

RESEARCH ARTICLE  

Rao-Blackwellized particle smoothing for simultaneous localization and mapping

Manon Kok¹ , Arno Solin²  and Thomas B. Schön³ 

¹Delft Center for Systems and Control, Delft University of Technology, Delft, the Netherlands

²Department of Computer Science, Aalto University, Aalto, Finland

³Department of Information Technology, Uppsala University, Uppsala, Sweden

Corresponding author: Manon Kok; Email: m.kok-1@tudelft.nl

Received: 07 June 2023; **Revised:** 09 April 2024; **Accepted:** 12 April 2024

Keywords: Gaussian processes; magnetic field localization; particle smoothing; sequential Monte Carlo; simultaneous localization and mapping

Abstract

Simultaneous localization and mapping (SLAM) is the task of building a map representation of an unknown environment while at the same time using it for positioning. A probabilistic interpretation of the SLAM task allows for incorporating prior knowledge and for operation under uncertainty. Contrary to the common practice of computing point estimates of the system states, we capture the full posterior density through approximate Bayesian inference. This dynamic learning task falls under state estimation, where the state-of-the-art is in sequential Monte Carlo methods that tackle the forward filtering problem. In this paper, we introduce a framework for probabilistic SLAM using particle smoothing that does not only incorporate observed data in current state estimates, but it also backtracks the updated knowledge to correct for past drift and ambiguities in both the map and in the states. Our solution can efficiently handle both dense and sparse map representations by Rao-Blackwellization of conditionally linear and conditionally linearized models. We show through simulations and real-world experiments how the principles apply to radio (Bluetooth low-energy/Wi-Fi), magnetic field, and visual SLAM. The proposed solution is general, efficient, and works well under confounding noise.

Impact statement

Simultaneous localization and mapping (SLAM) methods constitute an essential part of the backbone for autonomous robots, vehicles, and aircraft that operate in an environment with only on-board sensing, without external sensor support. The methods proposed in this paper help characterize the uncertainty of the pose and the map representation, and contribute to making SLAM methods capable of handling unexpected real-world conditions such as changing weather and lighting, signal attenuation, and confounding artifacts in sensor data.

1. Introduction

In ego-motion estimation, simultaneous localization and mapping (SLAM) is a ubiquitous approach of simultaneously estimating the time-varying pose of a robot, person, or object and a map of the

  This research article was awarded Open Data and Open Materials badges for transparent practices. See the Data Availability Statement for details.

© The Author(s), 2024. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (<http://creativecommons.org/licenses/by/4.0>), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.



environment (Bailey and Durrant-Whyte, 2006; Durrant-Whyte and Bailey, 2006; Cadena et al., 2016; Stachniss et al., 2016). SLAM is widely used for autonomous robots, vehicles, and aircraft. The wide adoption is due to the general nature of the concept behind SLAM, which makes it applicable across different sensor modalities and use case scenarios. Still, different setups for SLAM typically motivate different representations of the map data structure, thereby constraining how inference and learning is done under the SLAM paradigm.

In **visual SLAM**, the map is traditionally represented by a set of sparse landmark points which are observed as projections by a camera rigidly attached to the moving coordinate frame (e.g., Hartley and Zisserman, 2004; Davison et al., 2007). Alternatively, dense representations are used, which can take the form of continuous surfaces (e.g., Kerl et al., 2013; Whelan et al., 2015; Bloesch et al., 2018). These two extremes motivate the alternative views we bring into the map representation also in other sensor modalities. In **magnetic SLAM** (e.g., Kok and Solin, 2018), the map is inherently dense as it characterizes the anomalies of the Earth's magnetic vector field which are observed by a three-axis magnetometer. In **radio-based SLAM** (e.g., Ferris et al., 2007), the map can either represent point-wise radio emitter sources (typically Wi-Fi base stations or Bluetooth low-energy (BLE) beacons) or a dense anomaly map of receiver signal strength indicator (RSSI) values. To this end, dense maps of these anomaly fields have been constructed using Gaussian process (GP) regression (Ferris et al., 2007). In this work we consider both sparse and dense SLAM, as illustrated in Figure 1, with the intent of presenting general principles for SLAM that extend across sensor modalities. Our interest lies in settings where the map is static (not changing over time), but initially unknown.

The SLAM problem is inherently nonlinear. To perform inference and learning under the SLAM paradigm, all approaches resort to approximate inference of some kind. Typically, methods either approximate the estimation problem using linearization and Gaussian approximations or using Monte Carlo sampling methods. More specifically, when considering online recursive algorithms, the former approximation results in extended Kalman filter (EKF)-SLAM (Bailey et al., 2006; Mourikis and Roumeliotis, 2007; Barrau and Bonnabel, 2015), while the latter results in particle filter-based SLAM. The most well-known particle filter SLAM algorithm is FastSLAM (Montemerlo et al., 2002). An extension of this algorithm is FastSLAM 2.0 (Montemerlo et al., 2003), which recognizes that the bootstrap particle filter implementation from Montemerlo et al. (2002) suffers from particle degeneracy, and hence it does not capture the full posterior distribution. In this work, we will instead overcome this limitation using particle smoothing.

In recent years, it has become more common to consider smoothing problems for SLAM, where all information acquired up to the current time is backtracked through the model to ensure better global

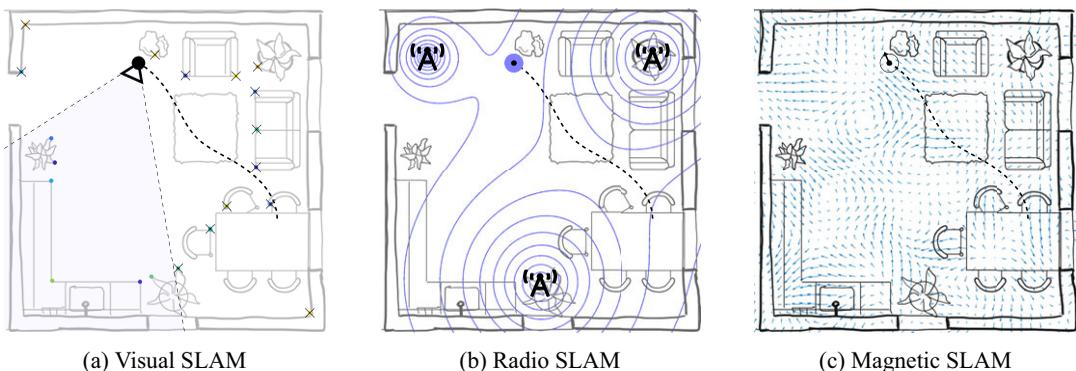


Figure 1. Illustration of the different SLAM modalities used throughout this paper. (a) Visual SLAM uses a sparse map representation of distinctive corner points observed as projections onto the camera frustum, (b) radio SLAM either uses a sparse map of the radio emitter locations or models the RSSI anomalies as a dense field, and (c) magnetic SLAM leverages anomalies in the local magnetic vector field.

consistency of the map and the past movement. This is closely related to graphSLAM (Thrun and Montemerlo, 2006; Grisetti et al., 2010) which turns the problem into a global (sparse) optimization task. Even though, these smoothing approaches allow for updated knowledge to correct for past drift and ambiguity in the map and path (bundle-adjustment), they are currently almost exclusively linearization-based as opposed to sampling-based. This means that rather than characterizing the full posterior, only point estimates are considered. In this work, we present a new approach for particle smoothing, that is, smoothing using sequential Monte Carlo (SMC) sampling. We show that it outperforms both particle filtering and extended Kalman filtering approaches. It is specifically more robust to unexpected real-world conditions, for instance, the case that the initial map estimates are quite off or that the sensors are not properly calibrated.

We develop a solution with similarities to FastSLAM and speed up the computations of our particle smoother by exploiting the conditionally linear substructure in the problem using Rao-Blackwellization (Doucet et al., 2000; Gustafsson et al., 2002; Schön et al., 2005). To this end, we leverage progress in the field of Rao-Blackwellized particle smoothing (Svensson et al., 2014, 2015). For these existing algorithms, however, it is not possible to assume a constant map that prevents using these algorithms for standard SLAM problems. To the best of our knowledge, the only existing work on particle smoothing for SLAM is Berntorp and Nordh (2014), which assumes that the map is slowly time-varying. This can, however, deteriorate the map quality, and have a negative effect on the estimation accuracy, especially in challenging applications. Our algorithm overcomes these limitations by explicitly assuming a constant map. An alternative interpretation of our SLAM problem is therefore in terms of joint state and parameter estimation, see Wigren et al. (2022) for a recent tutorial on the parameter estimation problem. This problem has traditionally been handled using maximum likelihood estimation (Schön et al., 2011), and a bit more than a decade ago, an interesting Bayesian solution was derived by Andrieu et al. (2010). We will derive a fully Bayesian solution to the SLAM problem, which allows for prior information to be included also on the map. This becomes particularly important when building GP maps. Our algorithm can be seen as a special case of Wigren et al. (2019), where for the specific form of the SLAM problem turns out to take a particularly nice form.

The contributions of this paper can be summarized as follows: (i) we present a general framework for SLAM which is agnostic to the map representation, (ii) we derive a Rao-Blackwellized particle smoothing method for SLAM, which exploits the conditionally linear or conditionally linearized substructure and circumvents typical pitfalls in particle filtering approaches, and (iii) we demonstrate the applicability of the proposed SLAM method in a range of simulated and real-world examples. A reference implementation of our method and code to reproduce the experiments can be found on <https://github.com/manonkok/Rao-Blackwellized-SLAM-smoothing>.

2. Background

In this work, we introduce a framework for probabilistic SLAM using particle smoothing. As our approach extends existing work on SMC methods—specifically particle filters—for localization and SLAM, we will introduce this concept in Section 2.1. We consider both sparse (feature-based) as well as dense maps. In terms of dense maps, we specifically focus on maps represented using GPs. Because of this, we briefly introduce the concept of GPs and their use to construct maps of the environment in Section 2.2.

2.1 SMC for localization and SLAM

In SMC, a posterior distribution is approximated using samples, also referred to as particles. Assuming for simplicity that we are interested in estimating the posterior $p(x_{1:t} | y_{1:t})$ of a set of time-varying states $x_{1:t} = (x_1, x_2, \dots, x_t)$ given a set of measurements $y_{1:t}$, the posterior is approximated as

$$\hat{p}(x_{1:t} | y_{1:t}) = \sum_{i=1}^N w_i^i \delta_{x_{1:t}}^i(x_{1:t}), \quad (1)$$

where N denotes the number of particles, w_t^i denotes the importance weight of each particle $x_{1:t}^i$ at time t and $\delta_{x_{1:t}^i}$ denotes the Dirac measure at $x_{1:t}^i$. Using SMC to approximate the posterior distribution $p(x_{1:t}|y_{1:t})$ is also synonymously called particle filtering. The idea was proposed in the beginning of the 1990s (Stewart and McCarty, 1992; Gordon et al., 1993; Kitagawa, 1993) and there are by now several tutorials (Doucet and Johansen, 2011; Naesseth et al., 2019) and textbooks (Särkkä, 2013; Chopin and Papaspiliopoulos, 2020) available on the subject. The use of a particle filter for localization in a known magnetic field map is illustrated in Figure 2 (estimated magnetic anomaly map to the left, localization result to the right). Here, a map of the strongly position-dependent indoor magnetic field is used to give information about how likely each particle is. The approach illustrated in the figure follows Solin et al. (2016), where the idea of *map matching* is cast as positioning in a vector-valued magnetic field map. As can be seen, the posterior is highly multimodal to start with, but becomes unimodal when time passes and a unique trajectory through the magnetic map can be reconstructed. The fact that SMC can handle multimodal distributions makes it especially well-suited for both localization and SLAM problems.

The challenge in SMC is to ensure that the particles represent the full posterior distribution. Because of this, the particles are regularly resampled, discarding the particles with low weight and replicating the particles with high weights. This, however, results in the phenomenon of particle degeneracy, where all particles at a certain time t descend from the same ancestor some time $t - s$ in the past. To overcome this issue, a number of particle smoothing algorithms have been developed—see Lindsten and Schön (2013) for a tutorial.

Using SMC for higher-dimensional state spaces typically requires a larger number of particles and hence increases the computational complexity. Because of this, a number of algorithms have been developed to exploit a conditionally linear substructure in the models and treat this using a Kalman filter (Chen and Liu, 2000; Andrieu and Doucet, 2002; Schön et al., 2005). This has resulted in the so-called marginalized or Rao-Blackwellized particle filters (RBPFs). The current work is inspired by these approaches and develops a Rao-Blackwellized particle smoothing algorithm for SLAM.

2.2. Representing maps using GP priors

GPs (Rasmussen and Williams, 2006) represent distributions over functions. A GP $h(x)$ is fully specified by its mean function $\mu(x)$ and covariance function $\kappa(x, x')$ as

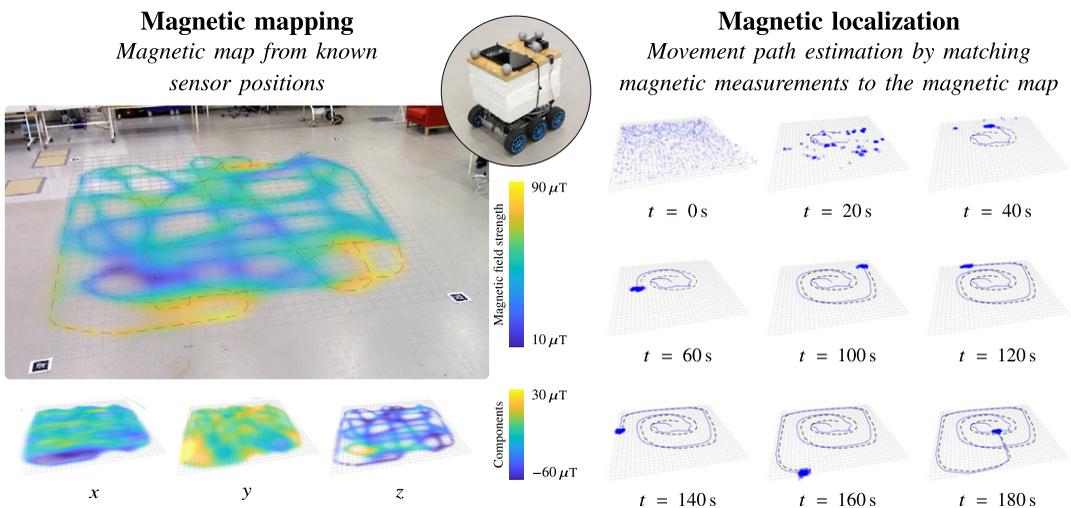


Figure 2. Left: A magnetic anomaly map mapped by a robot (smartphone for scale) equipped with a three-axis magnetometer. Map opacity follows the marginal variance (uncertainty), and mapping (training) paths shown by dashed lines. Right: Localization by map matching. Current estimate is characterized by a particle cloud, the dashed line shows the ground-truth, and the solid line the weighted mean path.

$$h(x) \sim \mathcal{GP}(\mu(x), \kappa(x, x')). \tag{2}$$

The covariance function (kernel) can encode prior information about the properties of the function, such as continuity, smoothness, or physical properties of a random field. Observational data $\mathcal{D} = \{(x_t, y_t)\}_{t=1}^T$ is coupled with the GP prior through a likelihood model. The Gaussian likelihood model reduces to observing noise-corrupted versions of the GP

$$y_t = h(x_t) + \varepsilon_t, \tag{3}$$

where $y_t \in \mathbb{R}^{n_y}$, and $\varepsilon_t \sim \mathcal{N}(0, \sigma^2 \mathcal{I}_{n_y})$ denotes zero-mean Gaussian independent and identically distributed (i.i.d.) measurement noise with covariance $\sigma^2 \mathcal{I}_{n_y}$, with \mathcal{I}_{n_y} being an identity matrix of size n_y . Under a Gaussian (conjugate) likelihood, GP regression can be conducted in closed form.

In recent years, GPs have been used to construct dense maps of the magnetic field or of the Wi-Fi/BLE RSSI field to be used for localization (Ferris et al., 2007; Vallivaara et al., 2010; Kok and Solin, 2018; Coulin et al., 2021; Viset et al., 2022). In these approaches, the field is modeled as a GP and the input to the GP, x , is in this case the position. The GP map is used to make predictions of the field at previously unseen locations. Since the GP map also provides information about the uncertainty of these predictions, such approaches are suitable to handle situations of uninformative maps at certain locations. In Figure 2, we visualize the magnitude of the magnetic field predicted using a GP that has learned the magnetic field based on measurements along the visualized path. The transparency visualizes the marginal variance, where larger transparency indicates a more uncertain prediction.

One of the main challenges with GPs is their computational complexity, which scales cubically with the number of data points. Because of this, there is a large literature dedicated to the construction and use of computationally efficient GP regression (see, e.g., Quiñero-Candela and Rasmussen, 2005; Hensman et al., 2013). One method that is particularly relevant for our work is the reduced-rank GP regression approach proposed by Solin and Särkkä (2020), which rewrites the GP model in terms of a Hilbert space representation. This approach approximates the covariance function in terms of an eigenfunction expansion of the Laplace operator in a confined domain as follows:

$$\kappa(x, x') \approx \sum_{j=1}^m S(\sqrt{\lambda_j}) \phi_j(x) \phi_j(x') = \Phi \Lambda \Phi^T, \tag{4}$$

where $\phi_j(x)$ denotes the orthonormal eigenfunctions and λ_j denotes the corresponding eigenvalues. Furthermore, $S(\cdot)$ is the spectral density function of the kernel. For rectangular domains, the expressions for the eigenfunctions and eigenvalues can be computed in closed form (Solin and Särkkä, 2020). For more general domains, they can be approximated numerically (see, e.g., Solin and Kok, 2019). The spectral density of the kernel can be computed in closed form for any stationary kernel (Rasmussen and Williams, 2006; Solin and Särkkä, 2020). The approximation in Eq. (4) has been shown to converge to the exact GP solution when the number of eigenfunctions and the size of the domain tend toward infinity. However, it is typically a good approximation for a relatively small number of eigenfunctions as long as we do not come too close to the boundary.

Using the approximation in Eq. (4), allows us to write the measurement model Eq. (3) as

$$y_t \approx \Phi(x_t) \theta + \varepsilon_t, \tag{5}$$

with $\Phi(x_t) = (\phi_1(x_t) \ \phi_2(x_t) \ \dots \ \phi_m(x_t))^T$, and to write the GP prior in terms of a mean θ_0 and covariance

$$P_0 = \text{diag}(S(\lambda_1) \ S(\lambda_2) \ \dots \ S(\lambda_m)). \tag{6}$$

The posterior can be computed recursively as new data arrives as (Särkkä, 2013)

$$\hat{\theta}_t = \hat{\theta}_{t-1} + K_t(y_t - \hat{y}_t), \quad \hat{y}_t = \Phi(x_t) \hat{\theta}_{t-1}, \tag{7a}$$

$$P_t = P_{t-1} - K_t S_t K_t^T, \quad K_t = P_{t-1} (\Phi(x_t))^T S_t^{-1}, \quad S_t = \Phi(x_t) P_{t-1} (\Phi(x_t))^T + \sigma^2 \mathcal{I}. \quad (7b)$$

In Eq. (7), we can recognize that the recursive map updating is now posed as a recursive linear parameter estimation problem. We will use this representation for our SLAM approach with dense maps.

3. Models

We are interested in jointly estimating time-varying states $x_{1:T}$ at times $t = 1, 2, \dots, T$ and a constant map, which we denote in terms of constant parameters θ . The sensor pose, x_t at least consists of the sensor's position p_t^n and its orientation q_t^{nb} . The superscript n in p_t^n specifically indicates that we represent the position in a fixed navigation frame n . We focus both on planar localization where $p_t^n \in \mathbb{R}^2$ and on full 3D localization, that is, $p_t^n \in \mathbb{R}^3$. The double superscript on q_t^{nb} specifically indicates that we consider a rotation from a body-fixed frame b to the navigation frame n . The origin of this body-fixed frame b coincides with that of the sensor triads and the axes of the body-fixed frame are aligned with the sensor axes. We represent the orientation q_t^{nb} using unit quaternions when considering full 3D localization while we represent it using a single heading angle when considering planar localization. Additional states can be included in x_t such as the sensor's velocity or sensor biases.

We model the dynamics of the state as

$$x_{t+1} = f(x_t, u_t, w_t), \quad (8)$$

where w_t denotes the process noise and u_t denotes a possible input to the dynamic model which can be used to incorporate available odometry. Note the generality of the model Eq. (8) which can contain both nonlinearities as well as non-Gaussian noise. Hence, all dynamic models that are commonly used in SLAM can be written in the form of Eq. (8).

We additionally assume that we have prior information on the map θ as

$$p(\theta) = \mathcal{N}(\theta; \mu_\theta, P_\theta). \quad (9)$$

This prior information is particularly crucial when doing SLAM with GP-based maps, since the prior on the map is equal to the GP prior. Other examples are prior information available from previous data collections.

We furthermore assume that if the states $x_{1:T}$ would be known, inferring θ would be a linear or an almost linear estimation problem. In other words, we assume that there is a conditionally linear substructure or a conditionally approximately linear substructure in the measurement model

$$y_t = C(x_t)\theta + \varepsilon_t, \quad \text{or} \quad y_t \approx C(x_t)\theta + \varepsilon_t, \quad (10)$$

and that the measurement noise ε_t is Gaussian i.i.d. white noise with $\varepsilon_t \sim \mathcal{N}(0, \Sigma)$. The model Eq. (10) is fairly general and diverse measurement models typically considered in SLAM scenarios can be written in this form. In Section 4, we will present our smoothing algorithm for SLAM that will assume the model structures in Eqs. (8)–(10). To highlight the generality of the method, or, in other words, to highlight the generality of Eq. (10) in the context of SLAM, in this work, we will focus on three types of models: visual, radio, and magnetic SLAM. We will introduce specific measurement models for these three cases in Sections 3.1–3.3. In Section 3.1, we will first introduce a measurement model for magnetic field SLAM that we have used in our previous work (Kok and Solin, 2018). In Section 3.2, we will then show that for dense radio SLAM a similar model structure can be obtained. Both the measurement models in Sections 3.1 and 3.2 are conditionally linear. In Section 3.3, we will subsequently introduce a widely used model for visual SLAM that has a conditionally approximately linear substructure.

3.1. Dense magnetic SLAM

In magnetic SLAM, at each time instance t , we receive a three-dimensional measurement $y_{m,t}$ of the magnetic field. These three dimensions are not independent since the ambient magnetic field follows laws

of physics captured by Maxwell’s equations. One way to take this into account is to model the magnetic field measurements as (Kok and Solin, 2018)

$$\varphi(p) \sim \mathcal{GP}(0, \kappa_{\text{lin.}}(p, p') + \kappa_{\text{SE}}(p, p')), \tag{11a}$$

$$y_{m,t} = R(q_t^{\text{bn}}) \nabla \varphi(p) \Big|_{p=p_t^n} + \varepsilon_{m,t}, \tag{11b}$$

where $\varepsilon_{m,t} \sim \mathcal{N}(0, \sigma_m^2 \mathcal{I}_3)$ and

$$\kappa_{\text{lin.}}(p, p') = \sigma_{\text{lin.}}^2 p^T p', \quad \kappa_{\text{SE}}(p, p') = \sigma_r^2 \exp\left(-\frac{\|p - p'\|_2^2}{2\ell^2}\right). \tag{12}$$

In Eq. (11a), $\varphi(\cdot)$ denotes a scalar potential field, the GP depends on the position p and $\kappa(p, p')$ represents the covariance function (Rasmussen and Williams, 2006). The linear term $\kappa_{\text{lin.}}(p, p')$ in the covariance function models the local Earth’s magnetic field while the squared exponential term $\kappa_{\text{SE}}(p, p')$ models the magnetic field anomalies. The GP in Eq. (11a) has three hyperparameters: the lengthscale ℓ , the magnitude of the squared exponential term σ_r , and the magnitude of the linear term $\sigma_{\text{lin.}}$. The measurements $y_{m,t}$ of the magnetic field in Eq. (11b) are the gradient of the scalar potential field, expressed in the body frame b . We use the notation $R(q_t^{\text{nb}})$ to denote the rotation matrix representation of the state q_t^{nb} and $R(q_t^{\text{nb}}) = (R(q_t^{\text{bn}}))^T$.

Making use of the reduced-rank GP regression introduced in Section 2.2, via Eq. (5), we can write Eq. (11b) as

$$y_{m,t} = R(q_t^{\text{bn}}) \nabla \Phi(p_t^n) \theta + \varepsilon_{m,t}, \tag{13}$$

where $\Phi(p_t^n) \in \mathbb{R}^{m+3}$ consists of $m + 3$ basis functions evaluated at p_t^n , and θ represents the weight of each of these basis functions. The first three basis functions model the linear kernel, while the other m basis functions approximate the squared exponential kernel. The prior Eq. (9) on the weights θ is in this case given by $\mu_\theta = 0_{m+3}$, with 0_{m+3} being an m -dimensional zero-vector, and P_θ given by

$$P_\theta = \text{diag}(\sigma_{\text{lin.}}^2, \sigma_{\text{lin.}}^2, \sigma_{\text{lin.}}^2, S_{\text{SE}}(\lambda_1), S_{\text{SE}}(\lambda_2), \dots, S_{\text{SE}}(\lambda_m)). \tag{14}$$

For more background concerning the model, we refer to Kok and Solin (2018). Note the conditionally linear structure is in line with the more general measurement model Eq. (10).

3.2. Dense radio SLAM

In dense radio SLAM, a map of the RSSI values is constructed. Representing this map using a GP, the measurements can be modeled as (Ferris et al., 2007)

$$h(p) \sim \mathcal{GP}(0, \kappa(p, p')), \tag{15a}$$

$$y_{r,t} = h(p) \Big|_{p=p_t^n} + \varepsilon_{r,t}, \tag{15b}$$

where $\varepsilon_{r,t} \sim \mathcal{N}(0, \sigma_r^2)$. Note that $y_{r,t} \in \mathbb{R}$ although in practice it is common to receive RSSI measurements from multiple Wi-Fi access points, resulting in multiple maps of the form of Eq. (15).

Similarly to Section 3.1, using Eq. (5), we can write Eq. (15b) as

$$y_{r,t} = \Phi(p_t^n) \theta + \varepsilon_{r,t}, \tag{16}$$

where $\Phi(p_t^n) \in \mathbb{R}^m$ consists of m basis functions evaluated at p_t^n , and θ represents the weight of each of these basis functions. The prior Eq. (9) on the weights θ is in this case given by $\mu_\theta = 0_m$, and P_θ given by Eq. (6). Note that Eq. (16) is again conditionally linear as in the more general measurement model Eq. (10).

3.3. Sparse visual SLAM

In sparse visual SLAM, the map is represented in terms of a number of landmark locations which we denote by $p_{1:L}$. Using a pinhole camera model (Hartley and Zisserman, 2004), a measurement $y_{j,t}$, $j = 1, 2, \dots, L$ (tracked feature points in images) of landmark p_j can be modeled as

$$y_{j,t} = \frac{1}{\rho} \begin{pmatrix} l_{u,t} \\ l_{v,t} \end{pmatrix} + \varepsilon_{v,t}, \quad (17)$$

where the points are based on the intrinsic and extrinsic camera matrices such that

$$\begin{pmatrix} l_{u,t} \\ l_{v,t} \\ \rho \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} R(q_t^{\text{bn}}) & -R(q_t^{\text{bn}})p_t^n \\ & \end{pmatrix} \begin{pmatrix} p_j \\ 1 \end{pmatrix}. \quad (18)$$

Here, $y_{j,t} \in \mathbb{R}^2$ denote the pixel coordinates of landmark p_j , f_x and f_y denote the focal lengths in the two different directions, c_x and c_y denote the origin of the image plane, and $\varepsilon_{v,t} \sim \mathcal{N}(0, \sigma_v^2 \mathcal{I}_2)$. Note that the model in Eq. (18) is linear in p_j conditioned on p_t^n and q_t^{bn} . The projection in Eq. (17) makes the model conditionally approximately linear.

4. Methods

We derive a particle smoothing algorithm for SLAM to jointly estimate the time-varying pose $x_{1:T}$ of the sensor and the static but initially unknown map θ it is navigating in. We assume that the (possibly nonlinear, non-Gaussian) dynamics of the state is modeled according to Eq. (8) and that the measurement model is nonlinear, but it is (approximately) linear when conditioned on the state, as modeled in Eq. (10). Furthermore, we assume that the map is constant and that prior map information is available according to Eq. (9). The SLAM problem is inherently unobservable, since it is possible to shift or rotate the entire map and trajectory (Gustafsson, 2012). To resolve this ambiguity, we assume that the initial pose is known, as is common practice in any SLAM formulation.

The joint smoothing distribution $p(x_{1:T}, \theta | y_{1:T})$ for the SLAM problem cannot be computed in closed form due to the nonlinear nature of the models Eqs. (8) and (10). Using a similar strategy as in Svensson et al. (2015) and Wigren et al. (2019), we instead approximate the joint smoothing distribution $p(x_{1:T}, \theta | y_{1:T})$ using Markov Chain Monte Carlo (MCMC). This approach has been shown to avoid the problem of particle degeneracy typically occurring in particle filters (Svensson et al., 2015) and has been shown to be equivalent to backward simulation strategies commonly used for particle smoothing (Lindsten et al., 2014). Contrary to Doucet et al. (2000) and Svensson et al. (2015), we assume that the conditionally linear parameter vector (the map) is not time-varying. The case of static, conditionally linear parameters is also studied by Wigren et al. (2019) and our algorithm can be considered to be a special case of their development. However, we will show that, contrary to Wigren et al. (2019), the specific structure of our model Eqs. (8)–(10) for SLAM allows us to compute certain terms in closed form.

In Section 4.1, we first introduce our MCMC smoothing algorithm for SLAM. This algorithm makes use of two specific implementations of the particle filter, which will subsequently be introduced in Sections 4.2 and 4.3. For notational simplicity, throughout this section we assume that the measurement model Eq. (10) is conditionally linear and we omit the explicit dependence of the dynamic model Eq. (8) on the inputs $u_{1:T}$. Note that the extension to a conditional approximately linear model is straightforward.

4.1. MCMC smoother for SLAM

Using a similar approach to Svensson et al. (2015) and Wigren et al. (2019), we approximate the joint smoothing distribution $p(x_{1:T}, \theta | y_{1:T})$ by generating K (correlated) samples using MCMC. In Svensson et al. (2015), each iteration of the MCMC algorithm used a conditional particle filter with ancestor sampling (AS). To also exploit the linear substructure in our problem, similar to Wigren et al. (2019), we use a conditional RBPF (CRBPF) with AS to generate samples $x_{1:T}[k], \theta[k]$, $k = 1, 2, \dots, K$. Our MCMC

smoother is presented in [Algorithm 1](#). It starts with an initial state trajectory $x_{1:T}[0]$ and then draws K samples from the Markov chain by running the CRBPF-AS for SLAM provided in [Algorithm 2](#). This algorithm makes use of the previously sampled state trajectory as an input, and outputs a new sampled state trajectory and its corresponding map. A natural choice for the initial state trajectory is the trajectory from a RBPF-AS. In [Section 4.2](#), we will first describe our RBPF-AS for SLAM. We will then describe the extension to a CRBPF-AS in [Section 4.3](#).

Algorithm 1: MCMC smoother for SLAM.

Input: Initial state trajectory $x_{1:T}[0]$ from the RBPF-AS described in [Section 4.2](#).

Output: K samples from the Markov chain with state trajectories $x_{1:T}[1], \dots, x_{1:T}[K]$ and the corresponding maps $\theta[1], \dots, \theta[K]$.

1. **for** $k = 1, 2, \dots, K$ **do**
 2. Run the CRBPF-AS from [Algorithm 2](#) conditional on $x_{1:T}[k-1]$ to obtain $x_{1:T}[k]$ and $\theta[k]$.
 3. **end for**
-

4.2. RBPF-AS for SLAM

In this section, we will first focus on deriving an RBPF-AS for SLAM that approximates the joint smoothing distribution $p(x_{1:T}, \theta | y_{1:T})$. We use a similar Rao-Blackwellization approach as Schön et al. (2005) and Wigren et al. (2019) and write the joint smoothing distribution at time t as

$$p(x_{1:t}, \theta | y_{1:t}) = p(\theta | x_{1:t}, y_{1:t}) p(x_{1:t} | y_{1:t}). \quad (19)$$

Contrary to Wigren et al. (2019), the specific form of our model for SLAM Eqs. (9) and (10) opens up for computing the distribution $p(\theta | x_{1:t}, y_{1:t})$ in the closed form

$$p(\theta | x_{1:t}, y_{1:t}) = \mathcal{N}(\theta; \hat{\theta}_t, P_t). \quad (20)$$

More specifically, $\hat{\theta}_t$ and P_t can be computed recursively as

$$\begin{aligned} \hat{\theta}_t &= \hat{\theta}_{t-1} + K_t \left(y_t - C(x_t) \hat{\theta}_{t-1} \right), & K_t &= P_{t-1} (C(x_t))^T (C(x_t) P_{t-1} C^T(x_t) + \Sigma)^{-1}, \\ P_t &= P_{t-1} - K_t C(x_t) P_{t-1}, \end{aligned} \quad (21)$$

with $\hat{\theta}_0, P_0$, respectively, equal to μ_θ, P_θ from [Eq. \(9\)](#).

In line with Wigren et al. (2019), we compute the distribution $p(x_{1:t} | y_{1:t})$ from [Eq. \(19\)](#) using SMC as (cf., Lindsten et al., 2014)

$$\hat{p}(x_{1:t} | y_{1:t}) = \sum_{i=1}^N w_t^i \delta_{x_{1:t}}^i(x_{1:t}). \quad (22)$$

Note that the Rao-Blackwellization allows us to use the particles in the SMC approach [Eq. \(22\)](#) to only represent the nonlinear states x_t . The Dirac delta in [Eq. \(22\)](#) implies that for each particle, x_t is deterministic and due to the conditionally linear structure of our model the map can be computed in closed form using [Eq. \(20\)](#). Hence, our approximation of the joint smoothing distribution $p(x_{1:t}, \theta | y_{1:t})$ makes use of N particles to represent the state trajectory $x_{1:t}$, where each particle carries its own map that is updated at each time step using [Eq. \(21\)](#). By writing $p(x_{1:t} | y_{1:t})$ as

$$p(x_{1:t} | y_{1:t}) = \frac{p(y_t | x_{1:t}, y_{1:t-1}) p(x_t | x_{1:t-1}, y_{1:t-1})}{p(y_t | y_{1:t-1})} p(x_{1:t-1} | y_{1:t-1}), \quad (23)$$

it can be seen that also the particles can be updated recursively. Using the Markov property of the state and the fact that our model [Eq. \(8\)](#) assumes that the dynamics of the state is independent of the map θ ,

$p(x_t | x_{1:t-1}, y_{1:t-1}) = p(x_t | x_{t-1})$ (note again that for notational simplicity we omitted the explicit conditioning on the input u_{t-1}). Furthermore, contrary to, for example, Svensson et al. (2015) and Wigren et al. (2019), the measurement model Eq. (10) for SLAM can be used to compute $p(y_t | x_{1:t}, y_{1:t-1})$ in closed form as

$$\begin{aligned} p(y_t | x_{1:t}, y_{1:t-1}) &= \int p(y_t | x_{1:t}, y_{1:t-1}, \theta) p(\theta | x_{1:t}, y_{1:t-1}) d\theta \\ &= \int p(y_t | x_t, \theta) p(\theta | x_{1:t-1}, y_{1:t-1}) d\theta \\ &= \mathcal{N}(y_t; C(x_t) \hat{\theta}_{t-1}, C(x_t) P_{t-1} (C(x_t))^T + \Sigma). \end{aligned} \tag{24}$$

Note that if the measurement model is only approximately linear, Eq. (24) will be an approximation. Similar to the commonly used approach of bootstrap particle filtering (Doucet et al., 2001), in our RBPF-AS, we propagate the particles using the dynamic model Eq. (8), compute their weights using Eq. (24) and resample the particles using systematic resampling. Specifically implementing this filter in terms of ancestor sampling (Lindsten et al., 2014) allows us to keep track of the history of each particle. The RBPF-AS for SLAM can be found in lines 1, 3, 4, 8–10, 12, 14, and 15 of Algorithm 2.

Algorithm 2: Conditional Rao-Blackwellized particle filter with ancestor sampling for SLAM.

Input: Reference state trajectory $x'_{1:T} = x_{1:T}[k-1]$, prior map mean and covariance μ_θ, P_θ , and initial pose x_0 .

Output: Sampled state trajectory $x_{1:T}[k]$ and its corresponding map $\theta[k]$.

1. Initialize $N - 1$ particles at initial pose as $x_1^i = x_0$ for $i = 1, \dots, N - 1$.
2. Add reference state trajectory by setting $x_t^N = x'_t$ for $t = 1, \dots, T$.
3. Initialize parameters using the prior $p(\theta)$ as $\hat{\theta}_0^i = \mu_\theta$ and $P_0^i = P_\theta$ for $i = 1, \dots, N$.
4. Set the initial weights equal to $w_1^i = 1/N$ for $i = 1, \dots, N$.
5. Optionally, precompute $C(x'_t)$ for $t = 2, \dots, T$.
6. **for** $t = 1, 2, \dots, T$ **do**
7. **if** $t > 1$ **then**
8. *Resampling and time update:*
9. Draw a_t^i with $\mathbb{P}(a_t^i = j) = w_{t-1}^j$ for $i = 1, \dots, N - 1$.
10. Draw x_t^i by sampling $x_t^i \sim p(x_t^i | x_{t-1}^{a_t^i})$ for $i = 1, \dots, N - 1$.
11. *Compute ancestor weights reference trajectory:*
 Draw a_t^N with $\mathbb{P}(a_t^N = i) \propto w_{t-1}^i p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1}) p(x'_t | x_{t-1}^i)$,
 where $p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1})$ is computed using Eq. (28).
12. Set $x_{1:t}^i = \{x_{1:t-1}^{a_t^i}, x_t^i\}$, $\hat{\theta}_{t-1}^i = \hat{\theta}_{t-1}^{a_t^i}$, and $P_{t-1}^i = P_{t-1}^{a_t^i}$ for $i = 1, \dots, N$.
13. **end if**
14. *Compute weights:*
 Set $w_t^i \propto p(y_t | x_{1:t}^i, y_{1:t-1})$ using Eq. (21) for $i = 1, \dots, N$ and normalize to $\sum_{i=1}^N w_t^i = 1$.
15. *Update parameters:*
 Compute $\hat{\theta}_t^i$ and P_t^i using Eq. (21).
16. **end for**
17. Sample a state trajectory and its corresponding map as $x_{1:T}[k] = x'_{1:T}$ and $\theta[k] = \theta_T^J$ with $\mathbb{P}(J = j) = w_T^j$.

4.3. CRBPF-AS for SLAM

During each iteration of the MCMC smoother from Algorithm 1, we run a CRBPF-AS to obtain a sample of $p(x_{1:T}, \theta | y_{1:T})$ using the weights of the trajectories and maps at time T . This sampled trajectory will be used as a so-called reference trajectory in the next iteration of the MCMC smoother (Andrieu et al., 2010). In practice, this means that we run an RBPF-AS as described in Section 4.2 with $N - 1$ particles and assign the reference trajectory to particle N for each time $t = 1, \dots, T$ (Svensson et al., 2015; Wigren et al., 2019). At each time instance, a history (ancestor index a_t^N) of this reference trajectory is sampled. In other words, we find a history for x'_t , where x'_t denotes the reference trajectory at time instance t . The ancestor index is drawn with probability (Svensson et al., 2014; Wigren et al., 2019)

$$\begin{aligned} \mathbb{P}(a_t^N = i) &\propto p(x_{1:t-1}^i | x'_{t:T}, y_{1:T}) \\ &\propto p(y_{t:T}, x'_{t:T} | x_{1:t-1}^i, y_{1:t-1}) p(x_{1:t-1}^i | y_{1:t-1}), \end{aligned} \tag{25}$$

where $p(x_{1:t-1}^i | y_{1:t-1})$ is equal to the importance weight w_{t-1}^i of particle i , $i = 1, \dots, N$. For our model, the term $p(y_{t:T}, x'_{t:T} | x_{1:t-1}^i, y_{1:t-1})$ can be rewritten according to

$$\begin{aligned} p(y_{t:T}, x'_{t:T} | x_{1:t-1}^i, y_{1:t-1}) &= p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1}) p(x'_{t:T} | x_{1:t-1}^i, y_{1:t-1}) \\ &= p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1}) p(x'_t | x_{t-1}^i) \prod_{\tau=t+1}^T p(x'_\tau | x'_{\tau-1}) \\ &\propto p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1}) p(x'_t | x_{t-1}^i). \end{aligned} \tag{26}$$

Note that the Markovian structure of the dynamics in the second line of Eq. (26) is due to the specific structure of our dynamic model Eq. (8). If the dynamic model would also depend on the map θ , as, for instance, assumed by Wigren et al. (2019), the marginalized dynamic model would not be Markovian or, in other words, in that case $p(x'_{t:T} | x_{1:t-1}^i, y_{1:t-1})$ would not be equal to $p(x'_t | x_{t-1}^i) \prod_{\tau=t+1}^T p(x'_\tau | x'_{\tau-1})$. Note also that in line 3 of Eq. (26) we have omitted the term $\prod_{\tau=t+1}^T p(x'_\tau | x'_{\tau-1})$ because it is equal for all particles and hence irrelevant for computing Eq. (25). The interpretation of Eqs. (25) and (26) is that the probability of drawing the ancestor index i depends on (i) the importance weight w_{t-1}^i , (ii) the probability of transitioning from x_{t-1}^i to the reference trajectory state x'_t according to the dynamic model in Eq. (8), and (iii) how well the map learned from $x_{1:t-1}^i, y_{1:t-1}$ can predict the current and future measurements $y_{t:T}$ at locations $x'_{t:T}$. The latter can be seen more clearly when writing $p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1})$ as

$$\begin{aligned} p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1}) &= \int p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, \theta) p(\theta | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1}) d\theta \\ &= \int p(y_{t:T} | x'_{t:T}, \theta) p(\theta | x_{1:t-1}^i, y_{1:t-1}) d\theta, \end{aligned} \tag{27}$$

where the explicit dependence on θ is introduced to write this probability distribution in terms of the measurement model $p(y_{t:T} | x'_{t:T}, \theta)$ and the map estimate $p(\theta | x_{1:t-1}^i, y_{1:t-1})$. Note that the conditioning of the first term inside the integral on $x_{1:t-1}^i, y_{1:t-1}$ is dropped since given θ all history is contained in x_t and that the conditioning of the second term inside the integral on $x_{1:t-1}^i$ is dropped since locations without measurements do not affect the map estimate. For our state-space model Eqs. (8)–(10), it is possible to compute $p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1})$ in closed form as

$$p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1}) = \mathcal{N}(\bar{y}_t; \bar{C}(x'_{t:T}) \hat{\theta}_{t-1}^i, \bar{C}(x'_{t:T}) P_{t-1}^i (\bar{C}(x'_{t:T}))^T + \bar{\Sigma}), \tag{28}$$

with

$$\bar{y}_t = (y_t^T \ y_{t+1}^T \ \dots \ y_T^T)^T, \quad \bar{C}(x'_{t:T}) = \left((C(x'_t))^T \ (C(x'_{t+1}))^T \ \dots \ (C(x'_T))^T \right)^T, \quad \bar{\Sigma} = \mathcal{I}_{T-t} \otimes \Sigma, \tag{29}$$

where \otimes denotes a Kronecker product. Note that if the measurement model is only approximately linear, Eq. (28) will be an approximation. Using Eq. (28) in Eq. (26), we can now sample ancestor indices for the reference trajectory at time $t = 1, 2, \dots, T$. Combining these steps with the RBPF-AS described in Section 4.2, the resulting conditional RBPF-AS for SLAM can be found in Algorithm 2.

5. Results

To validate and study the properties of our approach, we present simulation and experimental results. We first study the properties of our smoothing algorithm in terms of particle degeneracy and by analyzing the shape of the estimated trajectories for a simulated radio SLAM example in Section 5.1. Subsequently, we compare our estimation results for simulated magnetic SLAM with results from a particle filter and from an EKF in Section 5.2. In a third simulation experiment, we compare our estimation results for visual SLAM with results from a particle filter, an EKF and an extended Kalman smoother (EKS) in Section 5.3. Finally, in Section 5.4, we illustrate the efficacy of our algorithm to estimate a trajectory using magnetic SLAM based on real-world experimental data collected using a smartphone.

5.1. Simulation results for radio SLAM

To study the properties of Algorithm 1, we first simulate RSSI measurements and consider planar localization to estimate a time-varying position $p_t \in \mathbb{R}^2$ and a heading angle $q_t^{\text{nb}} \in \mathbb{R}$. We model the measurements according to the model Eq. (15) and the GP approximation Eq. (16), with a measurement noise covariance of $\sigma_r^2 = 0.01$. We define the GP prior through a squared exponential covariance function given by Eq. (12) that encodes a smoothness assumption on the model function. We use 128 basis functions, and fix the model hyperparameters to $\sigma_f^2 = 2$ (magnitude) and $\ell = 0.25$ (lengthscale). Furthermore, we use a square domain to ensure that the eigenfunctions for the GP approximation, see Section 2.2, can be computed in closed form (Solin and Särkkä, 2020). To avoid boundary effects, we ensure that all ground-truth positions are at least two lengthscales from the boundary of the domain.

We simulate odometry measurements $\Delta p_t, \Delta q_t$ that provide information about the change in position and heading at each time instance t , respectively, and model the discrete-time dynamics according to

$$p_{t+1}^n = p_t^n + R(q_t^{\text{nb}})\Delta p_t, \tag{30a}$$

$$q_{t+1}^{\text{nb}} = q_t^{\text{nb}} + \Delta q_t + w_t, \quad w_t \sim \mathcal{N}(0, \Delta T Q_t). \tag{30b}$$

Here, $\Delta T = 1$ and $R(q_t^{\text{nb}})$ denote the 2×2 rotation matrix representing a rotation around the z -axis with an angle q_t^{nb} . This model reflects the fact that odometry measurements are typically measured in a body-fixed frame while navigation is typically done in an earth-fixed frame. For illustrational purposes, we assume that the position odometry measurements Δp_t are noiseless, but model a noise w_t representing the error in the heading odometry measurements. More specifically, we simulate trajectories with $Q_t = 1 \cdot 10^{-6} \text{ rad}^2$ for all time instances where the user is moving straight. For the samples where a turn occurs, we model Q_t to be significantly larger ($Q_t = 0.1^2$ and $Q_t = 0.3^2$ for the examples in Figures 3 and 4, respectively). Such a model represents, for instance, a visual odometry where straight lines can be tracked very accurately but the odometry accuracy can have large uncertainties during fast turns.

For illustrative sanity-checks, we consider two different scenarios. First, we focus on a square trajectory where $Q_t = 0.1^2 \text{ rad}^2$ in the three 90° turns. This scenario is visualized in Figure 3a where both the true trajectory and the true RSSI field map are displayed. We then run the RBPF-AS from Section 4.2. Figure 3b visualizes the history of the 100 particles at time T as well as the RSSI map estimated by the particle with the highest weight at this time instance. The transparency of the map indicates its estimation uncertainty. Finally, we generate $K = 50$ samples $x_{1:T}[1], \dots, x_{1:T}[K], \theta[1], \dots, \theta[K]$ using Algorithm 1 and visualize these 50 trajectories as well as the RSSI map estimated during iteration K in Figure 3c. Again, the transparency of the map indicates the predictive marginal standard deviation (estimation uncertainty). In Figure 3b, it can be seen that the filter suffers from particle degeneracy as all

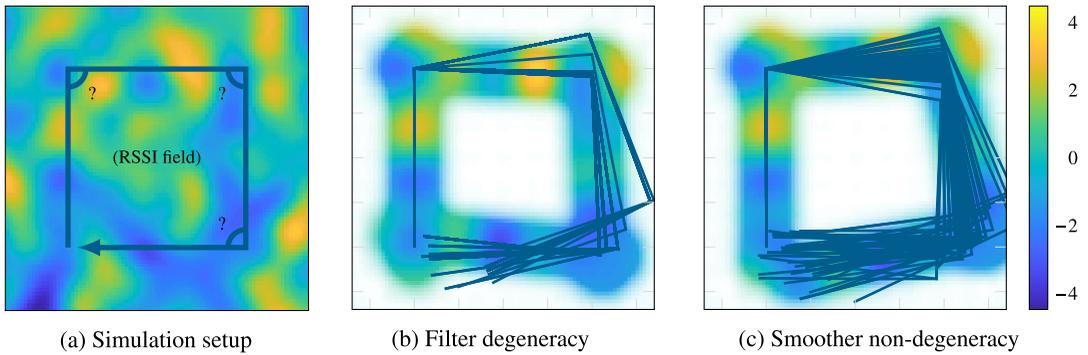


Figure 3. Highlight of the non-degeneracy of the particle smoothing approach with (a) the simulation setup, and simulation results for radio SLAM with (b) filtering samples showing that the filter is degenerate and (c) nondegenerate samples of the smoother.

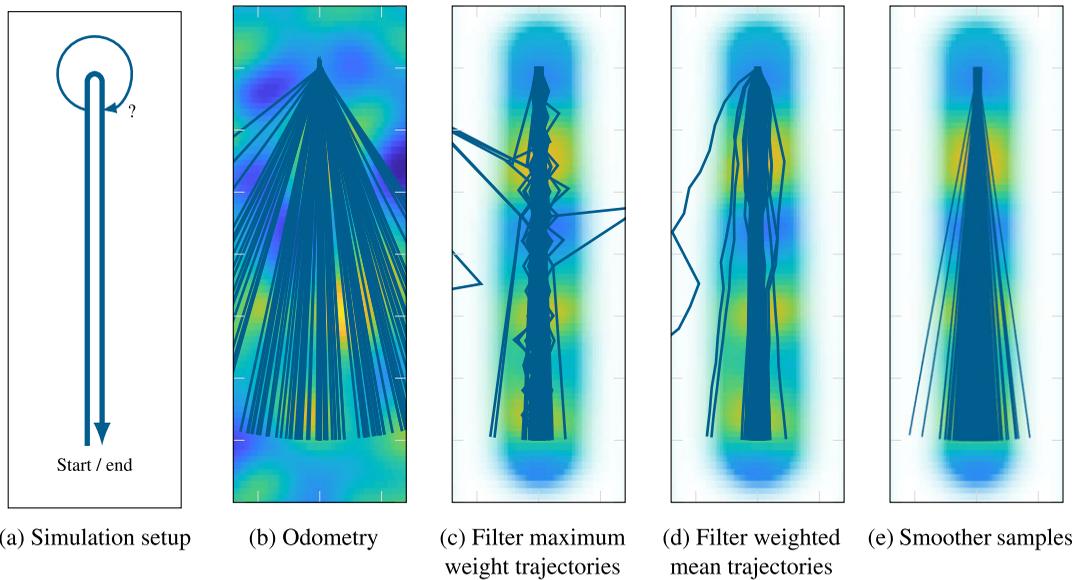


Figure 4. (a) Illustrative example of a back and forth path on an RSSI map; (b) the only source of uncertainty in the odometry is the turn angle at the farthest most point, which leads to spread; (c, d) Particle filtering SLAM reduces the spread, but does not backtrack the smooth odometry information; and (e) Our particle smoothing solution gives a tighter estimate and backtracks the smoothness along the sample paths.

particles at time T descend only from a small number of ancestors in the past. Please note that even though all particles seem to have a common ancestor at time $t = 1$, this is not due to particle degeneracy but due to how we initialize our SLAM problem: To overcome inherent unobservability in the problem, we use a standard approach in which we assume that the initial pose is known (Gustafsson, 2012). Nevertheless, there is clear particle degeneracy as all particles at time T descend from only five ancestors in the upper left corner and only seven ancestors in the upper right corner. The posterior as visualized by MCMC samples in Figure 3c can be seen to have a much wider spread representing possible trajectories. Note that the large spread is due to the large odometry uncertainty in each corner, and that this uncertainty decreases only when revisiting previous locations. This reduction of uncertainty will be illustrated in the next scenario.

In a second scenario, we assume that a user moves straight, turns and returns to the starting point as visualized in [Figure 4a](#). We run 100 Monte Carlo simulations where for each simulation we sample both the measurements of the field as well as the odometry. The 100 odometry trajectories as well as the true RSSI map are visualized in [Figure 4b](#). We run the filter from [Section 4.2](#) with 100 particles and for each of the Monte Carlo simulations we run 50 iterations of the smoother from [Algorithm 1](#). At each time instance of the filter, we save both the highest weight particle and the weighted mean particle. The resulting trajectories from the filter are shown in [Figure 4c,d](#). In [Figure 4c–e](#), we also visualize the estimated field for one of the Monte Carlo samples. The transparency again indicates the estimation uncertainty. For the smoother we visualize the field estimated by the K th sample. As can be seen, the sampled trajectories from [Algorithm 1](#) are significantly more smooth than the estimates of the filter from [Section 4.2](#). Furthermore, the estimates from the filter include some trajectories which significantly deviate from the true path while this is not the case for the samples from the particle smoother.

5.2. Simulation results for magnetic SLAM

We also study the properties of our algorithm for magnetic field SLAM. For this, we simulate a magnetic field using 512 basis functions and $\sigma_{\text{lin}}^2 = 650$, $\sigma_m^2 = 10$, $\ell = 1.3$, and $\sigma_f^2 = 200$. These hyperparameters are equal to the ones that we used previously in [Kok and Solin \(2018\)](#) and in line with values we found by estimating the hyperparameters from experimental data ([Solin et al., 2018](#)). We again use a square domain for the GP approximation and to avoid boundary effects, we ensure that the simulated trajectory, as shown in [Figure 5a](#) is at least two lengthscales from the boundary of the domain. We simulate the odometry using the following dynamic model

$$p_{t+1} = p_t + \Delta p_t + e_{p,t}, \tag{31a}$$

$$q_{t+1}^{\text{nb}} = q_{t+1}^{\text{nb}} \odot \Delta q_t \odot \exp_q(e_{q,t}), \tag{31b}$$

where $e_{p,t} \sim \mathcal{N}(0, \Delta T Q_p)$ and $e_{q,t} \sim \mathcal{N}(0, \Delta T Q_q)$, with $\Delta T = 0.01$ s, $Q_p = \text{diag}(0.25, 0.25, 0.01)$ and $Q_q = \text{diag}(0.01^2 \pi^2 / 180^2, 0.01^2 \pi^2 / 180^2, 0.3^2 \pi^2 / 180^2)$. We compare the results of our particle smoother with a particle filter similar to the one presented in [Kok and Solin \(2018\)](#) and an EKF similar to the one presented in [Viset et al. \(2022\)](#). To study the three algorithms for nonideal, real-world scenarios, we add a constant, unmodeled magnetometer bias of varying magnitude o to the y-axis of the measurements. Because we are not interested in the true magnetic field map but only in one that aids in obtaining a correct position, an incorrect bias estimate would not have any negative effect on the localization performance if the sensor would not rotate. However, rotating the sensor results in measuring the bias in a different direction in the navigation frame and hence inconsistencies in the map. The sensitivity of the algorithms to erroneous magnetometer calibration therefore depends on the shape of the trajectory. We use the trajectory shown in [Figure 5a](#), which is not that sensitive for erroneous calibration, for example, because the same path is not traversed twice in opposite directions. Because of this, we simulate large calibration errors of $o = 0, 1, 5, 10$. For each of these cases, we run 20 Monte Carlo simulations and sample $K = 10$ sample trajectories from our Rao-Blackwellized particle smoother. One of the maps is visualized in [Figure 5a](#). The resulting RMSE can be found in [Figure 5b](#). As can be seen, the RMSE of the particle smoother from [Algorithm 2](#) is smaller than that of the particle filter and remains consistently low, while that of the EKF increases for larger calibration errors.

5.3. Simulation results for visual SLAM

In a final set of simulations, we consider a visual SLAM problem. The previous experiments considered the case where our model was conditionally linear, whereas the sparse visual model is conditionally *linearized* (see [Section 3.3](#)). For simulation purposes, we consider the two-dimensional setup from

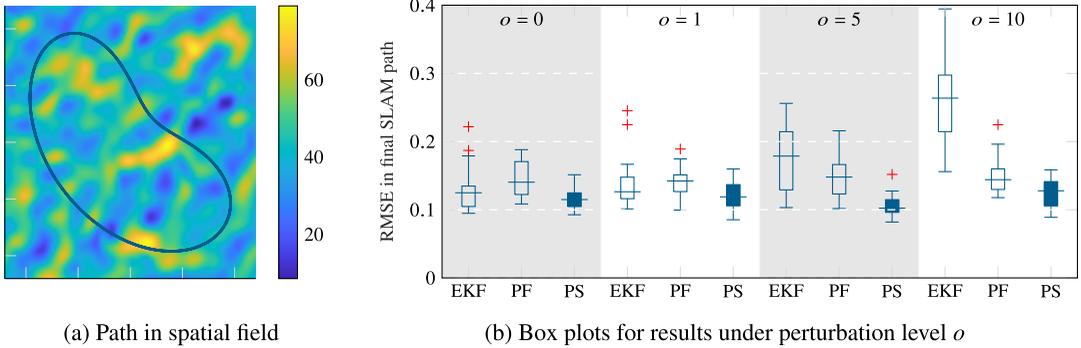


Figure 5. Robustness study with magnetometer perturbed with a constant bias o . Left: Setup showing the ground-truth path and one realization of the magnetic field map. Right: Box plots of RMSE in the final estimated SLAM path. Particle and extended Kalman filtering methods drawn with outlines, particle smoothing method in solid colors. The EKF performs well under negligible calibration errors, the particle filter (PF) and smoother (PS) perform well under large calibration errors.

Figure 1. In two spatial dimensions, the pinhole camera observations become one-dimensional and the observation model becomes

$$y_{j,t} = \frac{l_t}{\rho} + \varepsilon_{v,t}, \quad \text{with} \quad \begin{pmatrix} l_t \\ \rho \end{pmatrix} = \begin{pmatrix} f & c \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R(q_t^{\text{bn}}) & -R(q_t^{\text{bn}})p_t^{\text{n}} \\ & 1 \end{pmatrix} \begin{pmatrix} p_j \\ 1 \end{pmatrix}, \quad (32)$$

denotes the origin of the image line, and $\varepsilon_{v,t} \sim \mathcal{N}(0, \sigma_v^2)$. The task is to learn both the sparse map of $\theta = \{p_j\}_{j=1}^L$ and the 3-DoF trajectory of the camera. In the experiment, we use $f = 1.5$, $c = 0$ (the field-of-view is $\sim 67^\circ$), and $\sigma_v^2 = 0.1^2$. The dynamical model for the camera movement follows the setup in Eq. (31) with $\Delta T = 1$ and a total of $T = 197$ time steps. To corrupt the odometry signal, we add a constant drift of 1 cm/s and Gaussian noise with covariance $Q_p = 0.04^2 \mathcal{I}_2/s$ to the spatial increments Δp_t , and add a small amount of noise ($Q_q = 10^{-12}/s$) to the orientation increments. This setup corresponds to a typical visual-inertial odometry setup, where tracking orientation with the help of a gyroscope is easier than tracking the twice-integrated accelerometer signal. In the simulation setup, the camera traverses around the space two times, which means that learned landmark points are revisited and should improve the tracking.

A recurring issue in visual SLAM problems is the initialization of the feature point locations p_j . The initial guess is typically vague and could be, for example, based on triangulation from only two views, where errors in the pose of these views translate to large errors in the initialization. We study robustness to feature point initialization by controlling the initial error of the initialization in a simulated setting. We follow a similar setup as Solin et al. (2022, Sec. IV.A), where we initialize the points by taking their ground-truth locations and corrupting them with Gaussian noise (with variance σ^2). The visual SLAM setting follows that visualized in Figure 1a, where we have $L = 20$ feature points—most of which are not observed at the same time.

As explained in Section 3.3, the sparse visual SLAM model is a conditionally linearized filter/smoothing. Thus, it falls natural to compare Rao-Blackwellized particle filtering/smoothing (denoted PF/PS) to a vanilla EKF/EKS. For this two-dimensional simulated SLAM problem, we use $N = 100$ particles for the RBPF-AS and sample $K = 10$ sample trajectories from our Rao-Blackwellized particle smoother. We repeat the experiment with 20 random repetitions for each noise level σ^2 . The prior state covariance corresponding to the feature location is initialized to $4^2 \mathcal{I}_2$, which means that the scale of the corrupting noise can be considered moderate even for large noise scales.

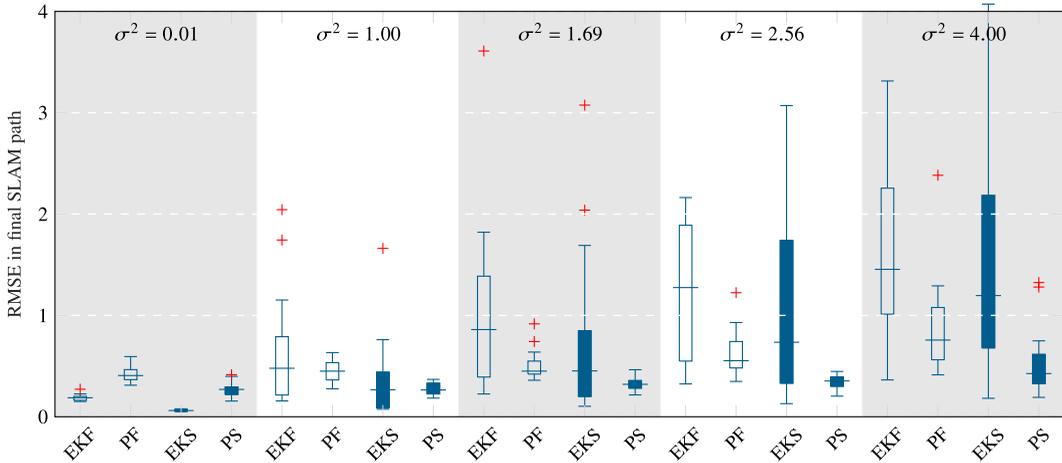


Figure 6. Robustness to feature point initialization in visual SLAM. Simulation study with initial feature locations perturbed with Gaussian noise (variance σ^2). Box plots of RMSE in the final estimated SLAM path. Forward filtering methods drawn with outlines, smoothing methods in solid colors. The EKF/EKS perform well under negligible perturbation, the particle filter (PF) and smoother (PS) perform well under large uncertainty.

Figure 6 shows results for controlling the feature point initialization. For each noise level, we show box plots for the final error of the weighted mean estimates compared to the ground-truth translation of the moving camera. We apply standard Procrustes correction (rotation, translation, and scalar scaling) based on the learned map points due to possible lack of scale information in the visual-only observations. The results in the box plots are as expected: The effect of deviating from the vicinity of the “true” linearization point is apparent, and the EKF and EKS performance deteriorates quickly as the noise scale grows. The particle filter appears robust to the initialization when compared to the EKF, and the particle smoother shows clearly improved robustness over the EKS.

5.4. Empirical results for magnetic field SLAM

To experimentally validate our algorithm, we use experimental data previously used for magnetic field-based SLAM in Kok and Solin (2018). In this experiment, we collected magnetometer and odometry data using an Apple iPhone 6s. To obtain the odometry data, we used the app ARKit which uses IMU and camera data to provide a 6-DoF (position and orientation) movement trajectory. ARKit visually recognizes previously visited locations and corrects the trajectory for this, resulting in discontinuities in the trajectories. We instead removed these discontinuities, resulting in a drifting odometry shown in Figure 7b. The goal of our SLAM algorithm is then to remove the drift in this trajectory. To this end, we use the dynamic model Eq. (31) and the measurement model Eq. (11) with the reduced-rank approximation from Eq. (13) and down-sample the data from the original 100 Hz to 10 Hz. We use the same hyperparameters as in Kok and Solin (2018): $\sigma_{\text{lin}}^2 = 650$, $\sigma_m^2 = 10$, $\ell = 1.3$ m and $\sigma_r^2 = 200$. Furthermore, we have $\Delta T \approx 0.1$, $\Sigma_p = \text{diag}(0.1^2, 0.1^2, 0.02^2)$, $\Sigma_q = \text{diag}(0.01^2 \pi^2 / 180^2, 0.01^2 \pi^2 / 180^2, 0.24^2 \pi^2 / 180^2)$.

Even though, we downsample the data from 100 Hz to 10 Hz, the computational complexity of both the particle filter as well as the particle smoother for this example quickly becomes prohibitively large. The reason for this is not only the large number of data points but also the large area that is covered and the fact that the number of basis functions that is needed to make a good approximation scales with this (Solin and Särkkä, 2020). Because of this, similar to our approach in Kok and Solin (2018), we use smaller hexagonal block domains (each with a radius of 5 m, a height of 2 m and 256 basis functions), as visualized in Figure 7d. Details on the basis functions can be found in Kok and Solin (2018). Note that similar to Kok

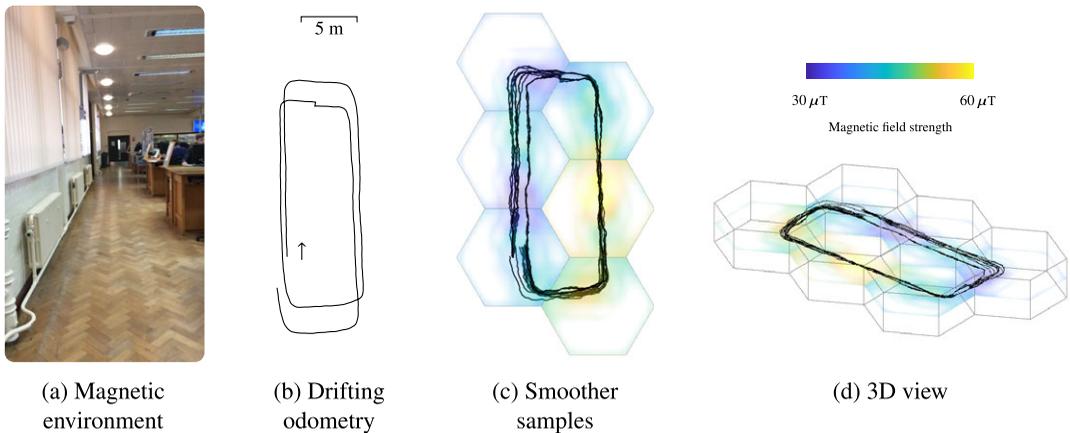


Figure 7. Empirical proof-of-concept magnetic indoor SLAM. Panel (a) shows a view through the mobile phone camera, (b) the drifting odometry, (c) samples of the smoothing distribution, and (d) a 3D view of these results. The color scaling visualizes the field strength of a learned magnetic map.

and Solin (2018), we model the actual hexagonal blocks to be slightly larger than the area that we use to reduce boundary effects. In principle, the hexagonal blocks can straightforwardly be included in Algorithms 1 and 2, except for line 10 in Algorithm 2. In that line, we compute the ancestor weights of the reference trajectory and hence compute the likelihood of every future measurement $y_{t:T}$ given the previous measurements $y_{1:t-1}$, the previous locations of that particle $x_{1:t-1}^i$ and the future locations of the reference trajectory $x'_{t:T}$. Note that $x'_{t:T}$ can span multiple hexagons. We therefore check for the whole reference trajectory $x'_{t:T}$ in which hexagon it lies. If a map has been started in this hexagon by any of the particles, we compute the likelihood according to Eqs. (28) and (29) either using the estimated map or using the prior (in case that particle has not yet started that hexagon). The part of the reference trajectory that is in hexagons not created by any of the particles contributes only a constant offset on the weights and is therefore omitted.

6. Discussion

We have presented a probabilistic approach for SLAM problems under the smoothing setup, *that is*, for conditioning the entire trajectory on all observed data. This is a particularly challenging problem as the state and parameter space become very high dimensional, which makes the general problem largely intractable. Key to our setup is to leverage the conditionally linear structure—through Rao-Blackwellization—that separates the map parameters θ from the poses x . The smoothing approach should in general not be considered a real-time method as the idea is to jointly consider all data over the entire time-horizon (incorporating knowledge from the future into past states). Compared to real-time filtering approaches, smoothing also adds to the computational load. Our method (see Algorithm 1) requires running a conditional particle filter for each sample k , which would translate to a cost of K times the cost of running a particle filter. However, Line 11 in Algorithm 2 is the most computationally heavy step of our algorithm as the computation of $p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1})$ requires a prediction along the entire future trajectory at every time instance. As shown in Eq. (28), this distribution is Gaussian with a covariance $\bar{C}(x'_{t:T}) P_{t-1}^i (\bar{C}(x'_{t:T}))^T$. Computing this covariance has a computational complexity in the order of $\mathcal{O}(mT^2 + m^2T)$ as $\bar{C}(x'_{t:T}) \in \mathbb{R}^{n_y(T-t) \times m}$, where n_y is the dimension of the measurement at time t , and $P_{t-1}^i \in \mathbb{R}^{m \times m}$. This results in an overall computational complexity of Algorithm 1 of $\mathcal{O}(KNT^3m + KNm^2T^2)$. Note that this finding is contrary to the linear time complexity $\mathcal{O}(T)$ reported in Wigren et al. (2019) even though our smoothing algorithm for SLAM can be seen as a special case of

their algorithm. In their work, $p(y_{1:T} | x_{1:t-1}^i, x'_{1:T}, y_{1:t-1})$ cannot be computed in closed form and because of this, they propagate sufficient statistics. We can use a similar strategy by implementing our recursive linear parameter estimation in information form. In other words, instead of estimating a mean $\hat{\theta}_t$ and covariance P_t , we can equivalently estimate an information vector $\iota_t = I_t \theta_t$ and an information matrix $I_t = P_t^{-1}$. The details on how our algorithm can be implemented in information form are available in [Appendix A](#). This implementation has a computational complexity of $\mathcal{O}(KNTm^3)$. In other words, our algorithm either scales cubically with the number of time steps T or cubically with the number of map parameters m . In practice, the implementation from [Appendix A](#) is therefore often preferred over direct implementation of [Algorithm 2](#). However, in the case where the map is represented using a large number of basis functions approximating a GP, it highly depends on the exact number of basis functions and time steps which implementation is most efficient. In [Section 5.4](#), we partially overcame this computational complexity by only computing this quantity for hexagonal block domains that have previously been created, avoiding unnecessary computations. In future work, it would be interesting to explore more ways of reducing the computational complexity, for example, by only doing predictions into the near future, or by doing independent instead of joint predictions.

When using [Algorithm 2](#) for other SLAM problems than the ones studied in this work, care should be taken in at least two respects. First, for conditionally approximately linear measurement models, the performance of [Algorithm 2](#) naturally depends on the quality of the linearization and will therefore be model dependent. Furthermore, in principle, one would expect that we would have to discard the first samples of the smoother to allow for convergence of the Markov chain. However, because we did not observe a clear convergence effect, throughout [Section 5](#) we did not discard any samples of the smoother. We suspect that we did not observe the convergence effect since the RBPF-AS initializes the Markov chain close to the smoothing distribution using a RBPF. However, care should be taken when using [Algorithm 2](#) for other smoothing problems and it should be checked if also in those cases it is not necessary to discard samples due to burn-in.

There are several promising and more or less straightforward extensions of [Algorithm 2](#) that would be interesting to explore in future work. First, we use bootstrap particle filtering, in which the proposal distribution is determined by the dynamic model. Our method can straightforwardly be extended to other proposal distributions (e.g., the one used in Montemerlo et al., 2003). Second, in the case of approximately linear models, the updates of the map in [Eq. \(21\)](#) are similar to measurement updates in an EKF. These updates could be replaced, for example, with measurement updates similar to an unscented Kalman filter. Unscented Kalman filters have been shown to perform better than EKFs for visual SLAM in Solin et al. (2022). Further experiments would also be needed for studying how smoothing approaches could improve the state-of-the-art in filtering-based visual-inertial odometry and SLAM (see Seiskari et al., 2022).

7. Conclusion

This paper introduced a framework for probabilistic SLAM using particle smoothing that does not only incorporate observed data in current state estimates, but also backtracks the updated knowledge to correct for past drift and ambiguities in both the map and in the states. Our solution can handle both dense and sparse map representations by Rao-Blackwellization of conditionally linear and conditionally linearized models. We structured the framework to cover a variety of SLAM paradigms, both for dense function-valued maps (typically used in magnetic and radio RSSI anomaly SLAM) and sparse feature-point-based maps (typically used in visual and radio RSSI emitter SLAM). The algorithm allows for modeling that the map is constant over time and for including a priori assumptions regarding the map, for example, important for magnetic SLAM.

The proposed algorithm alleviates particle degeneracy, results in smooth estimated trajectories, and is robust against calibration and initialization errors—features that were all demonstrated in the experiments. Interesting directions of future work include exploring ways to reduce the computational

complexity of the algorithm and changing the proposal distribution of the particle filter or the measurement updates for conditionally approximately linear models.

Data availability statement. We provide proof-of-concept reference implementations and data for replicating the experiments. This code and data can be found on <https://github.com/manonkok/Rao-Blackwellized-SLAM-smoothing>.

Acknowledgments. The authors would like to thank Anna Wigren and Fredrik Lindsten for the technical discussions related to the computational complexity of Line 11 in [Algorithm 2](#). The authors would also like to thank the reviewers for their detailed and important suggestions that helped them improve the quality of the manuscript.

Author contribution. Conceptualization: M.K., A.S., T.S.; Data curation: M.K., A.S.; Data visualization: M.K., A.S.; Methodology: M.K., A.S., T.S.; Writing original draft: M.K., A.S. All authors approved the final submitted draft.

Funding statement. This publication is part of the project “*Sensor Fusion for Indoor Localization Using the Magnetic Field*” with project number 18213 of the research program Veni which is (partly) financed by the Dutch Research Council (NWO). The research was also supported by grants from the Academy of Finland (grant ID: 339730), by the project “*NewLEADS—New Directions in Learning Dynamical Systems*” (contract number: 621-2016-06079) funded by the Swedish Research Council, and by the Kjell och Märta Beijer Foundation. The authors acknowledge the computational resources provided by the Aalto Science-IT project.

Competing interest. The authors declare none.

Ethical standard. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

References

- Andrieu C and Doucet A (2002) Particle filtering for partially observed Gaussian state space models. *Journal of the Royal Statistical Society* 64(4), 827–836.
- Andrieu C, Doucet A and Holenstein R (2010) Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society. Series B (Methodological)* 72(2), 1–33.
- Bailey T and Durrant-Whyte H (2006) Simultaneous localization and mapping (SLAM): Part II. *IEEE Robotics & Automation Magazine* 13(3), 108–117.
- Bailey T, Nieto J, Guivant J, Stevens M and Nebot E (2006) Consistency of the EKF-SLAM algorithm. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Beijing, China: IEEE, pp. 3562–3568.
- Barrau A and Bonnabel S (2015) An EKF-SLAM algorithm with consistency properties. <https://arxiv.org/abs/1510.06263>.
- Berntorp K and Nordh J (2014) Rao-Blackwellized particle smoothing for occupancy-grid based SLAM using low-cost sensors. *IFAC Proceedings* 47(3), 10174–10181.
- Bloesch M, Czarnowski J, Clark R, Leutenegger S and Davison AJ (2018) CodeSLAM — learning a compact, optimisable representation for dense visual SLAM. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, USA: IEEE.
- Cadena C, Carlone L, Carrillo H, Latif Y, Scaramuzza D, Neira J, Reid I and Leonard JJ (2016) Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. *IEEE Transactions on Robotics* 32(6), 1309–1332.
- Chen R and Liu JS (2000) Mixture Kalman filters. *Journal of the Royal Statistical Society. Series B (Methodology)* 62(3), 493–508.
- Chopin N and Papaspiliopoulos O (2020) *An Introduction to Sequential Monte Carlo*. Berlin: Springer.
- Coulin J, Guillemard R, Gay-Bellile V, Joly C and de La Fortelle A (2021) Tightly-coupled magneto-visual-inertial fusion for long term localization in indoor environment. *IEEE Robotics and Automation Letters* 7(2), 952–959.
- Davison AJ, Reid ID, Molton ND and Stasse O (2007) MonoSLAM: real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(6), 1052–1067.
- Doucet A, de Freitas N and Gordon N (2001) *Sequential Monte Carlo Methods in Practice*. New York: Springer.
- Doucet A, de Freitas N, Murphy K and Russell S (2000) Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Conference of Uncertainty in Artificial Intelligence (UAI)*. San Francisco, USA: Morgan Kaufmann, pp. 176–183.
- Doucet A and Johansen AM (2011) A tutorial on particle filtering and smoothing: fifteen years later. In Crisan D and Rozovsky B (eds.), *Nonlinear Filtering Handbook*. Oxford: Oxford University Press.
- Durrant-Whyte H and Bailey T (2006) Simultaneous localization and mapping: Part I. *IEEE Robotics & Automation Magazine* 13(2), 99–110.
- Ferris B, Fox D and Lawrence ND (2007) WiFi-SLAM using Gaussian process latent variable models. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*. Hyderabad, India: AAAI Press / International Joint Conferences on Artificial Intelligence, pp. 2480–2485.
- Gordon NJ, Salmond DJ and Smith AFM (1993) Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEEE Proceedings on Radar and Signal Processing* 140, 107–113.

- Grisetti G, Kümmerle R, Stachniss C and Burgard W** (2010) A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine* 2(4), 31–43.
- Gustafsson F** (2012) *Statistical Sensor Fusion*. Lund: Studentlitteratur.
- Gustafsson F, Gunnarsson F, Bergman N, Forssell U, Jansson J, Karlsson R and Nordlund P-J** (2002) Particle filters for positioning, navigation, and tracking. *IEEE Transactions on Signal Processing* 50(2), 425–437.
- Hartley RI and Zisserman A** (2004) *Multiple View Geometry in Computer Vision*. Cambridge, UK: Cambridge University Press.
- Hensman J, Fusi N and Lawrence ND** (2013) Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*. Bellevue, Washington: AUAI Press, pp. 282–290.
- Kerl C, Sturm J and Cremers D** (2013) Dense visual SLAM for RGB-D cameras. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Tokyo, Japan: IEEE, pp. 2100–2106.
- Kitagawa G** (1993) A Monte Carlo filtering and smoothing method for non-Gaussian nonlinear state space models. In *Proceedings of the 2nd US-Japan Joint Seminar on Statistical Time Series Analysis*. Honolulu, Hawaii, pp. 110–131.
- Kok M and Solin A** (2018) Scalable magnetic field SLAM in 3D using Gaussian process maps. In *Proceedings of the 21st International Conference on Information Fusion*. Cambridge, UK: IEEE.
- Lindsten F, Jordan MI and Schön TB** (2014) Particle Gibbs with ancestor sampling. *Journal of Machine Learning Research* 15 (1), 2145–2184.
- Lindsten F and Schön TB** (2013) Backward simulation methods for Monte Carlo statistical inference. *Foundations and Trends in Machine Learning* 6(1), 1–143.
- Montemerlo M, Thrun S, Koller D and Wegbreit B** (2002) FastSLAM: a factored solution to the simultaneous localization and mapping problem. In *Proceedings of the 18th AAAI National Conference on Artificial Intelligence*. Edmonton, Canada, pp. 593–598.
- Montemerlo M, Thrun S, Koller D, and Wegbreit B** (2003) FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*. Acapulco, Mexico: Morgan Kaufmann, pp. 1151–1156.
- Mourikis AI and Roumeliotis SI** (2007) A multi-state constraint Kalman filter for vision-aided inertial navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Rome, Italy: IEEE, pp. 3565–3572.
- Naesseth AC, Lindsten F and Schön TB** (2019) Elements of sequential Monte Carlo. *Foundations and Trends in Machine Learning* 12(3), 307–392.
- Quinonero-Candela J and Rasmussen CE** (2005) A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research* 6, 1939–1959.
- Rasmussen CE and Williams CKI** (2006) *Gaussian Processes for Machine Learning*. Cambridge, Massachusetts, USA: MIT Press.
- Särkkä S** (2013) *Bayesian Filtering and Smoothing*. UK: Cambridge University Press.
- Schön TB, Gustafsson F and Nordlund P-J** (2005) Marginalized particle filters for mixed linear/nonlinear state-space models. *IEEE Transactions on Signal Processing* 53(7), 2279–2289.
- Schön TB, Wills A and Ninness B** (2011) System identification of nonlinear state-space models. *Automatica* 47(1), 39–49.
- Seiskari O, Rantalankila P, Kannala J, Ylilammi J, Rahtu E and Solin A** (2022) HybVIO: pushing the limits of real-time visual-inertial odometry. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Waikoloa, Hawaii: IEEE, pp. 701–710.
- Solin A and Kok M** (2019) Know your boundaries: constraining Gaussian processes by variational harmonic features. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*. Naha, Okinawa, Japan: PMLR, pp. 2193–2202.
- Solin A, Kok M, Wahlström N, Schön TB and Särkkä S** (2018) Modeling and interpolation of the ambient magnetic field by Gaussian processes. *IEEE Transactions on Robotics* 34(4), 1112–1127.
- Solin A, Li R and Pilzer A** (2022). A look at improving robustness in visual-inertial SLAM by moment matching. In *Proceedings of the International Conference on Information Fusion*. Linköping, Sweden: IEEE, pp. 1–8.
- Solin A and Särkkä S** (2020) Hilbert space methods for reduced-rank Gaussian process regression. *Statistics and Computing* 30, 419–446.
- Solin A, Särkkä S, Kannala J and Rahtu E** (2016) Terrain navigation in the magnetic landscape: particle filtering for indoor positioning. In *Proceedings of the European Navigation Conference (ENC)*. Helsinki, Finland: IEEE, pp. 1–9.
- Stachniss C, Leonard JJ and Thrun S** (2016) *Simultaneous Localization and Mapping*. Cham: Springer Handbook of Robotics, pp. 1153–1176.
- Stewart L and McCarty P** (1992) The use of Bayesian belief networks to fuse continuous and discrete information for target recognition and discrete information for target recognition, tracking, and situation assessment. *Proceedings of SPIE Signal Processing, Sensor Fusion and Target Recognition* 1699, 177–185.
- Svensson A, Schön TB and Kok M** (2015) Nonlinear state space smoothing using the conditional particle filter. *IFAC-PapersOnLine* 48(28), 975–980.
- Svensson A, Schön TB and Lindsten F** (2014) Identification of jump Markov linear models using particle filters. In *Proceedings of the 53rd IEEE Annual Conference on Decision and Control (CDC)*. Los Angeles, CA: IEEE, pp. 6504–6509.
- Thrun S and Montemerlo M** (2006) The graph SLAM algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research* 25(5–6), 403–429.

Vallivaara I, Haverinen J, Kemppainen A, and Rönning J (2010) Simultaneous localization and mapping using ambient magnetic field. In *Proceedings of the IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. Salt Lake City, UT: IEEE, pp. 14–19.

Viset FM, Helmons R and Kok M (2022) An extended Kalman filter for magnetic field SLAM using Gaussian process regression. *MDPI Sensors* 22(8), 2833.

Whelan T, Leutenegger S, Salas-Moreno R, Glocker B and Davison A (2015) ElasticFusion: dense SLAM without a pose graph. In *Proceedings of Robotics: Science and Systems (RSS)*. Rome, Italy. www.roboticsproceedings.org/

Wigren A, Risuleo RS, Murray L and Lindsten F (2019) Parameter elimination in particle Gibbs sampling. In *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS)*. Vancouver, Canada: Curran Associates, Inc.

Wigren A, Wågberg J, Lindsten F, Wills AG and Schön TB (2022) Nonlinear system identification: learning while respecting physical models using a sequential Monte Carlo method. *IEEE Control Systems Magazine* 42(1), 75–102.

Appendix

A. Estimating the parameters in information form

Any recursive least squares or Kalman filter problem can be written in information form (Gustafsson, 2012). It is therefore possible to slightly adapt the method from Section 4 and estimate an information vector $l_t = I_t \hat{\theta}_t$ and information matrix $I_t = P_t^{-1}$ rather than estimating $\hat{\theta}_t$ and P_t . The time recursion Eq. (21) to update the parameter estimates in information form is

$$l_t = l_{t-1} + (C(x_t))^T \Sigma^{-1} y_t, \quad I_t = I_{t-1} + (C(x_t))^T \Sigma^{-1} C(x_t). \tag{33}$$

To compute $p(y_t | x_{1:t}, y_{1:t-1})$ using the information vector and matrix, let us first write the Gaussian distribution $p(\theta | x_{1:t}, y_{1:t})$ as

$$p(\theta | x_{1:t}, y_{1:t}) = \frac{\exp\left(-\frac{1}{2} l_t^T I_t^{-1} l_t\right)}{\sqrt{(2\pi)^m \det I_t^{-1}}} \exp\left(-\frac{1}{2} (\theta^T I_t \theta - 2\theta^T l_t)\right), \tag{34}$$

where we separated the terms that depend on θ from the terms that do not depend on θ . We can then write $p(y_t | x_{1:t}, y_{1:t-1})$ as

$$\begin{aligned} p(y_t | x_{1:t}, y_{1:t-1}) &= \int p(y_t | x_t, \theta) p(\theta | x_{1:t-1}, y_{1:t-1}) d\theta \\ &= \frac{\exp\left(-\frac{1}{2} l_{t-1}^T I_{t-1}^{-1} l_{t-1}\right) \exp\left(-\frac{1}{2} y_t^T \Sigma^{-1} y_t\right)}{\sqrt{(2\pi)^m \det I_{t-1}^{-1}} \sqrt{(2\pi)^{m_y} \det \Sigma}} \\ &\quad \int \exp\left(-\frac{1}{2} (\theta^T (I_{t-1} + (C(x_t))^T \Sigma^{-1} C(x_t)) \theta - 2\theta^T (l_{t-1} + (C(x_t))^T \Sigma^{-1} y_t))\right) d\theta \\ &= \frac{\exp\left(-\frac{1}{2} l_{t-1}^T I_{t-1}^{-1} l_{t-1}\right) \exp\left(-\frac{1}{2} y_t^T \Sigma^{-1} y_t\right) \sqrt{(2\pi)^m \det I_t^{-1}}}{\sqrt{(2\pi)^m \det I_{t-1}^{-1}} \sqrt{(2\pi)^{m_y} \det \Sigma} \exp\left(-\frac{1}{2} l_t^T I_t^{-1} l_t\right)}, \end{aligned} \tag{35}$$

where we used the definition of the update equations from Eq. (33), the fact that the term in the integral in the second step of Eq. (35) has the same form as Eq. (34) except for the terms that are independent of θ , and the fact that probability distributions integrate to one.

To compute the ancestor weights of the reference trajectory, we can compute $p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1})$ as

$$\begin{aligned} p(y_{t:T} | x_{1:t-1}^i, x'_{t:T}, y_{1:t-1}) &= \prod_{\tau=t}^T p(y_\tau | x_{1:t-1}^i, x'_{\tau:T}, y_{1:t-1}) \\ &\propto \frac{\exp\left(-\frac{1}{2} (l_{t-1}^i)^T (I_{t-1}^i)^{-1} l_{t-1}^i\right) \sqrt{(2\pi)^m \det \left((I_T^i)^{-1} \right)}}{\sqrt{(2\pi)^m \det \left((I_{t-1}^i)^{-1} \right)} \exp\left(-\frac{1}{2} (l_T^i)^T (I_T^i)^{-1} l_T^i\right)}, \end{aligned} \tag{36}$$

where we used Eq. (35) and omitted the terms that only depend on y_t and Σ since they are the same for each particle i . Note that the particularly elegant expression Eq. (36) is due to the cancellation of terms because of the special structure in Eq. (35). Furthermore, note that

$$l_T^i = l_0^i + \underbrace{\sum_{\tau=1}^{t-1} C(x_\tau^i)^T \Sigma^{-1} y_\tau}_{\tilde{l}_t^i} + \underbrace{\sum_{\tau=t}^T C(x'_\tau)^T \Sigma^{-1} y_\tau}_{\tilde{l}_T^i}, \tag{37a}$$

$$I_T^i = I_0^i + \underbrace{\sum_{\tau=1}^{t-1} C(x_\tau^i)^T \Sigma^{-1} C(x_\tau^i)}_{\tilde{I}_T} + \underbrace{\sum_{\tau=t}^T C(x_\tau^i)^T \Sigma^{-1} C(x_\tau^i)}_{\tilde{I}_T}, \tag{37b}$$

where \tilde{I}_T and \tilde{I}_T^i are independent of the particles and their parameters estimates. Because of this, it is possible to precompute each term in the sum and only add them once per iteration of the particle filter.

The computational complexity of both Eq. (35) and Eq. (36) is dominated by the inversion of the information matrix and is of order $\mathcal{O}(m^3)$. It is possible to avoid this cubic computational complexity in Eq. (35) by not only estimating ι_t and I_t but also P_t since this directly gives I_t^{-1} with only a complexity $\mathcal{O}(m^2)$ and since

$$\det I_t^{-1} = \frac{1}{\det(I_{t-1} + (C(x_t))^T \Sigma^{-1} C(x_t))} = \frac{1}{\det(\Sigma + C(x_t) I_{t-1}^{-1} (C(x_t))^T) \det(\Sigma^{-1}) \det(I_{t-1})} = \frac{\det(\Sigma) \det(P_{t-1})}{\det(\Sigma + C(x_t) P_{t-1} (C(x_t))^T)}. \tag{38}$$

It is, however, not possible to avoid the cubic complexity in Eq. (36) without introducing additional scaling with T as computing P_T from P_{t-1} is of order $\mathcal{O}(Tm^2)$ for every particle and every time instance.