Rao–Blackwellized Posterior Linearization Backward SLAM

Ángel F. García-Fernández, Roland Hostettler, and Simo Särkkä

DOI:

10.1109/TVT.2019.2903569

Rao-Blackwellised posterior linearisation backward SLAM

Ángel F. García-Fernández, Roland Hostettler, *Member, IEEE*, Simo Särkkä, *Senior Member, IEEE*

***Abstract*—This paper proposes the posterior linearisation backward simultaneous localisation and mapping (PLB-SLAM) algorithm for batch SLAM problems. Based on motion and landmark measurements, we aim to estimate the trajectory of the mobile agent and the landmark positions using an approximate Rao-Blackwellised Monte Carlo solution, as in FastSLAM. PLB-SLAM improves the accuracy of current FastSLAM solutions due to two key aspects: smoothing of the trajectory distribution via backward trajectory simulation and the use of iterated posterior linearisation to obtain Gaussian approximations of the distribution of the landmarks. PLB-SLAM is assessed via numerical simulations and real experiments for indoor localisation and mapping of radio beacons using a smartphone, Bluetooth beacons, and Wi-Fi access points.**

***Index Terms*—Simultaneous localisation and mapping, backward simulation, posterior linearisation, Rao-Blackwellisation, Bluetooth beacons, Wi-Fi access points, smartphone.**

I. Introduction

Simultaneous localisation and mapping (SLAM) consists of inferring the states of a mobile agent and static landmarks based on sensor measurements, which can be taken by the mobile agent and/or the landmarks, depending on the application. SLAM was developed in the robotics area [1] but also has important applications in related fields such as computer vision [2], [3], autonomous driving [4], unmanned aerial vehicles [5], and target tracking and localisation [6], [7]. With the widespread use of smartphones and ubiquitous radio beacons, such as Wi-Fi access points, Bluetooth beacons and radio-frequency identification (RFID) tags [8]–[13], SLAM techniques can also be used to localise a smartphone (agent) and map these signal sources (landmarks). This application of SLAM is specially important in indoor localisation, where global positioning system does not work well.

The SLAM problem is usually posed in a Bayesian framework [1]. Here, all information regarding the agent trajectory and the map is given by the conditional density of the agent trajectory and the map given the measurements, which is referred to as posterior density. In practice, due to non-Gaussian/nonlinear models, this density cannot be calculated in closed-form so it is approximated. An a priori simpler option is to just focus on the marginal posterior density of the agent state at the current time, instead of the trajectory, and the map. This marginal posterior density considers a variable with lower dimensionality (agent state at the current time vs. trajectory) but contains less information than the posterior density that includes the trajectory. The marginal posterior can be approximated using Bayesian filtering techniques [14], such as the extended Kalman filter (EKF), as in EKF-SLAM [1], or the unscented Kalman filter (UKF), as in UKF-SLAM [15].

Nevertheless, the posterior density over the trajectory and the map factorises in a way that is quite appealing for computation. Given the agent trajectory and the measurements, the density of each landmark becomes independent of the rest [16], [17]. FastSLAM exploits this factorisation by using an approximate Rao-Blackwellised particle filter [18]. In this algorithm, the posterior density over the trajectory is represented by weighted trajectory samples, and, for each trajectory sample, the density of the landmark is approximated as a Gaussian, which can be obtained using Gaussian filtering methods such as the EKF [16], [17] or the UKF [19].

In fact, these Gaussian filters approximate the nonlinear functions as affine functions plus Gaussian noise, which represents the linearisation error [20]. For example, the EKF linearises the measurement function while filtering using Taylor series around the predicted mean, while the UKF linearises the measurement function using statistical linear regression (SLR) with respect to the predicted density. The quality of the FastSLAM approximation therefore depends on how accurately the particles represent the trajectory posterior and how accurately the measurement functions are linearised for each trajectory sample. However, neither of these two types of approximations in FastSLAM properly takes into account all measurements when performing these approximations so there is room for improvement. First, particle filters severely suffer from path degeneracy for long time sequences [18]. That is, for long enough time sequences, all trajectory samples have one single ancestor at some point in the past. Second, the measurement function linearisations used by Kalman-filter-type algorithms, such as the EKF or UKF, do not take into account all received measurements [20].

In this paper, we address the two above-mentioned weaknesses of FastSLAM in a batch solution to the SLAM problem, in which the considered time frame is fixed. We first propose the use of backward simulation, which is based on running a particle filter forward and then a backward pass, to obtain non-

A. F. García-Fernández is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3GJ, United Kingdom. He is also with the ARIES Research Center, Universidad Antonio de Nebrija, Madrid, Spain (email: angel.garcia-fernandez@liverpool.ac.uk). R. Hostettler and S. Särkkä are with the Department of Electrical Engineering and Automation, Aalto University, 02150 Espoo, Finland (emails: {roland.hostettler, simo.sarkka}@aalto.fi).

Financial support by the Academy of Finland under grant number 295080 (CrowdSLAM) is hereby gratefully acknowledged. The authors would also like to thank IndoorAtlas for providing the data of the Wi-Fi experiments.

degenerate trajectory samples [21]–[23]. Backward simulation for SLAM was used in [24], though only for linear measurement models. An approximate Rao-Blackwellised smoother for nonlinear models is introduced in [22], although the solution requires the introduction of artificial priors. Instead, in this paper, we develop a backward simulator for SLAM that aims to use the optimal linearisations of the nonlinear measurement functions in mean square error sense for each trajectory sample, without artificial priors. This linearisation will turn out to be essential for accurate landmark estimation.

That is, once we have addressed the problem of trajectory degeneracy using backward simulation, approximate Rao-Blackwellisation requires the linearisation of the measurement functions for the new trajectory samples. In this paper, we propose the use of the posterior linearisation technique [20], [25] to select such a linearisation. In posterior linearisation, the idea is to obtain the best possible linearisation of the nonlinear functions in a mean square error sense by taking all measurements into account. The linearisation parameters and the Gaussian noise, which characterises the approximation error, are given by SLR with respect to the posterior. Since we do not have access to the posterior, practical posterior linearisation techniques use an iterated scheme, in which we linearise the nonlinearities with respect to the current posterior approximation to obtain a new posterior approximation, which will be used in the next iteration.

The resulting SLAM algorithm that integrates backward simulation and posterior linearisation is referred to as posterior linearisation backward SLAM (PLB-SLAM). PLB-SLAM is analysed in numerical simulations and in real indoor experiments in which we apply SLAM to track a pedestrian holding a smartphone and map the locations of Bluetooth beacons and Wi-Fi access points.

The rest of the paper is organised as follows. We formulate the problem in Section II. We review FastSLAM implementations based on sigma-points in Section III. Posterior linearisation backward SLAM is explained in Section IV. We show simulation and experimental results in Sections V and VI, respectively. Finally, conclusions are drawn in Section VII.

II. Problem formulation

In this paper, we aim to estimate the (mobile) agent state at all time steps and the landmark positions given all measurements taken by the agent. We pose this problem in the Bayesian framework in Section II-A and explain the type of solution we seek in this paper in Section II-B.

A. SLAM problem

The agent state at time k is $x_k \in \mathbb{R}^{n_x}$, which usually contains kinematic information such as position and velocity in a 2-D or 3-D environment. Furthermore, we assume there are M landmarks such that the multi-landmark state is $m = \left[\left(m^1\right)^T, ..., \left(m^M\right)^T\right]^T$, where $m^j \in \mathbb{R}^{n_m}$ is the state of the jth landmark, which usually denotes its position. At each time step, the agent takes a measurement of each landmark and we assume that the correspondence between measurements and landmarks is known [16]. For instance, this association is known if landmarks are beacons or Wi-Fi access points, as in our experimental set-up in Section VI.

The measurement $z_k^j \in \mathbb{R}^{n_z}$ at time k from landmark j is

$$z_k^j = h\left(x_k, m^j\right) + r_k^j \tag{1}$$

where $h(\cdot)$ is a given function and r_k^j is independent zero-mean Gaussian noise with covariance matrix R_k^j. This measurement model gives rise to the likelihood $p\left(z_k^j \mid x_k, m^j\right) = \mathcal{N}\left(z_k^j; h\left(x_k, m^j\right), R_k^j\right)$, which denotes a Gaussian density with mean $h\left(x_k, m^j\right)$ and covariance matrix R_k^j evaluated at z_k^j. Vector $z_k = \left[\left(z_k^1\right)^T, ..., \left(z_k^M\right)^T\right]^T$ represents all landmark measurements at time k, and measurements z_k^j are conditionally independent given x_k and m^j.

In addition, we assume that, at time k, we have a motion/odometry measurement $y_k \in \mathbb{R}^{n_y}$ of the form

$$y_k = h^y\left(x_k - x_{k-1}\right) + \eta_k \tag{2}$$

where $h^y(\cdot)$ is a function and η_k are independent zero-mean Gaussian noises with covariance matrix Θ. This measurement model gives rise to the likelihood $p\left(y_k | x_k, x_{k-1}\right) = \mathcal{N}\left(y_k; h^y\left(x_k - x_{k-1}\right), \Theta\right)$ and we assume it is independent of the rest of the measurements given x_k and x_{k-1}.

We also assume that the agent moves with a transition density $p\left(x_k \mid x_{k-1}\right)$ and that at time 0 the agent state and landmarks are independent with densities $p\left(x_0\right)$ and

$$p(m) = \prod_{j=1}^{M} p\left(m^j\right) = \prod_{j=1}^{M} \mathcal{N}\left(m^j; \overline{m}^j, P^j\right), \tag{3}$$

respectively. Given the measurements up to the final time step K, our objective is to compute the posterior PDF of the agent trajectory $x_{0:K} = \left[\left(x_0\right)^T, ..., \left(x_K\right)^T\right]^T$ and landmarks. This density is given by Bayes' rule

$$\begin{aligned} &p\left(x_{0:K}, m \mid z_{1:K}, y_{1:K}\right) \\ &\quad \propto \prod_{k=1}^{K} \left[p\left(z_k \mid x_k, m\right) p\left(y_k \mid x_k, x_{k-1}\right) p\left(x_k \mid x_{k-1}\right)\right] \\ &\qquad \times p\left(x_0\right) p(m) \end{aligned} \tag{4}$$

where $\propto$ denotes proportionality, and all the densities on the right-hand side of this equation are part of the problem formulation.

Computing (4) is intractable for nonlinear/non-Gaussian systems so we require approximations. In this paper, we pursue approximations that make use of the factorisation

$$\begin{aligned} p\left(x_{0:K}, m \mid z_{1:K}, y_{1:K}\right) = p\left(x_{0:K} \mid z_{1:K}, y_{1:K}\right) \\ \times p\left(m \mid z_{1:K}, x_{1:K}\right) \end{aligned} \tag{5}$$

where the distribution of m is conditionally independent of $y_{1:K}$ given $x_{1:K}$. As in FastSLAM [16], [17], this factorisation is of interest as the distribution of the map given the measurements and agent trajectory is made of independent PDFs over the landmarks

$$p\left(m | z_{1:K}, x_{1:K}\right) = \prod_{j=1}^{M} p\left(m^j \mid z_{1:K}^j, x_{1:K}\right). \tag{6}$$

This independence property is beneficial from a computational point of view as it lowers the number of parameters of the landmark distribution compared to a joint distribution of the landmarks [1]. In addition, using Bayes' rule for each factor in (6) yields

$$p\left(m^j \mid z_{1:K}^j, x_{1:K}\right) \propto p\left(z_{1:K}^j \mid m^j, x_{1:K}\right) p\left(m^j \mid x_{1:K}\right) = \prod_{k=1}^{K} p\left(z_k^j \mid m^j, x_k\right) p\left(m^j\right) \quad (7)$$

where $p\left(m^j \mid x_{1:K}\right) = p\left(m^j\right)$ as the landmark positions and agent trajectory are a priori independent, see (4). Therefore, the posterior distribution for landmark j given the agent trajectory can be obtained by considering the prior and updating it with K likelihoods $p\left(z_k^j | m^j, x_k^i\right)$.

B. Considered solution

In this subsection, we explain the type of approximations we consider to approximate (4). As in FastSLAM, we aim to have a Monte Carlo approximation to $p\left(x_{0:K} \mid z_{1:K}, y_{1:K}\right)$ of the form

$$p\left(x_{0:K} \mid z_{1:K}, y_{1:K}\right) \approx \sum_{i=1}^{N} w_K^i \delta\left(x_{0:K} - x_{0:K}^i\right) \quad (8)$$

where $\delta(\cdot)$ is the Dirac delta, $x_{0:K}^i$ is the ith trajectory particle at time K, w_K^i is its weight and N is the number of particles. Then, using (7), for each trajectory $x_{0:K}^i$, the posterior PDF of landmark j is

$$p\left(m^j \mid z_{1:K}^j, x_{1:K}^i\right) \propto \prod_{k=1}^{K} p\left(z_k^j \mid m^j, x_k^i\right) p\left(m^j\right), \quad (9)$$

which can be calculated by updating $p\left(m^j\right)$, which is given in (3), using the measurement model (1) with known agent trajectory $x_{0:K}^i$.

Due to the nonlinear measurement functions, each of the updates in (9) does not have a closed-form expression. One way to deal with nonlinearities in Gaussian updates is to approximate the nonlinear functions as affine functions with additive noise [20]. Then, we consider the approximation

$$h_k^{i,j}\left(m^j\right) \triangleq h\left(x_k^i, m^j\right) \approx H_k^{i,j} m^j + b_k^{i,j} + e_k^{i,j} \quad (10)$$

where $H_k^{i,j} \in \mathbb{R}^{n_z \times n_m}$, $b_k^{i,j} \in \mathbb{R}^{n_z \times 1}$ and $e_k^{i,j}$ is a zero-mean Gaussian noise with covariance matrix $\Omega_k^{i,j} \in \mathbb{R}^{n_z \times n_z}$ such that the likelihood $p\left(z_k^j \mid m^j, x_k^i\right) \approx \mathcal{N}\left(z_k^j; H_k^{i,j} m^j + b_k^{i,j}, R_k^j + \Omega_k^{i,j}\right)$. Then, under approximation (10), the posterior PDF of landmark j, see (9), is Gaussian with mean $\overline{m}_K^{i,j}$ and covariance matrix $P_K^{i,j}$, which can be obtained by performing K Kalman filter updates with an affine measurement [20, Eq. (6)-(7)]. Therefore, we can obtain $\overline{m}_K^{i,j}$ and $P_K^{i,j}$ with the recursion

$$\overline{m}_k^{i,j} = \overline{m}_{k-1}^{i,j} + \Phi_k^{i,j}\left(S_k^{i,j}\right)^{-1}\left(z_k^j - H_k^{i,j}\overline{m}_{k-1}^{i,j} - b_k^{i,j}\right) \quad (11)$$

$$P_k^{i,j} = P_{k-1}^{i,j} - \Phi_k^{i,j}\left(S_k^{i,j}\right)^{-1}\left(\Phi_k^{i,j}\right)^T \quad (12)$$

where k goes from 1 to K, $\overline{m}_0^{i,j} = \overline{m}^j$ and $P_0^{i,j} = P^j$ and

$$S_k^{i,j} = H_k^{i,j} P_{k-1}^{i,j}\left(H_k^{i,j}\right)^T + \Omega_k^{i,j} + R_k^j, \quad (13)$$

$$\Phi_k^{i,j} = P_{k-1}^{i,j}\left(H_k^{i,j}\right)^T. \quad (14)$$

Finally, the joint posterior approximation becomes

$$p\left(x_{0:K}, m \mid z_{1:K}, y_{1:K}\right) \approx \sum_{i=1}^{N} w_K^i \delta\left(x_{0:K} - x_{0:K}^i\right) \times \prod_{j=1}^{M} \mathcal{N}\left(m^j; \overline{m}_K^{i,j}, P_K^{i,j}\right). \quad (15)$$

It should be noted that, in this type of solution to the SLAM problem, the accuracy of the approximation of the posterior (15) only depends on how we obtain the Monte Carlo approximation (8) and the parameters $H_k^{i,j}$, $b_k^{i,j}$ and $\Omega_k^{i,j}$ of the approximations (10). In this work, we will provide a way of selecting these parameters that outperforms the previous methods in the literature. Before doing so, we proceed to briefly review statistical linear regression and how other approaches in the literature obtain the approximations (8) and (10).

C. Background work

In the original FastSLAM algorithm [16], the Monte Carlo samples (8) are obtained via sequential Monte Carlo sampling using an importance density that samples from the transition density $p\left(x_k \mid x_{k-1}\right)$, performing smoothing while filtering [26]. In addition, FastSLAM sets $\Omega_k^{i,j} = 0$ and the parameters $H_k^{i,j}$ and $b_k^{i,j}$ are obtained while filtering using a first-order Taylor series approximation of $h\left(x_k^i, m^j\right)$ at the current landmark mean, as in the extended Kalman filter [14]. In FastSLAM 2.0 [17], the importance density is changed so that the current measurement is taken into account to draw new samples but the parameters of the approximation (10) are chosen in the same way as in FastSLAM.

In unscented FastSLAM [19], which will be explained thoroughly in the next section, $H_k^{i,j}$, $b_k^{i,j}$ and $\Omega_k^{i,j}$ are obtained through the unscented Kalman filter (UKF) [27]. In other words, $H_k^{i,j}$, $b_k^{i,j}$ and $\Omega_k^{i,j}$ are selected using SLR, which will be explained in Section III-A, with respect to the predicted densities at each time step using the unscented transform. This is equivalent to the choice of measurement linearisation parameters of the UKF, see [20, Sec. II.A] for a full discussion on how different nonlinear Kalman filters select the parameters of the measurement linearisation at each update step. In unscented FastSLAM, the current measurement is also taken into account in the importance density to draw new particles. Apart from the unscented transform, we can use other sigma-point methods [28], [29].

However, there are two drawbacks of the above mentioned FastSLAM methods:

- D1: Trajectory samples degenerate for long time sequences. For long enough trajectories, all trajectory samples will have a common ancestor in the past due to the use of a particle filter to draw the samples.

- D2: The selection of the parameters $H_k^{i,j}$, $b_k^{i,j}$ and $\Omega_k^{i,j}$ does not take into account knowledge of all measurements. Therefore, all available information is not considered to choose the parameters that determine the quality of the posterior approximation, given in (4).

In this work, we propose a batch solution to SLAM that tackles D1 and D2. We address D1 by using particle smoothing by backward simulation, whose benefits compared to forward filtering become more significant for long sequences (high K). We address D2 by using posterior linearisation techniques, in which we use all available information to select the SLR parameters $H_k^{i,j}$, $b_k^{i,j}$ and $\Omega_k^{i,j}$ [20], [25]. The improvement of posterior linearisation techniques compared to non-iterated Kalman filters, such as UKF, is expected to be higher for high nonlinearities and low measurement noise. In this case, non-iterated Kalman filters do not provide an accurate approximation of the posterior, as proved by the Kullback-Leibler divergence in [30], but posterior linearisation can.

III. Sigma-point FastSLAM filtering algorithm

In this section, we review two important concepts that are useful for the rest of the paper, SLR in Section III-A and a general sigma-point FastSLAM filtering algorithm, from an SLR perspective, in Section III-B.

A. Statistical linear regression

In this section we review SLR, which plays an important role in this paper.

Definition 1. *Given a function $h(\cdot)$ and a PDF $p(\cdot)$ of a random vector, whose first two moments are $\overline{x}$ and P, the SLR of $h(\cdot)$ with respect to $p(\cdot)$ is given by* [31]

$$H^+ = \Psi^T P^{-1}, \quad b^+ = \overline{z} - H^+ \overline{x} \tag{16}$$

$$\Omega^+ = \Phi - H^+ P \left(H^+\right)^T \tag{17}$$

where

$$\overline{z} = \int h(x)\, p(x)\, dx \tag{18}$$

$$\Psi = \int (x - \overline{x}) \left(h(x) - \overline{z}\right)^T p(x)\, dx \tag{19}$$

$$\Phi = \int \left(h(x) - \overline{z}\right) \left(h(x) - \overline{z}\right)^T p(x)\, dx. \tag{20}$$

The approximation $h(x) \approx H^+ x + b^+$ minimises the mean square error (MSE) with respect to $p(\cdot)$ and Ω^+ is the MSE matrix [31]. In general, the moments (18)-(20) cannot be computed in closed-form but they can be approximated using sigma-point methods such as the unscented transform [27]. We first select n_s sigma-points $\mathcal{X}_1, ..., \mathcal{X}_{n_s}$ and weights $\omega_1, ..., \omega_{n_s}$, which match the moments $\overline{x}$ and P, according to a suitable sigma-point method [14], [27]. Then, the SLR is performed as indicated in Algorithm 1.

Algorithm 1 Statistical linear regression using sigma-points

Input: Function $h(\cdot)$ and first two moments $\overline{x}$, P of a PDF $p(\cdot)$.
Output: SLR parameters $\left(H^+, b^+, \Omega^+\right)$.

- Select n_s sigma-points $\mathcal{X}_1, \ldots, \mathcal{X}_{n_s}$ and weights $\omega_1, \ldots, \omega_{n_s}$ according to $\overline{x}$ and P.
- Transform the sigma-points $\mathcal{Z}_j = h(\mathcal{X}_j) \quad j = 1, \ldots, n_s$.
- Approximate the moments (18)-(20) as

$$\overline{z} \approx \sum_{j=1}^{n_s} \omega_j \mathcal{Z}_j, \; \Psi \approx \sum_{j=1}^{n_s} \omega_j \left(\mathcal{X}_j - \overline{x}\right) \left(\mathcal{Z}_j - \overline{z}\right)^T$$

$$\Phi \approx \sum_{j=1}^{n_s} \omega_j \left(\mathcal{Z}_j - \overline{z}\right) \left(\mathcal{Z}_j - \overline{z}\right)^T$$

- Obtain H^+, b^+, Ω^+ from (16)-(17).

B. FastSLAM filtering algorithm

In this section, we describe a general sigma-point FastSLAM filtering algorithm, because the first step of posterior linearisation backward SLAM is to run this type of algorithm.

FastSLAM assumes that, at time $k-1$, we have a density of the form (15), but with time index $k-1$ rather than K. The objective in FastSLAM is to approximate $p(x_{0:k}, m \mid z_{1:k}, y_{1:k})$ in the same form as in (15). We first need to obtain (weighted) samples from the PDF

$$p(x_{0:k} \mid z_{1:k}, y_{1:k}) \propto p(z_k \mid x_{0:k}, z_{1:k-1})\, p(y_k \mid x_k, x_{k-1}) \times p(x_k \mid x_{k-1})\, p(x_{0:k-1} \mid z_{1:k-1}, y_{1:k-1}). \tag{21}$$

The PDF (21) is sampled with an importance density $q_k(\cdot)$, which can depend on current and past measurements. In order to apply sequential Monte Carlo sampling, the importance density factorises as $q_k(x_{0:k}) = q_k(x_k \mid x_{0:k-1})\, q_{k-1}(x_{0:k-1})$. Then, we can draw a sample from $q_k(\cdot)$ by drawing a sample from $q_k(x_k \mid x_{0:k-1})$ and appending it to a previous sample from $q_{k-1}(\cdot)$. The ith sample is denoted as $x_{0:k}^i = \left[\left(x_0^i\right)^T, ..., \left(x_k^i\right)^T\right]^T$ where $x_k^i \sim q_k\left(x_k \mid x_{0:k-1}^i\right)$. The corresponding particle weight is given by

$$w_k^i \propto \frac{p\left(z_k \mid x_{0:k}^i, z_{1:k-1}\right) p\left(y_k \mid x_k^i, x_{k-1}^i\right)}{q_k\left(x_k^i \mid x_{0:k-1}^i\right)} \times p\left(x_k^i \mid x_{k-1}^i\right) w_{k-1}^i. \tag{22}$$

However, $p\left(z_k \mid x_{0:k}^i, z_{1:k-1}\right)$ does not admit a closed-form expression and is approximated as

$$\begin{aligned} &p\left(z_k \mid x_{0:k}^i, z_{1:k-1}\right) \\ &= \int p\left(z_k \mid x_k^i, m\right) p\left(m \mid x_{0:k-1}^i, z_{1:k-1}\right) dm \\ &\approx \prod_{j=1}^{M} \int \mathcal{N}\left(z_k^j; h\left(x_k^i, m^j\right), R\right) \mathcal{N}\left(m^j; \overline{m}_{k-1}^{i,j}, P_{k-1}^{i,j}\right) dm^j \end{aligned} \tag{23}$$

where the Gaussian approximation is given in (15). This integral can be solved by using the enabling approximation (10), where typical selections of $H_k^{i,j}$, $b_k^{i,j}$, $\Omega_k^{i,j}$ are obtained by analytical linearisation (first-order Taylor series) of $h\left(x_k^i, m^j\right)$ (EKF) [16], which sets $\Omega_k^{i,j} = 0$, or by SLR of

$h\left(x_k^i, m^j\right)$ w.r.t. $\mathcal{N}\left(\cdot; \overline{m}_{k-1}^{i,j}, P_{k-1}^{i,j}\right)$ using sigma-points [19], which usually deals better with nonlinearities than analytical linearisations [30]. Under approximation (10), (23) has a closed-form expression given by

$$p\left(z_k \mid x_{0:k}^i, z_{1:k-1}\right) \approx \prod_{j=1}^{M} \mathcal{N}\left(z_k^j; H_k^{i,j}\overline{m}_{k-1}^{i,j} + b_k^{i,j}, S_k^{i,j}\right) \tag{24}$$

where $S_k^{i,j}$ is given by (13).

Under approximation (10), we can also compute the updated distribution of each landmark and particle using the Kalman filter update

$$\begin{aligned} p\left(m^j \mid x_{0:k}^i, z_{1:k}\right) &\propto p\left(z_k^j \mid x_k^i, m^j\right) p\left(m^j \mid x_{0:k-1}^i, z_{1:k-1}\right) \\ &\propto \mathcal{N}\left(m^j; \overline{m}_k^{i,j}, P_k^{i,j}\right) \end{aligned} \tag{25}$$

where $\overline{m}_k^{i,j}$ and $P_k^{i,j}$are computed using (11) and (12). With (24), we can compute the updated weights (22), which complete the update step. Finally, the pseudocode of sigma-point FastSLAM is given in Algorithm 2.

Algorithm 2 Pseudocode for sigma-point FastSLAM

- Draw N samples $x_0^1, ..., x_0^N$ from $p\left(x_0\right)$ with $w_0^i = \frac{1}{N}$ for all i.
- Set $\overline{m}_0^{i,j} = \overline{m}^i$, $P_0^{i,j} = P^j$ for all i, j
for $k = 1$ to K **do**
  for $i = 1$ to N **do** ▷ We sample the current state
    - Draw $x_k^i \sim q\left(x_k^i \mid x_{0:k-1}^i\right)$.
    for $j = 1$ to M **do**
      - Select $H_k^{i,j}, b_k^{i,j}, \Omega_k^{i,j}$: SLR of $h\left(x_k^i, \cdot\right)$ w.r.t. $\mathcal{N}\left(\cdot; \overline{m}_{k-1}^{i,j}, P_{k-1}^{i,j}\right)$, see Alg. 1.
      - Update the map for landmark j by calculating $\overline{m}_k^{i,j}$ and $P_k^{i,j}$, as in (25).
    end for
    - Compute $p\left(z_k | x_{0:k}^i, z_{1:k-1}\right)$ using (24).
    - Compute w_k^i using (22).
  end for
  - Normalise particle weights and resample if necessary by calculating effective sampling size [14].
end for

IV. Posterior linearisation backward SLAM

In this section, we propose the posterior linearisation backward SLAM (PLB-SLAM) algorithm to tackle drawbacks D1 and D2 of FastSLAM using a batch solution. The general idea of PLB-SLAM is to first run a forward pass, which consists of a sigma-point FastSLAM algorithm, explained in Section III-B. Then, we obtain non-degenerate trajectory samples using backward simulation, explained in Section IV-A. Finally, as indicated in Section IV-B, we use iterated posterior linearisation to obtain the approximation (10) for each trajectory sample. The steps of PLB-SLAM are summarised in Algorithm 3. Finally, a discussion on the applicability of PLB-SLAM is provided in Section IV-C.

A. Backward simulation

Given the Rao-Backwellised Monte Carlo approximations to all the filtering distributions $p\left(x_{0:k}, m \mid z_{1:k}, y_{1:k}\right)$ for $k = 0, \ldots, K$, which have the form (15), we can obtain non-degenerate, evenly weighted samples from $p\left(x_{0:K} \mid z_{1:K}, y_{1:K}\right)$ by performing backward simulation [21], [22]. The backward simulation approximation is given by

$$p\left(x_{0:K} \mid z_{1:K}, y_{1:K}\right) \approx \frac{1}{\tilde{N}} \sum_{i=1}^{\tilde{N}} \delta\left(x_{0:K} - \tilde{x}_{0:K}^i\right) \tag{26}$$

where $\tilde{N}$ is the number of backward trajectories determined by the user and $\tilde{x}_{0:K}^i$ is the ith backward trajectory.

In the considered solution to SLAM, we perform an approximate Rao-Blackwellisation for the landmark positions so we need to take this into account in backward simulation [22]. In the following, we provide two propositions that indicate how backward simulation is performed in our problem.

Proposition 2. *Given the filtering distribution $p\left(x_{0:k}, m \mid z_{1:k}, y_{1:k}\right)$ at time k of the form (15) and a sample $\tilde{x}_{k+1:K}$ from $p\left(x_{k+1:K} \mid z_{1:K}, y_{1:K}\right)$, $\left(x_{0:k}^i, \tilde{x}_{k+1:K}\right)$ is a sample from $p\left(x_{0:K} \mid z_{1:K}, y_{1:K}\right)$ if particle $x_{0:k}^i$ is chosen with probability*

$$\tilde{w}^i \propto w_k^i p\left(\widetilde{x}_{k+1} \mid x_k^i\right) p\left(y_{k+1} \mid \widetilde{x}_{k+1}, x_k^i\right) \prod_{j=1}^{M} \xi_k^{i,j} \tag{27}$$

$$\xi_k^{i,j} = \int \mathcal{N}\left(m^i; \overline{m}_k^{i,j}, P_k^{i,j}\right) \prod_{p=k+1}^{K} p\left(z_p^j \mid \widetilde{x}_p, m^j\right) dm^j. \tag{28}$$

Proposition 2 is proved in Appendix A. In the next proposition, we indicate how $\xi_k^{i,j}$ is approximated using the linearised measurement model.

Proposition 3. *Let $\tilde{x}_p$ for $p \in \{k+1, ..., K\}$ denote a particle of the filtering recursion at time step p and let $\widetilde{H}_p^j$, $\widetilde{b}_p^j$ and $\widetilde{\Omega}_p^j$ denote the linearisation parameters obtained in filtering for $\tilde{x}_p$ and landmark j such that we assume*

$$p\left(z_p^j \mid \widetilde{x}_p, m^j\right) = \mathcal{N}\left(z_p^j; \widetilde{H}_p^j m^j + \widetilde{b}_p^j, R_p^j + \widetilde{\Omega}_p^j\right). \tag{29}$$

Then, factor $\xi_k^{i,j}$ in (28) is given by

$$\xi_k^{i,j} \propto \left| I + \left(P_k^{i,j}\right)^{T/2} L_k^j \left(P_k^{i,j}\right)^{1/2} \right|^{-1/2} \exp\left(-\kappa_k^{i,j}/2\right) \tag{30}$$

$$\begin{aligned} \kappa_k^{i,j} = &\left(\overline{m}_k^{i,j}\right)^T L_k^j \overline{m}_k^{i,j} - 2\left(\overline{m}_k^{i,j}\right)^T l_k^j - \left(L_k^j \overline{m}_k^{i,j} - l_k^j\right)^T \\ &\times \left(P_k^{i,j}\right)^{1/2} \left(I + \left(P_k^{i,j}\right)^{T/2} L_k^j \left(P_k^{i,j}\right)^{1/2}\right)^{-1} \end{aligned}$$

Algorithm 3 Steps of posterior linearisation backward SLAM

- Run a FastSLAM filtering algorithm: run Algorithm 2.
for $i = 1$ to $\tilde{N}$ **do** ▷ $\tilde{N}$ is the number of backward samples
  - Obtain a trajectory sample $\tilde{x}_{0:K}^i$ of the mobile agent using backward simulation: run Algorithm 4.
  for $j = 1$ to M **do**
    - Apply iterated posterior linearisation on the landmarks and sample $\tilde{x}_{0:K}^i$: run Algorithm 5.
  end for
end for

$$\times \left(P_k^{i,j}\right)^{T/2} \left(L_k^j \overline{m}_k^{i,j} - l_k^j\right) \tag{31}$$

where $\left(P_k^{i,j}\right)^{1/2}$ *is the Cholesky factorisation of* $P_k^{i,j} = \left(P_k^{i,j}\right)^{1/2} \left(P_k^{i,j}\right)^{T/2}$ *and* l_k^j *and* L_k^j *are calculated recursively*

$$l_k^j = l_{k+1}^j + \left(\widetilde{H}_{k+1}^j\right)^T \left(R_{k+1}^j + \widetilde{\Omega}_{k+1}^j\right)^{-1} \left(z_{k+1}^j - \widetilde{b}_{k+1}^j\right) \tag{32}$$

$$L_k^j = L_{k+1}^j + \left(\widetilde{H}_{k+1}^j\right)^T \left(R_{k+1}^j + \widetilde{\Omega}_{k+1}^j\right)^{-1} \widetilde{H}_{k+1}^j \tag{33}$$

by initialising them to $l_K^j = 0$ *and* $L_K^j = 0$.

Proposition 3 is proved in Appendix B. Note that the two propositions, once the system is linearised and without taking into account the motion measurements, correspond to a particular case of more general Rao-Blackwellised linear smoothers [22]. Nevertheless, we include the detailed derivations due to the importance of the specific formulation in SLAM and to explicitly consider the motion measurements. The resulting backward trajectory simulator is summarised in Algorithm 4. It is important to notice, that once we have performed filtering, we can sample backward trajectories in parallel by running Algorithm 4 in different processing units. Obtaining each trajectory sample using Algorithm 4 has complexity of $O(KN)$.

Algorithm 4 Backward trajectory simulation for SLAM

Input: Rao-Blackwellised particle filtering densities. Measurement function linearisation parameters.
Output: Backward trajectory sample $\tilde{x}_{0:K}$.
- Choose $\tilde{x}_K = x_K^i$ with probability w_K^i.
- Set $l_K^j = 0$ and $L_K^j = 0$ for $j = 1, ..., M$..
for $k = K-1$ to 0 **do**
  - Calculate $\xi_k^{i,j}$ using (30) for $i = 1, ..., N$ and $j = 1, ..., M$.
  - Calculate $\tilde{w}^i$ using (27) for $i = 1, ..., N$ and normalise.
  - Choose $\tilde{x}_k^i = x_k^i$ with probability $\tilde{w}^i$.
  - Compute l_k^j and L_k^j using (32) and (33) for $j = 1, ..., M$.
end for
- Return $\tilde{x}_{0:K} = (\tilde{x}_0, ..., \tilde{x}_K)$.

B. Posterior linearisation for the landmark distributions

Given the trajectory samples obtained by backward simulation, the next step is to obtain the posterior distributions for the landmarks $p\left(m^j \mid \tilde{x}_{0:K}^i, z_{1:K}, y_{1:K}\right)$. As the trajectory samples in backward simulation are different from the ones obtained while filtering, we must compute the distribution of the landmarks for each new trajectory. Given the agent trajectory, the posterior distribution of each landmark is independent of the rest of the landmarks, as indicated by (6), so we can update each landmark independently. The posterior of a landmark given a trajectory sample is given by (9), which can be computed in closed-form under the approximation (10).

For each trajectory $\tilde{x}_{0:K}^i$, the quality of the posterior approximation only depends on $H_k^{i,j}$, $b_k^{i,j}$, and $\Omega_k^{i,j}$ in (10). The main insight of posterior linearisation is that we want the approximation in (10) to be accurate in the area of interest, where the posterior has its mass. More specifically, given the measurements, the selected $H_k^{i,j}$ and $b_k^{i,j}$ are chosen to minimise the mean square error w.r.t. the measurement function $h_k^{i,j}\left(m^j\right) = h\left(\tilde{x}_k^i, m^j\right)$

$$\left(H_{k,+}^{i,j}, b_{k,+}^{i,j}\right) = \underset{\left(H_k^{i,j}, b_k^{i,j}\right)}{\arg\min}\, \mathrm{E}\left[(\cdot)^T \left(h_k^{i,j}\left(m^j\right) - H_k^{i,j} m^j - b_k^{i,j}\right)\right] \tag{34}$$

and $\Omega_k^{i,j}$ is the mean square error matrix of the resulting approximation [20]:

$$\Omega_{k,+}^{i,j} = \mathrm{E}\left[\left(h_k^{i,j}\left(m^j\right) - H_{k,+}^{i,j} m^j - b_{k,+}^{i,j}\right)(\cdot)^T\right] \tag{35}$$

where the previous expectations are taken w.r.t. to the posterior density $p\left(m^j \mid \tilde{x}_{0:K}^i, z_{1:K}, y_{1:K}\right)$ 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 $H_k^{i,j}$, $b_k^{i,j}$ and $\Omega_k^{i,j}$ using SLR w.r.t. $p\left(m^j \mid \tilde{x}_{0:K}^i, , z_{1:K}, y_{1:K}\right)$ [20, Sec. II].

The problem is that posterior linearisation is intractable, as we require the posterior to approximate it. Nevertheless, we can apply posterior linearisation in an iterated fashion, as in the iterated posterior linearisation filter (IPLF) [20]. That is, since we do not have the posterior, we perform SLR with respect to the current approximation of the posterior. At the end of each iteration, we expect to obtain an improved approximation of the posterior which means that it can be used to obtain a better SLR at the next iteration. The steps of the IPLF for updating a landmark position are given in Algorithm 5, where iterations only change the parameters in (10), not the prior.

Algorithm 5 has a computational complexity $O(KJ)$ and can be run in parallel for each trajectory sample and landmark. In addition, the for loop over all time steps in Algorithm 5 can be run in parallel, so it is quite suitable for parallel computing architectures, as the backward simulator in Section IV-A.

Finally, we would like to mention that posterior linearisation can also be applied while filtering in FastSLAM to select $H_k^{i,j}, b_k^{i,j}, \Omega_k^{i,j}$ in (10), which is used to compute the particle weights and update the landmark distribution, see (24) and (25). In other words, we can use the IPLF rather than a (non-iterated) sigma-point Kalman filter to select these parameters. However, for the received signal strength indicator measurements used in the experiments and simulations of the next section, the IPLF used in FastSLAM forward filtering did not improve performance, mainly due to possible multimodalities in the conditional PDFs of beacon positions that appear while filtering, but can be resolved in smoothing. Therefore, in the rest of this paper, we only consider the IPLF for updating the distribution of the landmarks for each backward trajectory sample.

C. Discussion on applicability

PLB-SLAM has the limitation that it is computationally heavy for online, real-time estimation due to the use of FastSLAM and then the backward simulator. This limitation is important in many applications, for instance, if the agent, such as a robot, unmanned aerial vehicle, or a self-driving vehicle, must estimate its position and the map in real time to navigate through the environment. In these scenarios, it is more convenient to use FastSLAM or other online SLAM

Algorithm 5 Iterated posterior linearisation filter for updating the state of landmark j

Input: Trajectory sample $\tilde{x}^i_{0:K}$, prior mean $\overline{m}^j$ and covariance matrix P^j, number of iterations J.
Output: Approximated mean $\overline{m}^{i,j}_K$ and covariance $P^{i,j}_K$ of $p\left(m^j \mid \tilde{x}^i_{0:K}, z_{1:K}, y_{1:K}\right)$.

- Set $\overline{u}^0 = \overline{m}^j$, $W^0 = P^j$.
for $p = 1$ to J **do**
  for $k = 1$ to K **do**
    - Calculate $H^{i,j}_k, b^{i,j}_k, \Omega^{i,j}_k$ using SLR of $h^{i,j}_k(\cdot)$ w.r.t. $\mathcal{N}\left(\cdot; \overline{u}^{p-1}, W^{p-1}\right)$, see Alg. 1.
  end for
  - Compute the updated moments $\overline{u}^p$ and W^p using prior moments $\overline{m}^j$, P^j and the current linearisation parameters $H^{i,j}_{1:K}, b^{i,j}_{1:K}, \Omega^{i,j}_{1:K}$ using (11)-(12).
end for
- Return $\overline{m}^{i,j}_K = \overline{u}^J$ and $P^{i,j}_K = W^J$.

methods. Nevertheless, there are other SLAM applications in which a real-time algorithm is not required and accuracy is more important.

In particular, we highlight crowdsourced map estimation [32]. Here, an agent enters an area, estimates its position and maps the positions of the beacons/landmarks using SLAM. Once the agent leaves the area, the estimated map is uploaded to a database that can then be shared and updated by other agents that perform SLAM as well. Here, there is no real-time processing requirement when the agent uploads the map to the database. Therefore, the agent can use FastSLAM while moving in the area of interest and, then, when leaving the area, the agent can perform PLB-SLAM to attain a higher accuracy in map estimation. Another option is to run PLB-SLAM off-line in the cloud.

We would also like to clarify that backward simulation can be used to deal with SLAM based on occupancy grids [33]. However, the Rao-Blackwellised backward simulator and posterior linearisation use Gaussian distributions to represent the map given the trajectory, so their use is not intended for occupancy-grid SLAM.

V. Simulations

In this section, we compare PLB-SLAM with previous sample-based SLAM solutions in a simulated scenario. In particular, we are especially interested in the accuracy of the estimated map, see Section IV-C. We have implemented the following sample-based SLAM algorithms: unscented FastSLAM and FastSLAM, which uses analytical linearisation (AL), which denotes a first-order Taylor series linearisation, instead of SLR, as in [16]. We have also implemented PLB-SLAM using AL rather than SLR to show the benefits of SLR. Moreover, we have implemented the iterated extended Kalman smoother (IEKS) [34] on the joint state defined by the agent and the map. The IEKS is a nonlinear least squares solver for the maximum a posteriori estimator, such as the algorithms in [35], [36].

A. Models

We consider that the state x_k at time k is $x_k = [p^x_k, v^x_k, p^y_k, v^y_k,]^T$ where $p_k = [p^x_k, p^y_k]^T$ is the position vector and $[v^x_k, v^y_k]^T$ is the velocity vector. The agent moves with the nearly-constant velocity model

$$p\left(x_k \mid x_{k-1}\right) = \mathcal{N}\left(x_k; Fx_{k-1}, Q\right) \tag{36}$$

where

$$F = I_2 \otimes \begin{pmatrix} 1 & \tau \\ 0 & 1 \end{pmatrix}, Q = qI_2 \otimes \begin{pmatrix} \tau^3/3 & \tau^2/2 \\ \tau^2/2 & \tau \end{pmatrix} \tag{37}$$

with $\otimes$ being the Kronecker product, I_n the identity matrix of size n, τ is the sampling period and q is a parameter of the model. We consider the agent trajectory in Figure 1, which contains 107 time steps with $\tau = 1\,\mathrm{s}$. Density $p(x_0)$ is Gaussian with a covariance matrix $0.01I_4$ and its mean is drawn from a Gaussian PDF with mean the true initial state and covariance matrix $0.01I_4$ at each Monte Carlo run.

Beacons are located at a constant height h_b with respect to the agent. Beacon j measures the received signal strength indicator (RSSI) according to the exponential path loss model [37]

$$h\left(x_k, m^j\right) = P_0 - 10\gamma \log_{10}\left(\sqrt{\|p_k - m^j\|^2 + h_b^2}\right) \tag{38}$$

where P_0 is the received power $(\mathrm{dB_m})$ at a reference distance of 1m, γ is the path loss exponent and m^j is the position of the jth beacon. It should be noted that the receiver knows what RSSI measurement corresponds to each beacon, which is the case for many radio signal-based sensors, for example, Bluetooth beacons, Wi-Fi, and mobile networks emitters. There are 10 beacons and all beacons have the same prior PDF with mean $\overline{m}^j = [9, 2]^T$ (m) and covariance $P^j = \mathrm{diag}\left([64, 4]\right)$ $\left(\mathrm{m}^2\right)$. We assume that the motion measurement in (2) consists of differences in positions such that

$$h^y\left(x_k - x_{k-1}\right) = H^y\left(x_k - x_{k-1}\right) \tag{39}$$

where $H^y = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$. The rest of the model parameters are: $\gamma = 1.5$, $P_0 = -70\mathrm{dB_m}$, $h_b = 0.4\mathrm{m}$, $R^j_k = 100$, $\Theta = 0.004I_2\mathrm{m}^2$ and $q = 0.25\mathrm{m}^2/\mathrm{s}^3$.

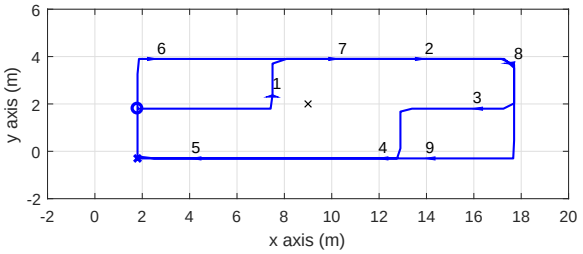

Figure 1: The agent initial and final positions are denoted as a blue circle and a cross, respectively. The agent position and its heading every ten time steps are marked as an arrow and a number, which indicates the sequence of agent positions. The prior mean $[9, 2]^T$ of the 10 beacons is shown as a black cross.

B. Algorithm implementation

In the forward FastSLAM filter, we need to determine the selection of $H^{i,j}_k, b^{i,j}_k, \Omega^{i,j}_k$, the importance density, and the resampling procedure. As in unscented FastSLAM [19], we

select $H_k^{i,j}, b_k^{i,j}, \Omega_k^{i,j}$ by SLR of the measurement function w.r.t. the prior approximated using the unscented transform. In these implementations, we use the unscented transform with a weight 1/3 for the sigma-point at the mean [27]. For the backward trajectory samples, we refine the selection of $H_k^{i,j}, b_k^{i,j}, \Omega_k^{i,j}$ by using the IPLF with the same unscented transform, see Algorithm 5. We will show results considering 1, 5 and 10 iterations of the IPLF, where we should note that the case with 1 iteration corresponds to the UKF.

For the importance density, we use the optimal importance density [14] for the motion measurements, as it can be computed in closed-form, and is given by

$$q_k\left(x_k \mid x_{0:k-1}^i\right) = \frac{p\left(y_k \mid x_k, x_{k-1}^i\right) p\left(x_k \mid x_{k-1}^i\right)}{\rho_i} \tag{40}$$

$$= \mathcal{N}\left(x_k; \bar{x}^q, P^q\right), \tag{41}$$

where

$$\bar{x}^q = F x_{k-1}^i + \Psi S^{-1}\left(y_k - H^y\left(F - I_4\right) x_{k-1}^i\right), \tag{42}$$

$$P^q = Q - \Psi S^{-1} \Psi^T, \quad \Psi = Q\left(H^y\right)^T, \tag{43}$$

$$S = H^y Q \left(H^y\right)^T + \Theta, \tag{44}$$

$$\rho_i = \mathcal{N}\left(y_k; H^y\left(F - I_4\right) x_{k-1}^i, S\right). \tag{45}$$

Substituting (40) into (22), the resulting updated weight is

$$w_k^i \propto p\left(z_k \mid x_{0:k}^i, z_{1:k-1}\right) \rho_i w_{k-1}^i \tag{46}$$

where $p\left(z_k \mid x_{0:k}^i, z_{1:k-1}\right)$ is computed using (23)-(24). For the resampling procedure, we perform resampling if the effective sampling size [14] is smaller than $N/3$. As a result, the forward filter of PLB-SLAM is a version of unscented FastSLAM, which is the basis for PLB-SLAM. Also, the forward filter of PLB-SLAM that uses AL instead of SLR is a version of FastSLAM where the EKF is used.

C. Results

In this section, we analyse the effect of backward simulation and the iterations of the IPLF on estimation performance. We evaluate the algorithms using Monte Carlo simulation with $N_{mc} = 300$ runs. In each run, we draw new measurements and beacon positions from the prior. We use 300 particles both in filtering and in backward simulation. In order to illustrate the benefits of backward simulation, we show the trajectory samples for forward filtering and backward simulation for an exemplar Monte Carlo run in Figure 2. In forward filtering, trajectories degenerate for past states so they do not represent the underlying uncertainty in the trajectories properly. Therefore, the beacon estimates, which depend on the trajectory samples, can be improved by using backward simulation, as trajectories do not degenerate for past states.

In the rest of the section, we analyse the errors in the estimation of the map and the trajectory so we proceed to explain how we compute them. Let $m^{j,l}$ and $\hat{m}^{j,l}$ denote the true and estimated position of the jth landmark in the lth

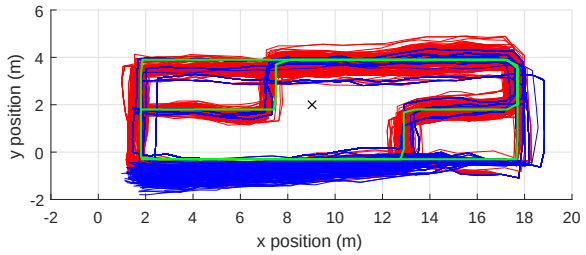

Figure 2: Three hundred trajectory samples from forward filtering (blue), backward simulation (red), and real trajectory (green), see Figure 1 as well. Trajectory samples obtained by filtering degenerate for the initial time steps. Trajectory samples obtained by backward simulation do not degenerate and represent the underlying uncertainty more accurately.

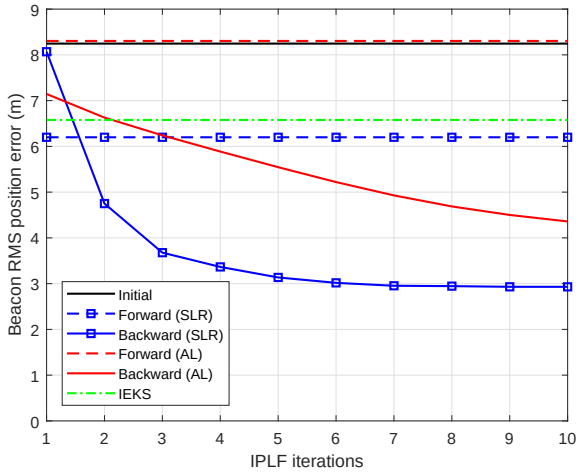

Figure 3: RMS position error of the beacons: Initial (prior error), forward FastSLAM filtering and posterior linearisation backward SLAM using SLR and AL. Iterations of the IPLF on the backward trajectories can substantially reduce the error, especially with SLR.

Monte Carlo run. The RMS error for the sensor position is defined as

$$E_m \triangleq \sqrt{\frac{1}{M N_{mc}} \sum_{l=1}^{N_{mc}} \sum_{j=1}^{M} \left(\hat{m}^{j,l} - m^{j,l}\right)^T \left(\hat{m}^{j,l} - m^{j,l}\right)}, \tag{47}$$

where we have normalised by M so that the error is defined per sensor. Let p_k denote the true position of the agent at time k and $\hat{p}_k^l$ its estimation at the lth Monte Carlo run. The RMS position error for the trajectory is defined as

$$E_p \triangleq \sqrt{\frac{1}{K N_{mc}} \sum_{l=1}^{N_{mc}} \sum_{k=1}^{K} \left(\hat{p}_k^l - p_k\right)^T \left(\hat{p}_k^l - p_k\right)}, \tag{48}$$

where we have normalised by K so that the error is defined with respect to one time step.

In Figure 3, we show the RMS position error of the beacons for the algorithms against the number of IPLF iterations. As expected, the IEKS and forward FastSLAM reduce the error compared to the initial error, which is given by the prior uncertainty. Nevertheless, backward simulation and the use of the IPLF can significantly lower the error with respect to the IEKS and forward FastSLAM solutions. Within the

first few iterations of the IPLF, the error is significantly lowered so it becomes clear the importance of the use of backward simulation and the IPLF for accurate beacon map estimation. Also, the use of SLR instead of AL to linearise the measurement model, which is expected to be beneficial as indicated in Section IV-B, has a major effect on performance.

We proceed to analyse the effect of varying the number of particles. We show the RMS error in trajectory and beacon estimates for the IEKS and 100, 200, 300 and 900 particles, both in filtering and backward simulation, in Table I. We first recall that the error in the trajectory estimates is independent of the IPLF iterations, as they are only performed to update the beacon positions for each backward trajectory sample. For all the particle numbers, backward simulation lowers the error of the trajectory estimate. In this scenario, the difference between forward and backward simulation for the trajectory estimates in error is slight, though backward trajectories represent the underlying uncertainty better, as can be seen in Figure 2. The most important improvement for beacon location is obtained by the application of the IPLF in the backward trajectories. The RMS error in beacon positions is reduced up to a 50% compared to the forward FastSLAM filter and this happens for all the considered number of particles. In addition, the use of SLR outperforms the use of AL in beacon position estimation, as expected. IEKS performs better than particle based methods to estimate the trajectory, but its error in map estimation, which is relevant in SLAM applications, is remarkably higher than the error of PLB-SLAM.

The running times of our Matlab implementations of the algorithms in a computer with a 3.5 GHz Intel Xeon E5 processor are given in Table II. IEKS has a remarkably low running time. However, as analysed before, its performance is rather low compared to the PLB-SLAM methods in terms of map estimation.

VI. Experimental results

In this section, the proposed method PLB-SLAM is evaluated in two sets of experimental data. First, in Section VI-A, we consider controlled experiments involving Bluetooth beacons and a pedestrian with a smartphone as the mobile agent. These experiments were designed so that we know the location of the beacon positions (ground truth). This enables us to compute the error in the estimation, as in the simulation results.

In the second set of experiments, which are explained in Section VI-B, we consider experiments involving Wi-Fi access points and a pedestrian with a smartphone in a transport hub in Helsinki, Finland. The main objective of these experiments is to show applications in real scenarios, for instance, an accurate mapping of the Wi-Fi access points in a transport hub can be used to aid indoor localisation and guide travelers. In this case, there is no ground truth available so we cannot compute the errors. Nevertheless, we rank the different algorithms based on their resulting log-likelihoods on a test set. Better position estimates are expected to explain the data of a test set better, which results in a higher log-likelihood.

Figure 4: Scenario of the experiments. Ten beacons are placed on tables in a canteen at Aalto University. To aid visualisation, beacons are marked by a red circle.

A. Controlled Bluetooth beacons experiments

In this section, we first discuss the setup of the experiments and then the results of the SLAM algorithms.

1) Setup: In order to evaluate the error in map estimation, which is our main objective, we need to know the true positions of the beacons so we designed the following controlled experiment. We placed 10 Bluetooth beacons (Kontakt.io smart beacons) on tables in a canteen, as shown in Figure 4, annotating their positions with the layout in Figure 5. The mobile agent is a pedestrian carrying a smartphone (Huawei Nexus 6) measuring the RSSI from the device's Bluetooth transceiver as well as the motion through the smartphone's inertial measurement unit (IMU; Bosch BMI160). The RSSI measurements are assumed to follow the exponential path loss model in (38). The measurement model parameters were calibrated for each beacon beforehand, and providing values quite similar to the ones in Section V-A. In our set-up, the Bluetooth beacons transmit their IDs asynchronously every 350 ms, so the smartphone can determine what RSSI measurements corresponds to each beacon and their timestamp. For simplicity, we synchronise the measurements using linear interpolation at every second.

The IMU measurements are used to provide the motion measurements for the proposed SLAM algorithm. The motion measurements are based on a pedestrian dead reckoning (PDR) algorithm that detects steps and estimates heading [38]–[40]. We used a step-detection algorithm based on the zero-crossings in the accelerometer signal [39], [41] and estimated the step length l_k using the Weinberg step length estimator [42]. The heading θ_k is estimated by first rotating the gyroscope measurements from the IMU coordinate frame into the person's coordinate frame, using the gravity reference vector from the accelerometer, and subsequent integration, similar to [39]. This procedure yields measurements of the change in position

$$y_k = l_k \left[\cos(\theta_k), \sin(\theta_k)\right]^T \tag{49}$$

where the function $h^y(\cdot)$ of the measurement model (2) is given by (39). Finally, the PDR as well as the RSSI

Table I: RMS error (m) in trajectory and beacon positions against the number of particles for forward filtering (F) and backward simulation with J IPLF iterations (BJ) using SLR and AL, and for the IEKS.

	Trajectory				Beacon							
	SLR		AL		SLR				AL			
	F	B	F	B	F	B1	B5	B10	F	B1	B5	B10
$N = 100$	0.86	0.85	0.86	0.85	6.17	8.06	3.10	3.00	8.40	7.13	5.58	4.45
$N = 200$	0.81	0.79	0.84	0.83	6.21	8.07	3.11	2.96	8.32	7.14	5.57	4.41
$N = 300$	0.78	0.77	0.88	0.87	6.20	8.07	3.14	2.93	8.30	7.14	5.55	4.36
$N = 900$	0.75	0.74	0.84	0.84	6.10	8.03	3.07	2.92	8.26	7.15	5.53	4.34
IEKS	0.62				6.58							

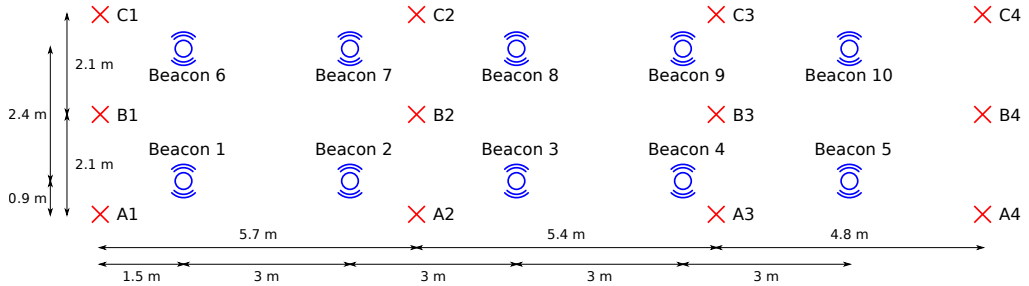

Figure 5: Illustration of the experimental layout with the Bluetooth beacons' arrangement (blue circles) and the anchor points (red crosses)

Table II: Running times in seconds of the algorithms in forward filtering (F) and backward simulation with J IPLF iterations (BJ)

	SLR				AL			
N	F	B1	B5	B10	F	B1	B5	B10
100	3.2	8.9	13.9	20.2	1.3	6.2	7.1	8.3
200	6.4	22.1	32.3	45.1	2.7	16.8	18.6	21.1
300	9.6	40.6	56.0	75.1	4.0	32.7	35.6	39.3
900	29.4	293.6	332.9	387.0	12.2	260.1	269.3	279.6
IEKS	0.22							

Table III: Trajectories in experiments

No.	Sequence of anchor points
1	B1–B2–C2–C4–B4–B3–A3–A1–C1–C4–A4–A1
2	C1–C4–B4–B1–A1–A4
3	C1–C2–A2–A3–C3–C4–A4–A3–C3–C2–A2–A1
4	B1–B2–C2–C4–B4–B2–A2–A1
5	C1–C4–B4–B1–A1–A4–B4–B1

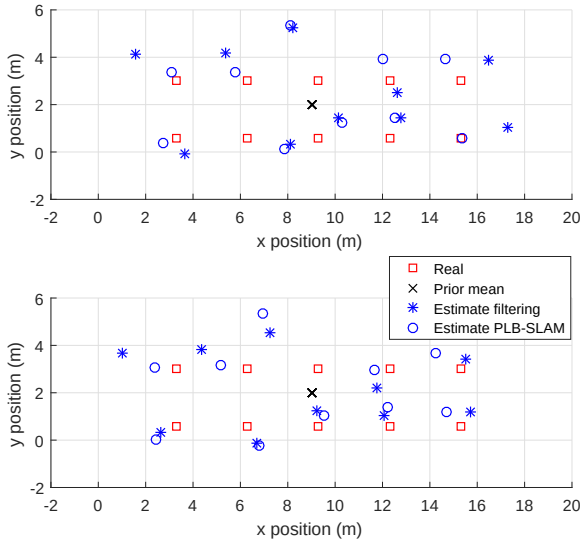

Figure 6: Estimates for beacon positions in Experiment 1 (top) using PDR based on the anchor positions (bottom) using smartphone PDR. The prior mean for all beacons is the same. Overall, both forward FastSLAM filtering and PLB-SLAM improve the beacon locations, and PLB-SLAM is more accurate in RMS sense, see Table IV.

data was linearly interpolated to obtain uniformly sampled measurements with a sampling time of 1 s.

Based on the setup in Figure 5, five different experiments were made. In each experiment, the anchor points shown in Figure 5 were used such that the walking path consisted of straight segments between two of these points. Reference trajectories were then calculated for each experiment by extracting the timestamps when the anchor points were passed and linking them with straight movements with constant velocity. The different trajectories are listed in Table III. An important benefit of having these accurate reference trajectories is that, apart from the PDR based on the smartphone IMU, we can also obtain a PDR based on reference trajectories and evaluate the algorithms independently of the step-detection system, which is not the topic of this paper. The PDR measurement y_k based on anchor points is obtained by using (39) with the corresponding reference trajectory.

2) Results: In the experiments, we use the same dynamic and measurement models and filter parameters for PLB-SLAM as in Section V, which consider 300 particles in filtering and smoothing. We first analyse Experiment 1, whose trajectory can be found in Table III. This trajectory, reconstructed using the anchor points and the simulated motion measurements explained in the previous subsection, is in fact the trajectory in Figure 1, which was used in the simulations in Section V.

We show the estimated beacon positions for forward FastSLAM filtering and PLB-SLAM (with SLR) with 5 iterations in the IPLF in Figure 6. For both types of motion measurements, FastSLAM and PLB-SLAM satisfactorily improve the locations of all the beacons, except for beacon 8, see Figure 5. Nevertheless, PLB-SLAM is able to provide more accurate results in RMS sense, as will be analysed in the following. For example, it provides remarkably more accurate estimates than FastSLAM for beacons 5, 6 and 7 with the anchor point PDR.

Table IV: Averaged RMS error (m) for beacon positions for forward filtering and backward simulation with 1 and 5 IPLF iterations

	Anchor point PDR							Smartphone PDR						
	SLR			AL			IEKS	SLR			AL			IEKS
Experiment	F	B1	B5	F	B1	B5		F	B1	B5	F	B1	B5	
1	1.60	8.41	1.22	1.96	2.34	1.09	1.14	1.42	7.23	1.36	1.85	2.37	1.26	0.80
2	1.45	8.04	0.69	1.16	3.17	0.83	0.91	1.41	7.71	1.11	2.45	2.70	2.09	1.78
3	1.97	7.07	1.29	3.96	3.71	3.12	2.07	2.08	4.98	1.88	5.07	3.42	3.21	6.01
4	1.62	6.42	0.99	2.25	3.45	1.60	1.22	1.29	6.44	0.88	2.83	3.13	1.05	1.91
5	1.96	10.17	1.05	0.86	3.78	1.01	0.91	2.17	10.80	1.28	1.26	2.86	1.07	1.08

We also show the averaged RMS error for estimated beacon positions by the algorithms, considering 5 iterations of the IPLF, in the experiments in in Table IV. Considering all experiments, we can see that the application of several iterations of the IPLF, using SLR, in combination with backward simulation usually provide the lowest errors. Importantly, just running backward simulation and applying the UKF (one IPLF iteration) increases the error with respect to the FastSLAM filtering solution. This highlights the importance of performing iterations in the IPLF in highly nonlinear settings. In general, the PDR based on anchor points provides lower errors than the PDR based on the smartphone IMU, as trajectory reconstruction is more accurate using anchor points. In addition, the improvement of PLB-SLAM with respect to forward filtering (FastSLAM) is higher with the anchor point PDR and can be quite significant. The highest improvement in RMS error with respect to forward filtering using SLR is 91 cm for Experiment 5 and anchor point PDF. In addition, with SLR, there is over a 50 cm reduction in averaged RMS error for the beacon positions in Experiment 2 (anchor point PDR), Experiment 3 (anchor point PDR), Experiment 4 (anchor point PDR), and Experiment 5 (both PDRs). For smartphone PDR and SLR, there is a reduction in the RMS error of PLB-SLAM with respect to FastSLAM higher than 20 cm for all experiments, except for Experiment 1. Though this difference is smaller than with anchor point PDR, it is still substantial. The IEKS performs very well in some scenarios, such as Experiment 1 and 5, but it performs considerably worse than PLB-SLAM in Experiment 2, 3 and 4. On average, PLB-SLAM is the best performing algorithm.

B. Experiments with Wi-Fi access points

In these experiments, we perform SLAM to map the locations of Wi-Fi access points on one floor in a transport hub in Helsinki, Finland. The data have been collected by the company IndoorAtlas. In this data, a pedestrian with an LG Nexus 5X smartphone measures RSSI from Wi-Fi access points. The data also contains accurate trajectory information obtained by a proprietary IndoorAtlas algorithm based on PDR and map matching. As in the previous experiments, we use this trajectory information to create the odometry measurements, as odometry is not the topic of this paper.

We consider four experiments, for which the agent trajectories are shown in Figure 7. The lengths of the four trajectories are approximately 120 m, 150 m, 175 m and 140 m. Among all the detected Wi-Fi access points, we use a maximum likelihood criterion to choose the access points that are in the horizontal plane of the trajectories, which are 16. Then, we use the first experiment to calibrate the measurement model parameters of these Wi-Fi access points by maximising the likelihood on the sensor positions and parameters P_0 and γ, and assuming that the agent trajectory is perfectly known.

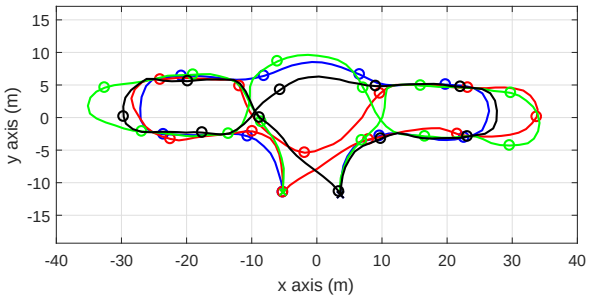

Figure 7: Four considered trajectories: Experiment 1 (blue), Experiment 2 (red), Experiment 3 (green) and Experiment 4 (black). Initial agent position is marked by a cross and every 10 time steps by circles.

In the SLAM algorithms, we consider the calibrated values of P_0 and γ for each sensor. Also, all sensors have the same prior PDF with mean $\overline{m}^j = [0, 0]^T$ (m) and covariance matrix $P^j = \text{diag}\left(\left[50^2, 20^2\right]\right)$ $\left(\text{m}^2\right)$. The rest of the parameters of the filters are as in the previous experiments and we consider 300 particles in forward filtering and backward simulation. The previous SLAM algorithms are run on the four experiments and we estimate the location of the Wi-Fi access points.

As we do not know the true locations of the Wi-Fi access points, we cannot compute the error with respect to the ground truth, as in the previous examples. Instead, we compute the log-likelihood of the resulting estimates for the fourth experiment, assuming that the agent trajectory is perfectly known and using the measurement model (38). The more accurate the positions of the Wi-Fi access points are estimated, the log-likelihood is expected to be higher as the estimated sensor positions explain the data better.

The resulting log-likelihoods for the estimated positions in each experiment are shown in Table V. The best performing algorithm is PLB-SLAM with SLR and the application of IPLF iterations. The worst performing algorithm in this data set is the IEKS.

Therefore, we can conclude that, on the whole, PLB-SLAM has important benefits compared to other SLAM algorithms in batch problems, as demonstrated by simulated data and two sets of experimental data with different sensors.

VII. Conclusions

In this paper, we have proposed the posterior linearisation backward SLAM algorithm as a batch solution to the

Table V: Log-likelihood (divided by a factor 10^3) of the estimated sensor positions for each experiment (Exp.)

	SLR			AL			IEKS
Exp.	F	B1	B5	F	B1	B5	
1	-1.623	-2.148	-1.592	-1.631	-1.872	-1.645	-1.704
2	-1.620	-2.116	-1.617	-1.653	-1.868	-1.648	-1.762
3	-1.632	-2.189	-1.618	-1.619	-1.869	-1.676	-1.766
4	-1.615	-2.210	-1.615	-1.660	-1.879	-1.655	-1.717

SLAM problem. PLB-SLAM is based on approximate Rao-Blackwellised particle smoothing and the posterior linearisation technique to obtain accurate approximation of the PDFs of the agent trajectory and the landmark map. The higher performance of PLB-SLAM compared to FastSLAM comes from two aspects: 1) PLB-SLAM uses backward simulation to obtain non-degenerate trajectory samples and 2) PLB-SLAM makes use of the iterated posterior linearisation filter rather than a sigma-point Kalman filter to obtain the conditional PDFs of the beacon position given the trajectory samples.

We have shown the benefits of PLB-SLAM in relation to FastSLAM and the IEKS for batch SLAM problems using simulations and experiments with smarthphones, Bluetooth beacons, and Wi-Fi access points.

APPENDIX A

In this appendix, we prove Proposition 2. We have that

$$\begin{aligned} &p\left(x_{0:k} \mid x_{k+1:K}, z_{1:K}, y_{1:K}\right) \\ &= \frac{p\left(x_{0:k}, x_{k+1:K}, z_{k+1:K}, y_{k+1:K} \mid z_{1:k}, y_{1:k}\right)}{p\left(x_{k+1:K}, z_{k+1:K}, y_{k+1:K} \mid z_{1:k}, y_{1:k}\right)} \\ &= \frac{p\left(x_{k+1:K}, z_{k+1:K}, y_{k+1:K} \mid z_{1:k}, y_{1:k}, x_{0:k}\right)}{p\left(x_{k+1:K}, z_{k+1:K}, y_{k+1:K} \mid z_{1:k}, y_{1:k}\right)} \\ &\quad \times p\left(x_{0:k} \mid z_{1:k}, y_{1:k}\right). \end{aligned} \tag{50}$$

Then,

$$\begin{aligned} &p\left(x_{k+1:K}, z_{k+1:K}, y_{k+1:K} \mid z_{1:k}, y_{1:k}, x_{0:k}\right) \\ &= \int p\left(x_{k+1:K}, z_{k+1:K}, y_{k+1:K}, m \mid z_{1:k}, y_{1:k}, x_{0:k}\right) dm \\ &= \int p\left(m \mid z_{1:k}, y_{1:k}, x_{0:k}\right) \\ &\quad \times \prod_{p=k+1}^{K} \left[p\left(x_p \mid x_{p-1}\right) p\left(y_p \mid x_p, x_{p-1}\right) p\left(z_p \mid x_p, m\right)\right] dm. \end{aligned} \tag{51}$$

Removing all the terms that are constants given $x_{k+1:K}, z_{1:K}, y_{1:K}$, the backward kernel is

$$\begin{aligned} &p\left(x_{0:k} \mid x_{k+1:K}, z_{1:K}, y_{1:K}\right) \\ &\quad \propto p\left(x_{0:k} \mid z_{1:k}, y_{1:k}\right) p\left(x_{k+1} \mid x_k\right) p\left(y_{k+1} \mid x_{k+1}, x_k\right) \\ &\quad \times \int p\left(m \mid z_{1:k}, y_{1:k}, x_{0:k}\right) \prod_{p=k+1}^{K} p\left(z_p \mid x_p, m\right) dm \\ &\quad = p\left(x_{0:k} \mid z_{1:k}, y_{1:k}\right) p\left(x_{k+1} \mid x_k\right) p\left(y_{k+1} \mid x_{k+1}, x_k\right) \\ &\quad \times \prod_{j=1}^{M} \int p\left(m^j \mid z_{1:k}, y_{1:k}, x_{0:k}\right) \prod_{p=k+1}^{K} p\left(z_p^j \mid x_p, m^j\right) dm^j. \end{aligned} \tag{52}$$

Using the Rao-Blackwellised particle filtering approximation, see (15), and considering we have drawn a sample $\tilde{x}_{k+1:K}$, see Proposition 2, we have

$$\begin{aligned} &p\left(x_{0:k} \mid \tilde{x}_{k+1:K}, z_{1:K}, y_{1:K}\right) \\ &\propto \sum_{i=1}^{N} w_k^i \delta\left(x_{0:k} - x_{0:k}^i\right) p\left(\widetilde{x}_{k+1} \mid x_k^i\right) p\left(y_{k+1} \mid \widetilde{x}_{k+1}, x_k^i\right) \\ &\quad \times \prod_{j=1}^{M} \int \mathcal{N}\left(m^j; \overline{m}_k^{i,j}, P_k^{i,j}\right) \prod_{p=k+1}^{K} p\left(z_p^j \mid \widetilde{x}_p, m^j\right) dm^j, \end{aligned} \tag{53}$$

which proves Proposition 2.

APPENDIX B

In this appendix, we prove Proposition 3. Given the linearised measurement model (29), using the information filter update equation [43, Sec. 6.3], we have that

$$\prod_{p=k+1}^{K} p\left(z_p^j \mid \widetilde{x}_p, m^j\right) \propto \mathcal{N}\left(m^j; \left(L_k^j\right)^{-1} l_k^j, \left(L_k^j\right)^{-1}\right) \tag{54}$$

where l_k^j and L_k^j are given as in Proposition 3. In the rest of this appendix, we drop superindex j on l_k^j and L_k^j for clarity. Substituting the previous equation into (28), we find

$$\xi_k^{i,j} = \mathcal{N}\left(L_k^{-1} l_k; \overline{m}_k^{i,j}, P_k^{i,j} + L_k^{-1}\right). \tag{55}$$

We can derive an alternative expresion that avoids inverting L_k making use of the following two identities. We first have that, by the matrix determinant lemma,

$$\left|P_k^{i,j} + L_k^{-1}\right| = \left|I + \left(P_k^{i,j}\right)^{T/2} L_k \left(P_k^{i,j}\right)^{1/2}\right| \left|L_k^{-1}\right|. \tag{56}$$

On the other hand, by the matrix inversion lemma, we have

$$\begin{aligned} &\left(P_k^{i,j} + L_k^{-1}\right)^{-1} = L_k - L_k P_k^{i,j} \\ &\quad \times \left(I + \left(P_k^{i,j}\right)^{T/2} L_k \left(P_k^{i,j}\right)^{1/2}\right)^{-1} \left(P_k^{i,j}\right)^{T/2} L_k. \end{aligned} \tag{57}$$

Then, plugging (56) and (57) into (55) and removing the factors that do not depend on i, we obtain (30).

REFERENCES

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," *IEEE Robotics Automation Magazine*, vol. 13, no. 2, pp. 99–110, June 2006.

[2] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, Oct 2015.

[3] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu, "StructSLAM: Visual SLAM with building structure lines," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1364–1375, April 2015.

[4] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: A survey of current trends in autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194–220, Sep. 2017.

[5] H. Qin, Z. Meng, W. Meng, X. Chen, H. Sun, F. Lin, and M. H. Ang, "Autonomous exploration and mapping system using heterogeneous UAVs and UGVs in GPS-denied environments," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1339–1350, Feb. 2019.

[6] V. Savic, H. Wymeersch, and E. G. Larsson, "Target tracking in confined environments with uncertain sensor positions," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 2, pp. 870–882, Feb. 2016.
[7] A. F. García-Fernández, M. R. Morelande, and J. Grajal, "Multitarget simultaneous localization and mapping of a sensor network," *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4544–4558, Oct. 2011.
[8] P. Mirowski, T. K. Ho, S. Yi, and M. MacDonald, "SignalSLAM: Simultaneous localization and mapping with mixed WiFi, Bluetooth, LTE and magnetic signals," in *International Conference on Indoor Positioning and Indoor Navigation*, Oct. 2013, pp. 1–10.
[9] W. W. Kao and B. Q. Huy, "Indoor navigation with smartphone-based visual SLAM and Bluetooth-connected wheel-robot," in *2013 CACS International Automatic Control Conference*, Dec. 2013, pp. 395–400.
[10] D. Hahnel, W. Burgard, D. Fox, K. Fishkin, and M. Philipose, "Mapping and localization with RFID technology," in *IEEE International Conference on Robotics and Automation*, vol. 1, April 2004, pp. 1015–1020.
[11] J.-F. Chen and C.-C. Wang, "Long-term RFID SLAM using short-range sparse tags," *International Journal of Automation and Smart Technology*, vol. 5, no. 1, pp. 61–75, 2015.
[12] B. Ferris, D. Fox, and N. Lawrence, "WiFi-SLAM using Gaussian process latent variable models," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007, pp. 2480–2485.
[13] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal, "Efficient, generalized indoor WiFi GraphSLAM," in *IEEE International Conference on Robotics and Automation*, May 2011, pp. 1038–1043.
[14] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013.
[15] R. Martínez-Cantín and J. A. Castellanos, "Unscented SLAM for large-scale outdoor environments," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Aug. 2005, pp. 3427–3432.
[16] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2002, pp. 593–598.
[17] ——, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proceedings of the International Conference on Artificial Intelligence (IJCAI)*, 2003, pp. 1151–1156.
[18] 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, Feb. 2002.
[19] C. Kim, R. Sakthivel, and W. K. Chung, "Unscented FastSLAM: A robust and efficient solution to the SLAM problem," *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 808–820, Aug. 2008.
[20] A. F. García-Fernández, L. Svensson, M. R. Morelande, and S. Särkkä, "Posterior linearization filter: principles and implementation using sigma points," *IEEE Transactions on Signal Processing*, vol. 63, no. 20, pp. 5561–5573, Oct. 2015.
[21] F. Lindsten and T. Schön, "Backward simulation methods for Monte Carlo statistical inference," *Foundations and Trends in Machine Learning*, vol. 6, no. 1, pp. 1–143, 2013.
[22] F. Lindsten, P. Bunch, S. Särkkä, T. B. Schön, and S. J. Godsill, "Rao-Blackwellized particle smoothers for conditionally linear Gaussian models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 2, pp. 353–365, March 2016.
[23] S. Särkkä, P. Bunch, and S. Godsill, "A backward-simulation based Rao-Blackwellized particle smoother for conditionally linear Gaussian models," in *16th IFAC Symposium on System Identification*, 2012, pp. 506–511.
[24] K. Berntorp and J. Nordh, "Rao-Blackwellized particle smoothing for occupancy-grid based SLAM using low-cost sensors," in *Proceedings of the 19th IFAC World congress*, 2014.
[25] A. F. García-Fernández, L. Svensson, and S. Särkkä, "Iterated posterior linearization smoother," *IEEE Transactions on Automatic Control*, vol. 62, pp. 2056–2063, April 2017.
[26] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, Mar. 1996.
[27] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004.
[28] Y. Song, Q. Li, Y. Kang, and Y. Song, "CFastSLAM: a new Jacobian free solution to SLAM problem," in *IEEE International Conference on Robotics and Automation*, 2012.
[29] R. Havangi, H. D. Taghirad, M. A. Nekoui, and M. Teshnehlab, "A square root unscented FastSLAM with improved proposal distribution and resampling," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 5, pp. 2334–2345, May 2014.
[30] M. R. Morelande and A. F. García-Fernández, "Analysis of Kalman filter approximations for nonlinear measurements," *IEEE Transactions on Signal Processing*, vol. 61, no. 22, pp. 5477–5484, Nov. 2013.
[31] I. Arasaratnam, S. Haykin, and R. Elliott, "Discrete-time nonlinear filtering algorithms using Gauss-Hermite quadrature," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 953–977, May 2007.
[32] A. Rai, K. Chintalapudi, V. Padmanabhan, and R. Sen, "Zee: Zero-effort crowdsourcing for indoor localization," in *Mobicom*, August 2012.
[33] J. Jessup, S. N. Givigi, and A. Beaulieu, "Robust and efficient multi-robot 3-D mapping merging with Octree-based occupancy grids," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1723–1732, Sep. 2017.
[34] B. M. Bell, "The iterated Kalman smoother as a Gauss-Newton method," *SIAM Journal on Optimization*, vol. 4, no. 3, pp. 626–636, Aug. 1994.
[35] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation*, May 2011, pp. 3607–3613.
[36] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *The International Journal of Robotics Research*, vol. 25, pp. 1181–1203, December 2006.
[37] D. Dardari, P. Closas, and P. M. Djuric, "Indoor tracking: Theory, methods, and technologies," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1263–1278, April 2015.
[38] R. Harle, "A survey of indoor inertial positioning systems for pedestrians," *IEEE Communications Surveys Tutorials*, vol. 15, no. 3, pp. 1281–1293, 2013.
[39] Z. Xiao, H. Wen, A. Markham, and N. Trigoni, "Robust pedestrian dead reckoning (R-PDR) for arbitrary mobile device placement," in *International Conference on Indoor Positioning and Indoor Navigation*, Oct. 2014, pp. 187–196.
[40] R. Hostettler and S. Särkkä, "IMU and magnetometer modeling for smartphone-based PDR," in *International Conference on Indoor Positioning and Indoor Navigation*, Oct 2016, pp. 1–8.
[41] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," in *3rd Workshop on Positioning, Navigation and Communication*, 2006, pp. 27–35.
[42] H. Weinberg, "Using the ADXL202 in pedometer and personal navigation applications," Analog Devices, Tech. Rep., 2002.
[43] B. O. Anderson and J. B. Moore, *Optimal Filtering*. Prentice-Hall, 1979.

Ángel F. García-Fernández received the telecommunication engineering degree (with honours) and the Ph.D. degree from Universidad Politécnica de Madrid, Madrid, Spain, in 2007 and 2011, respectively.

He is currently a Lecturer in the Department of Electrical Engineering and Electronics at the University of Liverpool, Liverpool, UK. He is also an external research associate in the ARIES Research Center, Universidad Antonio de Nebrija, Madrid, Spain. He previously held postdoctoral positions at Universidad Politécnica de Madrid, Chalmers University of Technology, Gothenburg, Sweden, Curtin University, Perth, Australia, and Aalto University, Espoo, Finland. His main research activities and interests are in the area of Bayesian estimation, with emphasis on dynamic systems. He was recipient of the best paper award at the International Conference on Information Fusion in 2017.

Roland Hostettler (S'10-M'14) received the Dipl. Ing. degree in Electrical and Communication Engineering from Bern University of Applied Sciences, Switzerland in 2007, and the M.Sc. degree in Electrical Engineering and Ph.D. degree in Automatic Control from Luleå University of Technology, Sweden in 2009 and 2014, respectively. He has held Post-Doctoral Researcher positions at Luleå University of Technology, Sweden and Aalto University, Finland. Currently, he is a Research Fellow with the Department of Electrical Engineering and Automation, Aalto University, Finland. His research interests include statistical signal processing with applications to target tracking, biomedical engineering, and sensor networks.

Simo Särkkä received his Master of Science (Tech.) degree (with distinction) in engineering physics and mathematics, and Doctor of Science (Tech.) degree (with distinction) in electrical and communications engineering from Helsinki University of Technology, Espoo, Finland, in 2000 and 2006, respectively. From 2000 to 2010 he worked with Nokia Ltd., Indagon Ltd., and Nalco Company in various industrial research projects related to telecommunications, positioning systems, and industrial process control. From 2010 to 2013 he worked as a Senior Researcher with the Department of Biomedical Engineering and Computational Science (BECS) at Aalto University, Finland. Currently, Dr. Särkkä is an Associate Professor and Academy Research Fellow with Aalto University, Technical Advisor of IndoorAtlas Ltd., and an Adjunct Professor with Tampere University of Technology and Lappeenranta University of Technology. In 2013 he was a Visiting Professor with the Department of Statistics of Oxford University and in 2011 he was a Visiting Scholar with the Department of Engineering at the University of Cambridge, UK. His research interests are in multi-sensor data processing systems with applications in location sensing, health and medical technology, machine learning, inverse problems, and brain imaging. He has authored or coauthored 100 peer-reviewed scientific articles and his book "Bayesian Filtering and Smoothing" along with its Chinese translation were recently published via the Cambridge University Press. He is a Senior Member of IEEE and serving as an Associate Editor of IEEE Signal Processing Letters.