I. Güvenç and C. C. Chong, "A survey on TOA based wireless localization and NLOS mitigation techniques," *IEEE Commun. Surv. Tuts.*, vol. 11, no. 3, pp. 107–124, Third Quart. 2009.

[18] D. M. Malioutov, J. K. Johnson, and A. S. Willsky, "Walk-sums and belief propagation in Gaussian graphical models," *J. Mach. Learn. Res.*, vol. 7, pp. 2031–2064, 2006.

[19] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Bethe free energy, Kikuchi approximations, and belief propagation algorithms," Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, Tech. Rep. TR-2001-16, 2001.

[20] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge, MA, USA: Cambridge Univ. Press, 2013.

[21] D. Bickson, "Gaussian belief propagation: Theory and application," Ph.D. dissertation, The Hebrew Univ. Jerusalem, Jerusalem, Israel, 2008.

[22] Q. Su and Y.-C. Wu, "On convergence conditions of Gaussian belief propagation," *IEEE Trans. Signal Process.*, vol. 63, no. 5, pp. 1144–1155, Mar. 2015.

[23] P. Tichavsky, C. H. Muravchik, and A. Nehorai, "Posterior Cramér–Rao bounds for discrete-time nonlinear filtering," *IEEE Trans. Signal Process.*, vol. 46, no. 5, pp. 1386–1396, May 1998.