



Delft University of Technology

Spatially scalable recursive estimation of Gaussian process terrain maps using local basis functions

Viset, Frida; Helmons, Rudy; Kok, Manon

DOI

[10.1109/TSP.2025.3549966](https://doi.org/10.1109/TSP.2025.3549966)

Publication date

2025

Document Version

Final published version

Published in

IEEE Transactions on Signal Processing

Citation (APA)

Viset, F., Helmons, R., & Kok, M. (2025). Spatially scalable recursive estimation of Gaussian process terrain maps using local basis functions. *IEEE Transactions on Signal Processing*, 73, 1444-1453.
<https://doi.org/10.1109/TSP.2025.3549966>

Important note

To cite this publication, please use the final published version (if applicable).

Please check the document version above.

Copyright

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy

Please contact us and provide details if you believe this document breaches copyrights.

We will remove access to the work immediately and investigate your claim.

Green Open Access added to TU Delft Institutional Repository

'You share, we take care!' - Taverne project

<https://www.openaccess.nl/en/you-share-we-take-care>

Otherwise as indicated in the copyright section: the publisher is the copyright holder of this work and the author uses the Dutch legislation to make this work public.

Spatially Scalable Recursive Estimation of Gaussian Process Terrain Maps Using Local Basis Functions

Frida Viset , Rudy Helmons, and Manon Kok 

Abstract—We address the computational challenges of large-scale geospatial mapping with Gaussian process (GP) regression by performing localized computations rather than processing the entire map simultaneously. Traditional approaches to GP regression often involve computational and storage costs that either scale with the number of measurements, or with the spatial extent of the mapped area, limiting their scalability for real-time applications. Our method places a global grid of finite-support basis functions and restricts computations to a local subset of the grid 1) surrounding the measurement when the map is updated, and 2) surrounding the query point when the map is queried. This localized approach ensures that only the relevant area is updated or queried at each timestep, significantly reducing computational complexity while maintaining accuracy. Unlike many existing methods, which suffer from boundary effects or increased computational costs with mapped area, our localized approach avoids discontinuities and ensures that computational costs remain manageable regardless of map size. This approximation to GP mapping provides high accuracy with limited computational budget for the specialized task of performing fast online map updates and fast online queries of large-scale geospatial maps. It is therefore a suitable approximation for use in real-time applications where such properties are desirable, such as real-time simultaneous localization and mapping (SLAM) in large, nonlinear geospatial fields. We show on experimental data with magnetic field measurements that our algorithm is faster and equally accurate compared to existing methods, both for recursive magnetic field mapping and for magnetic field SLAM.

Index Terms—Kalman filters, Gaussian processes, terrain navigation, simultaneous localization and mapping.

I. INTRODUCTION

NAVIGATION in unknown terrains is often performed using on-board sensors measuring the change in position and orientation [1]. However, integrating such measurements

causes unbounded error growth in position and orientation estimates over time [2], [3]. This error growth can be overcome if absolute measurements of the position are included, for example the position relative to a known map. Our focus is on creating such maps using Gaussian process (GP) regression, combining measurements with available prior physical information [4], [5]. Our approach can be used for recursively estimating these maps, as well as for simultaneous localization and mapping (SLAM) [6], [7]. The latter allows for obtaining position estimates and simultaneously estimating a terrain map to correct for the growing position errors. Examples of terrain maps that can be used to limit this error growth are magnetic field maps [8], [9], [10], [11], underwater bathymetry maps [12], [13], or nonlinear terrain fields [14], [15].

Traditional methods for GP regression face computational and storage challenges, limiting their applicability for real-time applications. Many existing methods therefore approximate GPs using basis functions [8], [11], [16], [17]. Basis function approximations to GPs are for instance used for online creation of magnetic field maps [8], underwater bathymetry maps [12] and ground elevation maps [18]. However, for large-scale fields with small-scale variations a large number of basis functions is needed [16], [19]. This results in a large computational complexity even for these approximate methods. To overcome this limitation, our proposed approximation 1) uses a global grid of finite-support basis functions to represent the map and 2) restricts computations to a local subset of the grid around the region of interest at each timestep. The effect of considering only a local subset of the grid for map approximation is illustrated in Fig. 1. The illustration shows that the map represents the nonlinear field with high confidence in the region of interest, while not spending computational resources on accurately representing the entire field. Our method is inherently recursive, making it suitable both for online creation of GP maps as well as for SLAM.

The remainder of the paper is organized as follows: Section II presents related work, while Section III gives an overview of relevant background information. Specifically, it introduces GP regression and basis function approximations to GPs. Section IV gives a description of our method for approximating a large GP scale map. It also shows how the map can be incorporated in an extended Kalman filter (EKF) for SLAM, building on [20]. Section V presents experimental results comparing our proposed approach to other mapping and SLAM algorithms. Section VI gives some concluding remarks and recommendations for future work.

Received 3 January 2024; revised 10 December 2024 and 19 February 2025; accepted 26 February 2025. Date of publication 11 March 2025; date of current version 25 March 2025. This work was supported in part by the Dutch Research Council (NWO) through the project “Sensor Fusion For Indoor localization Using The Magnetic Field” of the research program Veni under Project 18213. The associate editor coordinating the review of this article and approving it for publication was Emanuele Grossi. (Corresponding author: Frida Viset.)

Frida Viset and Manon Kok are with the Department for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: frida.viset@gmail.com; r.j.l.helmons@tudelft.nl).

Rudy Helmons is with the Marine and Transport Technology Department, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: m.kok-1@tudelft.nl).

Digital Object Identifier 10.1109/TSP.2025.3549966

1053-587X © 2025 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See <https://www.ieee.org/publications/rights/index.html> for more information.

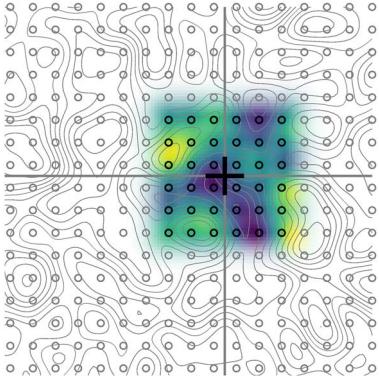


Fig. 1. A locally reconstructed approximation (indicated by the color of the heatmap) of a simulated large, nonlinear geospatial field (indicated with gray level curves) based on a local subset (marked with the black circles) of a global grid of basis functions (marked with the gray circles).

II. RELATED WORK

Our interest lies in online construction of a terrain map using a constant stream of incoming measurements. Out-of-the-box GP regression [4] to construct these maps has a computational complexity that scales with order $\mathcal{O}(N^3)$, where N is the number of measurements. The cost to create the maps hence increases cubically with time. To overcome this limitation, online GP regression typically uses a basis function approximation [8], [12], [18], [20], [21], [22], [23].¹ Using a finite number of m basis functions gives a computational cost at each timestep of $\mathcal{O}(m^2)$. To further mitigate the issue of computational complexity, [27] proposed to use finite-support basis functions in combination with a sparse-weight Kalman filter [28]. This approach reduces the computational complexity of including a new measurement to $\mathcal{O}(m)$. These basis functions are placed in a grid-like pattern over the mapped area. The bigger the area, the more basis functions are required to cover it [16]. This means that as the area increases, the number of basis functions m increases, and the computational complexity $\mathcal{O}(m)$ also increases. However, we still want to be able to create these large maps, because they can aid navigation when moving through the mapped area [29], [30], [31], [32]. Our approach places a grid of basis functions across the entire mapped area, but only updates values associated with a small local subset of basis functions when receiving new measurements, and only uses a small local subset when making predictions, making it faster than state of the art at both online map creation and online querying of the map.

Other approaches that overcome the limitations of computational complexity of basis function approximations e.g. split the map into subdomains [8], [33] or split the measurements into local groups of measurements [34], [35], [36], [37]. These approaches, however, suffer from boundary-effects. Patched local GPs [36] and domain decomposition methods [37] both remedy this issue by introducing constraints connecting the local domains. However, this remedy to the boundary effect

¹Note that the widely used inducing point approximations [24], [25] can be seen as a special case of basis function approximations [26].

problem requires a number of computations that scales with the number of domains and thus does not truly achieve a prediction time complexity independent of the spatial size of the nonlinear field [36]. Other spatially scalable alternatives like KD-tree-based nearest neighbor approaches can include new measurements with a finite computational cost. However, the computational cost for querying the map increases with the number of measurements [11], [30]. In contrast, our approach has a finite computational cost both for including a new measurement, and for querying the map.

Our method is most closely related to a GP approximation called SKI [38]. That method can be used to efficiently include a new measurement in the map using a local subset of basis functions [39], but has a large computational cost for querying the map. While we use exactly the same strategy as SKI to include new measurements, we propose a new and faster approach to query the map, a procedure which is essential to use the terrain map for navigation purposes.

III. BACKGROUND

A. GP Regression

We are interested in estimating a terrain map using GP regression. GP regression allows for estimating a nonlinear function $f : \mathbb{R}^d \rightarrow \mathbb{R}$, distributed according to

$$f \sim \mathcal{GP}(0, \kappa(\cdot, \cdot)), \quad (1)$$

where $\kappa(x, x') : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is some known kernel function, and $\mathcal{GP}(0, \kappa(\cdot, \cdot))$ denotes the GP prior with a mean of 0 and covariance defined by the kernel function [4]. The kernel most commonly used for terrain mapping is the squared exponential kernel defined as

$$\kappa_{\text{SE}}(x, x') = \sigma_{\text{SE}}^2 \exp\left(-\frac{\|x - x'\|_2^2}{2l_{\text{SE}}^2}\right), \quad (2)$$

where $\|\cdot\|_2$ is the Euclidean norm, σ_{SE} is a hyperparameter representing the magnitude of the spatial variations and l_{SE} is a hyperparameter representing the expected lengthscale of the spatial variations. Furthermore, x and x' are two input locations in \mathbb{R}^d [4]. GP regression uses N noisy measurements of the function $y_{1:N} = \{y_t\}_{t=1}^N$ modelled as

$$y_t = f(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, \sigma_y^2), \quad (3)$$

where $x_{1:N} = \{x_t\}_{t=1}^N$ with $x_t \in \mathbb{R}^d$ are known input locations, e_t is measurement noise, σ_y^2 the noise variance, and $\mathcal{N}(0, \sigma_y^2)$ denotes the normal distribution with mean 0 and covariance σ_y^2 . The expected value and variance of the function in any arbitrary location $x^* \in \mathbb{R}^d$ is then given by

$$\mathbb{E}[f(x^*)] = K(x^*, x_{1:N}) (K(x_{1:N}, x_{1:N}) + \sigma_y^2 I_N)^{-1} y_{1:N}, \quad (4a)$$

$$\begin{aligned} \text{Var}[f(x^*)] &= K(x^*, x^*) - K(x^*, x_{1:N}) \\ &\quad \times (K(x_{1:N}, x_{1:N}) + \sigma_y^2 I_N)^{-1} K(x_{1:N}, x^*), \end{aligned} \quad (4b)$$

respectively. Here, the matrix $K(x_{1:N}, x_{1:N})$ is constructed by evaluating the kernel along each possible cross-combination of

the entries in the vector $x_{1:N}$, such that the entry on the i th row and the j th column of $K(x_{1:N}, x_{1:N})$ is $\kappa(x_i, x_j)$. Similarly, the row vector $K(x^*, x_{1:N})$ is defined such that the j th column is $\kappa(x^*, x_j)$. Using the same notation, $K(x^*, x^*)$ is a single-entry matrix with value $\kappa(x^*, x^*)$. In this work we approximate the GP posterior from (4a), (4b). To distinguish the posterior in (4a), (4b) from any approximation of it, we refer to it as the full GP posterior. Computing the full GP posterior has a complexity of $\mathcal{O}(N^3)$, as it requires the inversion of a $N \times N$ matrix.

B. Sparse Approximations to GP Regression With Basis Functions

Sparse approximations to GP regression approximate the function f with a linear combination of m basis functions according to

$$f \approx \Phi^\top w, \quad w \sim \mathcal{N}(0, P), \quad (5)$$

where $w = [w_1, \dots, w_m]^\top$ is a vector of m scalar weights, and $\Phi = [\phi_1, \dots, \phi_m]^\top$ is a vector of m basis functions $\phi_i : \mathbb{R}^d \rightarrow \mathbb{R}$ [24]. The prior covariance on the weights $P \in \mathbb{R}^{m \times m}$ is chosen so that (5) approximates (1) [24].

There exist different methods to approximate predictions using the assumption in (5). A commonly used method is the Deterministic Training Conditional (DTC) approximation [24]. The sparse predictions with DTC are given by

$$\begin{aligned} \mathbb{E}[f(x^*)] &\approx \Phi(x^*)^\top (\Phi(x_{1:N})\Phi(x_{1:N})^\top + \sigma_y^2 P^{-1})^{-1} \\ &\quad \times \Phi(x_{1:N})y_{1:N}, \end{aligned} \quad (6a)$$

$$\begin{aligned} \text{Var}[f(x^*)] &\approx \sigma_y^2 \Phi(x^*)^\top (\Phi(x_{1:N})\Phi(x_{1:N})^\top + \sigma_y^2 P^{-1})^{-1} \\ &\quad \times \Phi(x^*) + K(x^*, x^*) - \Phi(x^*)^\top P\Phi(x^*). \end{aligned} \quad (6b)$$

We refer to the entry on the i th row and the j th column of the matrix $\Phi(x_{1:N})$ as $\phi_i(x_j)$, and to the i th row of the column vector $\Phi(x^*)$ as $\phi_i(x^*)$. The expressions in (6a) and (6b) require $\mathcal{O}(Nm^2 + m^3)$ operations to compute, and $\mathcal{O}(Nm^2)$ storage. This is significantly smaller than the computational cost and storage associated with the full GP predictions (4a), (4b) when the number of basis functions m is much smaller than the number of data points N . The number of basis functions required to accurately approximate the GP scales with the size of the input domain relative to the lengthscale (l_{SE} in (2)) of the kernel [16].

IV. METHOD

In terrain mapping we typically have large areas and small lengthscales. The approach from Section III-B therefore needs many basis functions resulting in a high computational complexity. To remedy this, we instead propose to use a large grid of finite-support basis functions but only use a local subset of these at each timestep. Section IV-A introduces the finite-support basis functions. Section IV-B describes how we include a new measurement.

Section IV-C describes how we use this trained map to make a prediction.

Section IV-D presents how our proposed GP map approximation can be integrated into EKF SLAM with GP maps.

A. Choice of Basis Functions

We choose a set of truncated basis functions $\{\phi_j\}_{j=1}^m$ with centers $u_j \in \mathbb{R}^d$ distributed uniformly on a d -dimensional grid as

$$\phi_j(x) = \begin{cases} \kappa(u_j, x), & \|x - u_j\|_\infty \leq r \\ 0, & \|x - u_j\|_\infty > r \end{cases}, \quad (7)$$

where $\|\cdot\|_\infty$ denotes the sup-norm and r the truncation limit. Since the function $\kappa_{\text{SE}}(u_j, x)$ tends to zero as $\|x - u_j\|_\infty \rightarrow \infty$, this truncation means that values which are close to zero are approximated as exactly zero. This approximation has a low impact on the accuracy when r is chosen large relative to the lengthscale l_{SE} from (2).

The finite support of our basis functions from (7) ensures that there is always only a finite number of basis functions with overlapping support in any given location. We denote this number of basis functions by m' . This results in sparsity in the matrices requiring inversion in (6a) and (6b).

B. Constructing the Map

The GP approximation in terms of basis functions (see (5)) is a parametric model that is linear in the weights w . Hence, the posterior of these weights can be found using stochastic least squares. We solve this stochastic least squares problem recursively using an information filter without any dynamics. Specifically, obtaining the posterior on information form corresponds to computing the terms $\Phi(x_{1:N})y_{1:N}$ and $\Phi(x_{1:N})\Phi(x_{1:N})^\top$ in (6a) and (6b) recursively [40]. In the remainder of this paper, we let the information vector at time t be defined as $\iota_{1:t} = \Phi(x_{1:t})y_{1:t}$, and the information matrix at time t be defined as $\mathcal{I}_{1:t} = \Phi(x_{1:t})\Phi(x_{1:t})^\top$. The information vector and the information matrix encode the aggregated information from all the available measurements up until and including time t . We update the information vector $\iota_{1:t}$ and the information matrix $\mathcal{I}_{1:t}$ as [40]

$$\iota_{1:t} = \iota_{1:t-1} + \Phi(x_t)y_t, \quad (8a)$$

$$\mathcal{I}_{1:t} = \mathcal{I}_{1:t-1} + \Phi(x_t)\Phi(x_t)^\top. \quad (8b)$$

This only requires updating a finite amount of elements $m' \ll m$ in each update step due to the finite support of our basis functions (7). Specifically, since only m' basis functions have overlapping support in any given location, the terms $\Phi(x_t)y_t$ and $\Phi(x_t)\Phi(x_t)^\top$ only contain m' non-zero elements. Which elements of the information vector and matrices need to be updated can be identified by defining the subset \mathcal{S} of the basis functions from (7) that are non-zero in x_t as

$$\mathcal{S}(x_t, r) = \{j \mid \|x_t - u_j\|_\infty \leq r\}. \quad (9)$$

The updates (8a), (8b) only need to be applied to entries $j \in \mathcal{S}(x_t, r)$ for the information vector, and $j, j' \in \mathcal{S}(x_t, r) \times \mathcal{S}(x_t, r)$ for the information matrix. The contribution from the term $\phi_j(x_t)y_t$ and $\phi_j(x_t)\phi_{j'}(x_t)$ will inherently be zero for all other entries.

C. Querying the Map - Prediction of Terrain Values in New Locations

To use a map constructed using the method from Section IV-B for navigation purposes, we need to be able to predict terrain value at new locations. In other words, we need to evaluate the GP posterior mean $E[f(x^*)]$ and variance $\text{Var}[f(x^*)]$ at that point.

In principle, we could straightforwardly use the full information vector $\iota_{1:N}$ and information matrix $\mathcal{I}_{1:N}$ in (6a), (6b). However, this requires inversion of the information matrix, an $O(m^3)$ operation. To optimize computation, we instead utilize a smaller, local subset of basis functions, denoted as S^* , that are near the prediction point. This subset S^* consists of all basis functions within a distance r^* from the prediction point x^* , as illustrated in Fig. 1. Note that the entries of the information vector and matrix can be expressed as

$$\mathcal{I}_{1:N}^{i,j} = \sum_{t=1}^N \phi_i(x_t) \phi_j(x_t), \quad \iota_{1:N}^i = \sum_{t=1}^N \phi_i(x_t) y_t, \quad (10)$$

where $i \in 1, \dots, m$ and $j \in 1, \dots, m$ are indices corresponding to each entry of the information matrix. Hence, using this property, we define the local information vector $\iota_{1:N}^*$ and matrix $\mathcal{I}_{1:N}^*$ as

$$\iota_{1:N}^* = \Phi_{S^*}(\mathbf{x}_{1:N}) \mathbf{y}_{1:N}, \quad (11a)$$

$$\mathcal{I}_{1:N}^* = \Phi_{S^*}(\mathbf{x}_{1:N}) \Phi_{S^*}(\mathbf{x}_{1:N})^\top, \quad (11b)$$

where $\Phi_{S^*}(\mathbf{x})$ represent the evaluations of the basis functions in the local subset S^* at any location \mathbf{x} . We now use $\iota_{1:N}^*$, $\mathcal{I}_{1:N}^*$ to approximate the predicted mean and variance as

$$E[f(x^*)] \approx \Phi_{S^*}(x^*)^\top (\mathcal{I}_{1:N}^* + \sigma_y^2 P^*)^{-1} \iota_{S^*}, \quad (12a)$$

$$\begin{aligned} \text{Var}[f(x^*)] &\approx \sigma_y^2 \Phi_{S^*}(x^*)^\top (\mathcal{I}_{1:N}^* + \sigma_y^2 P^*)^{-1} \Phi_{S^*}(x^*) \\ &\quad + K(x^*, x^*) - \Phi_{S^*}(x^*)^\top P^* \Phi_{S^*}(x^*). \end{aligned} \quad (12b)$$

Here, P^* is the prior covariance of the basis function weights, set to recover the prior accurately at the centers of the selected basis functions. The detailed derivation of P^* is provided in Appendix A. Note that the maximum size of the local subset S^* is $m'' \leq (\frac{2r^*}{l_u} + 1)^d$, where d is the dimension of the input vector. The computational complexity for computing (12) is therefore $O(m''^3)$.

D. Integration of Mapping Algorithm Into an EKF for Magnetic Field SLAM

Our approach allows for constructing a map (see Section IV-B) and for querying the map to use it for navigation (see Section IV-C). In this section we combine both, and show that our approach can also be used for SLAM. We illustrate this for magnetic field SLAM using an extended Kalman filter (EKF), which we refer to as EKF Mag-SLAM. We specifically modify the algorithm presented in [20], but use our proposed map approximation instead of the Hilbert space (HS) basis functions used in [20].

Our task for SLAM is to estimate the joint posterior distribution:

$$p(p_t, q_t, w_t | y_{1:t}),$$

where p_t is a three-dimensional position vector, q_t is a 4-dimensional unit quaternion representing the orientation, and w_t is an m -dimensional vector of weights associated with finite-support basis functions $\phi_i(p)$ representing the magnetic field. This posterior is approximated in the EKF framework using a Gaussian distribution. To deal with the fact that orientations are non-Euclidean, we implement an error-state EKF. This implies that the EKF not only consists of a dynamic update and a measurement update, but also a re-linearization step. In line with Sections IV-B, IV-C, we implement this EKF on information form. This means that instead of keeping track of the state estimates $\hat{p}_{t|t}$, $\hat{q}_{t|t}$, $\hat{m}_{t|t}$ and their corresponding covariances, we keep track of the corresponding information matrix $\mathcal{I}_{t|t}^{\text{EKF}}$ and information vector $\iota_{t|t}^{\text{EKF}}$.

1) State-Space Model: Let us assume that information about the change in position and orientation between two time steps is available, e.g. from wheel encoders or inertial sensors. Furthermore, let us assume that the magnetic field map is constant over time. The dynamic model of our states is then

$$p_{t+1} = p_t + \Delta p_t + e_{p,t}, \quad e_{p,t} \sim \mathcal{N}(0, \sigma_p^2 \mathcal{I}_3), \quad (13a)$$

$$q_{t+1} = q_t \odot \Delta q_t \odot \exp_q(e_{q,t}), \quad e_{q,t} \sim \mathcal{N}(0, \sigma_q^2 \mathcal{I}_3), \quad (13b)$$

$$w_{t+1} = w_t. \quad (13c)$$

Here, \odot is the quaternion product and \exp_q is the operator that maps an axis-angle orientation deviation to a quaternion (see [3] for details on quaternion algebra).

Our map of the magnetic field, expressed in a world-fixed frame, assumes that the field is curl-free. It can therefore be expressed as the gradient of a scalar potential $\varphi(p_t)$ with respect to the position p_t [41]. We model the scalar potential $\varphi : \mathbb{R}^3 \rightarrow \mathbb{R}$ as a GP with prior

$$\varphi \sim \mathcal{N}(0, \kappa_{\text{SE}}(\cdot, \cdot) + \kappa_{\text{lin}}(\cdot, \cdot)), \quad (14)$$

where $\kappa_{\text{lin}}(\cdot, \cdot)$ is a linear kernel [4] used to model the static Earth's magnetic field. The squared exponential kernel is used to model the local magnetic field anomalies.

We assume that measurements $y_t \in \mathbb{R}^3$ of the magnetic field expressed in a sensor-fixed frame are available. Approximating the scalar potential field in terms of the finite-support basis functions from Section IV-A results in the measurement model

$$y_t = R_t \nabla_p \Phi(p_t)^\top w_t + e_{m,t}, \quad e_{m,t} \sim \mathcal{N}(0, \sigma_m^2 \mathcal{I}_3). \quad (15)$$

Here, $e_{m,t}$ denotes the zero-mean measurement noise with covariance $\sigma_m^2 \mathcal{I}_3$, and R_t the rotation from the world-fixed to the sensor-fixed frame.

In line with existing SLAM literature, we use choose the prior on the position and orientation to be zero-mean with a very small covariance. The GP prior results in a prior mean of zero for the states w_t with a covariance based on the GP prior [20].

2) *Dynamic Update*: The dynamic update on information form [42] is given by

$$\mathcal{I}_{t+1|t}^{\text{EKF}} = (\mathcal{I}_{t|t}^{\text{EKF}} + Q^{-1})^{-1}, \quad (16)$$

where Q is the process noise covariance for the model (13). Since we assume the map to be static, the matrix Q can be factorized according to

$$\begin{bmatrix} \tilde{Q} & 0 \\ 0 & 0 \end{bmatrix} = V^T Q V, \quad V = [I \ 0], \quad (17)$$

where \tilde{Q} is a 6×6 matrix representing the process noise on the position and orientation.² After applying the matrix inversion lemma to (16), it reduces to

$$\mathcal{I}_{t+1|t}^{\text{EKF}} = \mathcal{I}_{t|t}^{\text{EKF}} - \mathcal{I}_{t|t}^{\text{EKF}} V^T (V \mathcal{I}_{t|t}^{\text{EKF}} V^T + \tilde{Q}^{-1})^{-1} V \mathcal{I}_{t|t}^{\text{EKF}}. \quad (18)$$

3) *Measurement Update*: The Kalman filter measurement update EKF Mag-SLAM is given by [42]

$$\iota_{t|t}^{\text{EKF}} = \frac{1}{\sigma_m^2} H_t y_t, \quad (19a)$$

$$\mathcal{I}_{t|t}^{\text{EKF}} = \mathcal{I}_{t|t-1}^{\text{EKF}} + \frac{1}{\sigma_m^2} H_t H_t^T, \quad (19b)$$

where the H_t is the Jacobian of the measurement model (15). Note the absence of a summation in (19a) due to the error-state implementation of the EKF. The update of the information matrix in (19b) is inherently sparse due to our use of finite-support basis functions. This is similar to the sparsity in (8b), and has a complexity of $\mathcal{O}(m'')$.

4) *Re-Linearization*: The re-linearization of the estimated position, orientation and magnetic field of our error-state EKF requires evaluation of the error-state. For this, an inversion of the information matrix is essential. We instead perform an approximate re-linearization by only using the basis functions that are close to the prediction point (which in this case is the estimated location). This allows us to execute the re-linearization at a computational cost of $\mathcal{O}(m''^3)$ as

$$[\delta_t^\top \ \eta_t^\top \ \nu_{*,t}^\top]^\top = (\mathcal{I}_{t,*}^{\text{EKF}})^{-1} \iota_{t,*}^{\text{EKF}}, \quad (20a)$$

$$\hat{p}_{t|t} = \hat{p}_{t|t-1} + \delta_t, \quad (20b)$$

$$\hat{q}_{t|t} = \hat{q}_{t|t-1} \odot \exp_q(\eta_t), \quad (20c)$$

$$\hat{w}_{*,t|t} = \hat{w}_{*,t|t-1} + \nu_{*,t}. \quad (20d)$$

Here, δ_t , η_t are the error states for the position and orientation, respectively. Furthermore, $\nu_{*,t}$ is the error state corresponding to the weights of the basis functions in the subset S_* , see also (9) and Section IV-C. In other words, we only correct the linearization point of a local subset of the magnetic field map.

5) *Note on Computational Complexity*: As shown above, the computational complexity of the measurement update of our EKF Mag-SLAM algorithm is $\mathcal{O}(m'')$ and of the relinearization is $\mathcal{O}(m''^3)$. In principle, the dynamic update would have a computational complexity of $\mathcal{O}(m^2)$. However, there are large

²Note that the error state representing the orientation is three-dimensional.

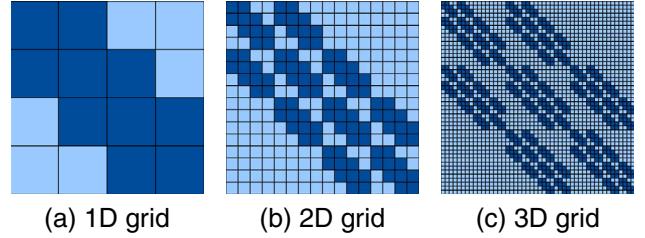


Fig. 2. Sparsity patterns illustrating which entries i, j in the information matrix correspond to pairs of basis function locations x_i, x_j that are closer than $2r^*$ according to the infinity norm (dark blue) and which are not (light blue). The patterns arise from the ordering of the indexes of the basis functions, relative to their locations along each of the three dimensions.

portions of the matrix $\mathcal{I}_t^{\text{EKF}}$ that never need to be explicitly computed. The only entries of $\mathcal{I}_t^{\text{EKF}}$ that need to be computed, are entries that affect the output of the relinearization, the measurement update, or the dynamic update.

Firstly, we will describe the set of entries in $\mathcal{I}_{t,*}^{\text{EKF}}$ that affect the output of the relinearization and the measurement update. There are many basis function pairs i, j that are so far away from each other that they are never used at the same time in the relinearization step, nor in the measurement update. We denote the set of all pairs of basis functions i, j that are *close enough* to each other that we *do need* to compute the corresponding entries in $\mathcal{I}_{t,*}^{\text{EKF}}$ by $S_{\forall*} = \{i, j \mid \|p_{i,t} - p_{j,t}\|_\infty \leq 2r^*\}$. This definition includes all index pairs i, j where the corresponding basis functions are closer to each other according to the infinity norm than $2r^*$. The set $S_{\forall*}$ contains $\mathcal{O}(mm'^2)$ elements. Assuming $m \gg m''$, we simplify this notation, and say that $S_{\forall*}$ contains $\mathcal{O}(m)$ elements. Fig. 2 illustrates examples of where the entries i, j in $S_{\forall*}$ are located in an information matrix corresponding to a one-dimensional, a two-dimensional, and a three-dimensional grid of equispaced basis functions. In the one-dimensional grid, four basis functions are located along the x-axis at $[-1.5, -0.5, 0.5, 1.5]$, and the indices are ordered correspondingly in Fig. 2. In the two-dimensional grid, 16 basis functions are located along the x and y-axis at

$$\begin{bmatrix} -1.5 & -1.5 & -1.5 & \dots & 1.5 & 1.5 \\ -1.5 & -0.5 & 0.5 & \dots & 0.5 & 1.5 \end{bmatrix}, \quad (21)$$

and also indexed chronologically. The sparsity pattern shows a fractal-like structure that is explained by the fact that we are displaying the pattern for a 3D grid of basis functions, and the nature of the ordering of the indexes of these basis functions. In all cases, only $\mathcal{O}(m)$ elements are part of the set $S_{\forall*}$, meaning that only $\mathcal{O}(m)$ elements need to be computed in the information matrix to carry out the relinearization and the measurement update.

Secondly, we describe the set of entries in $\mathcal{I}_{t,*}^{\text{EKF}}$ that affect the output of the dynamic update. The dynamic update would, in general, require knowledge of the entire information matrix. In our case, the only entries of the information matrix that are used to compute the term $\mathcal{I}^{\text{EKF}} V^T (V \mathcal{I}^{\text{EKF}} V^T + \tilde{Q}^{-1})^{-1} V \mathcal{I}^{\text{EKF}}$ are the entries in the first 6 rows and the first 6 columns of the

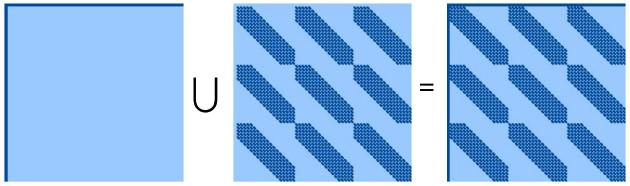


Fig. 3. Sparsity pattern illustration of the information matrix for the full state consisting of the position, orientation and magnetic field. The dark blue entries indicate entries which are necessary to compute with our assumptions, and the light blue entries indicate values that are not necessary to compute. The first sparsity pattern indicate the values we need associated with the position and orientation (the first 6 states in the full state-space). The second sparsity pattern has a dark blue color in the entries in the set $S_{\forall*}$, of all entries that can possibly be necessary to make a map prediction in any location. The last sparsity pattern is the union of these two sets.

TABLE I
COMPARISON OF COMPUTATIONAL COMPLEXITIES OF
EKF MAG-SLAM

Step	EKF Mag-SLAM with our method	EKF Mag-SLAM with Hilbert Space functions
Meas update	$\mathcal{O}(m'^2)$	$\mathcal{O}(m^2)$
Prediction	$\mathcal{O}(m''^3)$	$\mathcal{O}(m)$
Dyn update	$\mathcal{O}(m)$	$\mathcal{O}(m^2)$

information matrix. The only entries of the information matrix we need to keep track to ensure the output of the dynamic update is correct of are therefore the union of the first 6 columns and the first 6 rows and the values in $S_{\forall*}$. Fig. 3 shows an illustration of the set of indices necessary to perform the dynamic update (the leftmost matrix), the indices necessary to perform the measurement update and the relinearization (the middle matrix) and the union of these two sets (the rightmost matrix). The union of the sets of indices are all possible indices necessary to get a correct output from the measurement update, the relinearization and the dynamic update. In other words, the union of these sets of indices are the only entries of the information matrix I_t^{EKF} we need to compute to for our EKF Mag-SLAM algorithm.

In conclusion, an overview of the computational complexities of EKF Mag-SLAM with our mapping technique compared to EKF Mag-SLAM with Hilbert Space basis functions from [20] is given in Table I. When m' and m'' are chosen to be considerably smaller than m , our proposed algorithm will be faster than the approach from [20]. This is a reasonable choice when the terrain is large relative to the spatial variations we wish to map.

V. RESULTS

In this section, we first compare the performance of our method with existing approaches on two low-dimensional benchmark data sets. We also study how long it takes for our method to make online predictions using a short length scale and millions of basis functions on a bathymetry dataset that is too large for existing approaches given our hardware constraints.

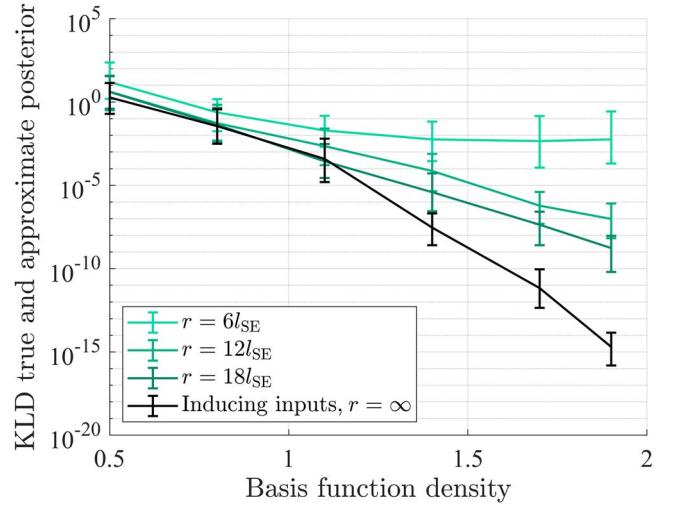


Fig. 4. KL divergence (KLD) between the full GP posterior, and approximations with various local domain sizes r , trained on the audio dataset. The error bars indicate the average deviation above and below the mean, respectively, after 100 repeated experiments with 100 randomly sampled measurements from the training set.

As the data sets considered in Sections V-B and V-C are geospatial data with a non-zero mean value, the average of the output is subtracted before training. This average is subsequently added to each prediction. All computation times reported are measured while running on a Dell XPS 15 9560 laptop, with 16 GB RAM and an Intel Core i7-7700HQ CPU running at 2.80 GHz. In all experiments we set $r = 2r^*$, as picking $r \geq 2r^*$ gives that the expression for P_{\star}^{-1} reduces to $P_{\star}^{-1} = K(u_{S^*}, u_{S^*})$, as derived in Appendix A. This means that we have a closed-form expression for the inverse of the prior covariance which does not rely upon computing any numerical inverses, further reducing the necessary computational efforts to make each prediction.

A. One-Dimensional Sound Map

In this section we will construct a one-dimensional GP map of the amplitude of a sound wave using the data from [43]. We treat time as the spatial axis of this map. The same data and approach was used in [38] and [39] to demonstrate the ability of their approximations to learn a GP model from a large dataset with a large input domain relative to the size of the spatial variations in the field. The dataset contains a training set of 59 309 measurements of sound amplitude collected at a known input time and a fixed test set of 691 points. We compare the accuracy and required computation time of our method with existing work. To this end, similar to [39] we use a squared exponential kernel (2) with hyperparameters $\sigma_{\text{SE}} = 0.009$, $l_{\text{SE}} = 10.895$, $\sigma_y = 0.002$.

In Fig. 4 we investigate how large the size of the local domain r (see Section IV-A) has to be for our approach to accurately approximate the posterior. We assess our results in terms of the KL divergence between the approximate posterior and the full GP posterior. The KL divergence is a measure that

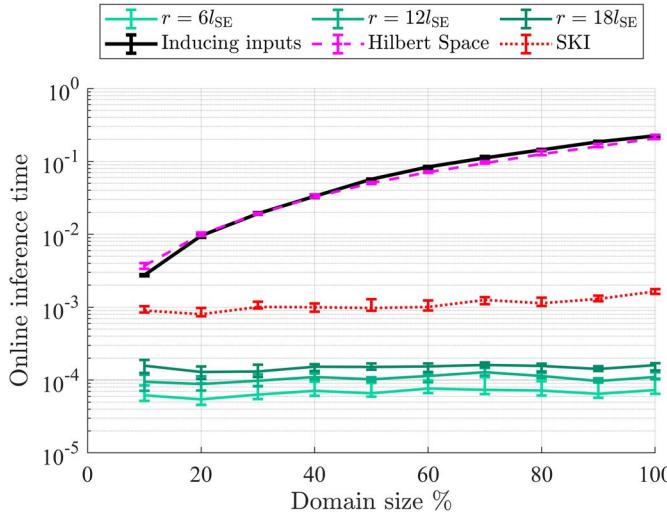


Fig. 5. Online inference time of the sound map for a growing domain size. We compare our proposed method (with various local domain sizes r) to the inducing input approximation using inducing inputs on a grid, the Hilbert space basis function approximation, and SKI. All methods were run using the same amount of basis functions. The error bars indicate the average deviation above and below the mean after 100 repeated experiments, respectively.

compares how similar distributions are. It can attain values between 0 and ∞ , where a lower value means that the distributions are more similar. First of all, it can be seen that a higher basis functions density (measured in the number of basis functions per lengthscale) results in a smaller KL divergence, i.e. a better approximation, of the inducing input approximation. Our method can be seen to approach the inducing input approximation for larger r .

To measure how long it takes for our approach to perform online mapping, we measure the time it takes to include one additional measurement and perform one prediction step. In Fig. 5, we compare our proposed method to the inducing input solution implemented with an online Kalman filter [23], an online Kalman filter implementation with Hilbert space basis functions [16], and an online implementation of SKI [39], for increasing domain sizes. As the domain size increases, we keep the basis function density constant to retain the same approximation accuracy. The amount of basis functions m is therefore increasing linearly. As the online inference time of SKI is affected by how many iterations are used in the conjugate gradient solver to improve the approximation accuracy, we measure the run-time using just one iteration in the solver. To compare the prediction accuracy of SKI to our approach in Table II, however, we use the exact solution that the solver is approximating. We can therefore conclude that an online evaluation of our algorithm is faster than competing approaches while being able to recover the same or better SMAE (standardized mean average error) and MSLL (mean squared log loss) scores (see Table II). The online computational complexity of both Hilbert space basis functions and inducing inputs increases quadratically as the domain size increases. The computational complexity of our approach is bounded by $O(m'^3)$ independent of the increase in the domain size, which is why our online computational

complexity remains lower than 10^{-3} seconds independent of the growth of the domain size.

B. Mapping Daily Precipitation Levels

In this section, we compare the prediction accuracy and computation time of our mapping approach to alternative methods on a large geo-spatial dataset which is used as a benchmark dataset for evaluation accuracy and computation time by [39], and [38]. The dataset contains 528 474 measurements of daily precipitations from the US in the training set, and 100 000 measurements in the test set [39]. The input dimensions are latitude, longitude, and time. We use the squared exponential kernel using the same hyperparameters as [39]. These hyperparameters are $\sigma_{SE} = 3.99$, $\sigma_y = 2.789$ and $l_{SE} = [3.094, 2.030, 0.189]$, where the three lengthscales l_{SE} apply to each of the three dimensions, respectively.

In Table III, the standardized mean squared error (SMSE) of our approach with $r^* = 3.5l_{SE}$ is compared to the full GP prediction, and to the inducing input approximation with basis functions placed on the same grid. The SMSE of our approach almost matches the inducing input approximation with the same number of basis functions. Using 200 000 basis functions, it matches the GP prediction accuracy with the same number of measurements. Using all the measurements and 200 000 basis functions gives the highest prediction accuracy, which is a combination that is computationally infeasible given our hardware constraints for both the full GP regression and the inducing input approximation to give a prediction. However, our proposed method has an online training time of only 0.017 seconds per measurement and 0.016 seconds per prediction.

C. Global Bathymetry Field Mapping

To investigate the time required for our method to include a new measurement and make a prediction in a large geospatial field with fine-scale variations, we run it on a dataset containing values of the height difference with respect to sea level across the globe [44]. The input domain of this data is huge compared to the scale of the spatial variations, and the data is therefore challenging to train on using state-of-the-art methods. We retrieve 37.5 million depth values from the database, and we consider the latitude and longitude as input locations.

We test our approach using the squared exponential kernel from (2). We choose σ_{SE}^2 equal to the variance of the 37.5 million measurements and $\sigma_y = 0.1\sigma_{SE}$ to ensure a reasonable signal-to-noise ratio. Furthermore, we set the lengthscale $l_{SE} = 0.16$ degrees, which corresponds to 18.3 km on the equator. Note that this lengthscale can straightforwardly be changed to a more physically informed value by an end-user.

The result in Fig. 6 shows the bathymetry map learned using 10% of the measurements and 2.33 million (M) basis functions (Fig. 6(a)), the map learned using 100% of the measurements and 268 thousand (K) basis functions (Fig. 6(b)), and map learned using 100% of the measurements and a dense grid of 2.33 million basis functions (Fig. 6(c)). We use a local subset contained within $r^* = 3l_{SE}$ to approximate the GP posterior mean. For the results in Fig. 6(c), including each new

TABLE II

STANDARDISED MEAN ABSOLUTE ERRORS (SMAEs), COMBINED TIME TO INCLUDE A NEW MEASUREMENT AND MAKE A NEW PREDICTION, MEAN STANDARDIZED LOG-LIKELIHOOD (MSLL) SCORES FOR CONSTRUCTING THE SOUND MAP. THE RESULTS THAT ATTAIN THE LOWEST RUN-TIME WHILE ALSO ATTAINING THE LOWEST SMAE ARE HIGHLIGHTED. SMAEs AND MSLL SCORES ARE EVALUATED ON ALL STANDARDIZED TEST POINTS INSIDE THE CONSIDERED DOMAIN, AND THE AVERAGE TIME PLUS MINUS ONE STANDARD DEVIATION IS CALCULATED BASED ON 100 REPETITIONS

	10% of Domain, $m = 800$			100% of Domain, $m = 8000$		
	SMAE	Time[s]	MSLL	SMAE	Time[s]	MSLL
$r = 6l_{SE}$	0.42	$6.2 \cdot 10^{-5} \pm 2.1 \cdot 10^{-5}$	37.5	0.34	$7.3 \cdot 10^{-5} \pm 2.2 \cdot 10^{-5}$	48.0
$r = 12l_{SE}$	0.22	$9.4 \cdot 10^{-5} \pm 2.1 \cdot 10^{-5}$	4.18	0.20	$1.1 \cdot 10^{-4} \pm 1.6 \cdot 10^{-5}$	12.2
$r = 18l_{SE}$	0.22	$1.3 \cdot 10^{-4} \pm 2.7 \cdot 10^{-5}$	3.94	0.20	$1.6 \cdot 10^{-4} \pm 2.1 \cdot 10^{-5}$	12.2
SKI	0.22	$8.0 \cdot 10^{-4} \pm 1.2 \cdot 10^{-4}$	5.74	0.20	$1.6 \cdot 10^{-3} \pm 1.7 \cdot 10^{-4}$	18.2
Inducing inputs	0.22	$1.0 \cdot 10^{-2} \pm 7.4 \cdot 10^{-4}$	3.81	0.20	$2.1 \cdot 10^{-1} \pm 1.6 \cdot 10^{-3}$	12.2
Hilbert space	0.22	$9.5 \cdot 10^{-3} \pm 5.1 \cdot 10^{-4}$	3.80	0.20	$2.2 \cdot 10^{-1} \pm 4.6 \cdot 10^{-3}$	12.2

TABLE III

SMSE ACCURACIES OF DAILY PRECIPITATION LEVEL PREDICTIONS. THE PREDICTIONS ARE OBTAINED WITH A LOCAL DOMAIN WITH SIZE $r^* = 3l_{SE}$, WHICH CORRESPONDS TO USING AT MOST 144 BASIS FUNCTIONS IN EACH LOCAL PREDICTION FOR THE HIGHEST $m = 200K$

N	full GP	Inducing Inputs		Local Information Filter			
		m=10K	m=20K	m=10 K	m=20K	m=100K	m=200K
10 000	0.823	0.957	0.905	0.957	0.906	0.824	0.823
20 000	0.766	0.946	0.861	0.947	0.862	0.770	0.766
100 000	N/A	0.907	0.782	0.907	0.786	0.561	0.545
528 474	N/A	0.894	0.746	0.895	0.751	0.468	0.435

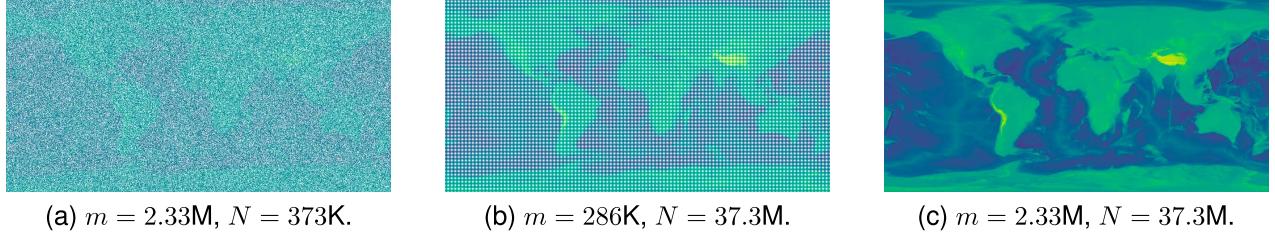


Fig. 6. Bathymetry dataset reconstruction with GP regression. The color corresponds to the posterior predicted elevation of the earth surface both above and below the sea, and the opacity is inversely proportional to the variance of the approximate GP prediction in each location.

measurement takes $3.7 \times 10^{-4} \pm 0.12 \times 10^{-4}$ seconds. Each prediction takes 0.0097 ± 0.017 seconds. These results demonstrate that our proposed approach can remain computationally feasible in cases where the measurement density is high, and the map area is large relative to the length scale of the spatial variations. The disadvantage of discarding measurements is visible in Fig. 6(a). When fewer measurements are included, less information is available about the map, causing the image to appear blurry. The disadvantage of discarding basis functions is visible in Fig. 6(b). When there are not enough basis functions, the map is not represented at a high enough resolution, causing the image to appear grid-like.

D. Using Local Information Filter for Faster Mag-SLAM

To experimentally compare our EKF Mag-SLAM approach from Section IV-D with that from [20], we apply both

algorithms to a dataset from a foot-mounted sensor that was collected by [45] and subsequently used in [20] to demonstrate the performance of the method. We measure the average computation time required for each iteration of our algorithm and compare this with the average computation time required, on the same laptop, for each iteration of the algorithm from [20].

In Fig. 7(a), the odometry obtained by using only accelerometer and gyroscope measurements from a foot-mounted sensor using the algorithm in [46] is displayed. In Fig. 7(e), the estimated trajectory using the EKF with Hilbert space basis functions as in [20] is displayed. All estimates are overlaid on the floorplan of the building where the measurements were collected. This gives a rudimentary means of evaluating the position estimation accuracy, since the subject walked through the same hallways in a repeated pattern 8-motion. The drift in the odometry in Fig. 7(a) therefore shows up as a slow displacement of the position estimate away from the hallway. In

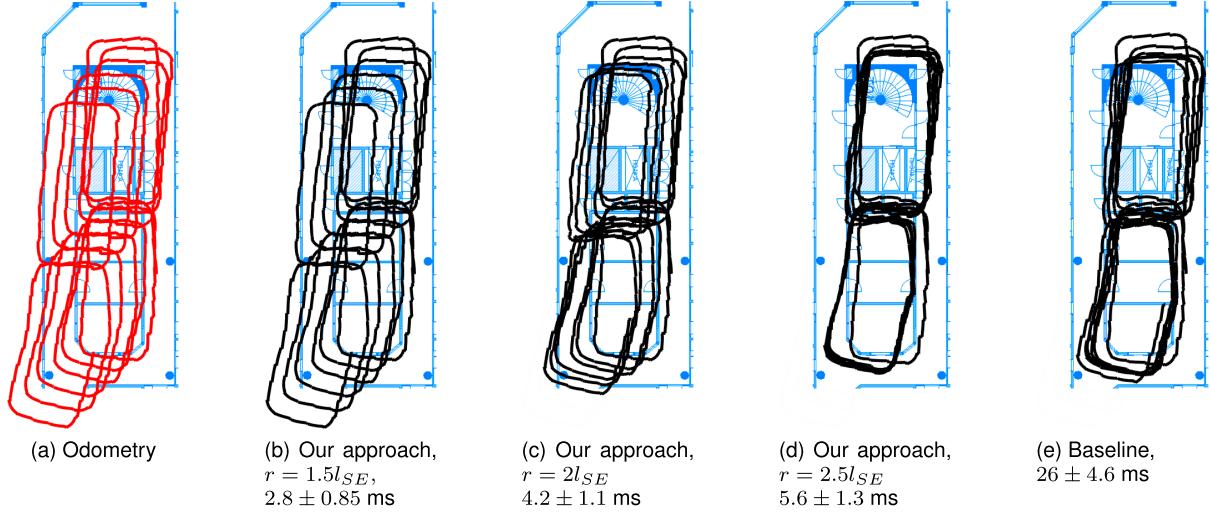


Fig. 7. (a) Trajectory estimates for indoor pedestrian walking laps in a hallway using only foot-mounted odometry, (b)–(d) EKF Mag-SLAM with various sizes of the local domain determined by r , and (e) the baseline algorithm that uses with Hilbert space basis functions. The average computation time in milliseconds for one iteration of each filter (dynamic update + measurement update) is written below each subfigure.

contrast, the estimates obtained using our approximate mapping (see Fig. 7(d)) and using the algorithm from [20] (see Fig. 7(e)) compensate for this drift. The notable difference between our approach and the approach from [20], is that the latter uses Hilbert space basis functions resulting in a time of 26 ± 4.6 ms to run at each iteration, while our equally accurate algorithm requires only 5.6 ± 1.3 ms to run at each iteration. Although comparing the computational time of different algorithms depends heavily on implementation, we used the same implementation as [20], and wrote the implementation of our algorithm in the same language and on the same format, simply swapping the terms used by [20] with ours.

VI. CONCLUSION

To improve position estimation online with few computational resources, we have presented an efficient online mapping technique that approximates the GP posterior. The required number of computations neither scales with the number of measurements, nor with the spatial extent of the map. The storage requirements of our presented mapping algorithm scale linearly with the spatial extent of the map, and also does not scale with the number of measurements. We have also demonstrated the ability of our mapping algorithm to match the accuracy of previously proposed approximations on benchmark datasets using a lower computation time. Our mapping algorithm can also be used for SLAM. We have shown experimentally that our proposed method achieves the same prediction accuracy for magnetic field SLAM using a foot-mounted sensor as in previous work, while requiring a shorter computation time. Future work could investigate the existence of theoretical bounds on the approximation errors, or investigate ways to incorporate the mapping technique for SLAM in different geospatial fields, and for other applications such as navigation of robots or vehicles.

APPENDIX A PRIOR COVARIANCE FOR THE PREDICTION-POINT DEPENDENT BASIS FUNCTION APPROXIMATION

The prior for the parametric approximation to the GP regression using the local subset of basis functions is given by

$$\tilde{f}^* = \Phi_{S^*}^\top w_{S^*}, \quad w_{S^*} \sim \mathcal{N}(0, P_*) \quad (22)$$

where P_* is the prior covariance on the local weights. The condition that the prior should be recovered in the center locations of the basis functions is given as

$$p(\tilde{f}^*(u_{S^*})) = p(f(u_{S^*})) \quad (23)$$

where u_{S^*} denotes the center locations of the basis functions contained in the set S^* . As both distributions in (23) are normal distributions with mean 0, this condition holds if and only if the two covariances are equal according to

$$\Phi_{S^*}(u_{S^*})^\top P_* \Phi_{S^*}(u_{S^*}) = K(u_{S^*}, u_{S^*}) \quad (24)$$

This results in the closed-form expression for P_*

$$P_* = (\Phi_{S^*}(u_{S^*})^\top)^{-1} K(u_{S^*}, u_{S^*}) \Phi_{S^*}(u_{S^*})^{-1} \quad (25)$$

when $(\Phi_{S^*}(u_{S^*})^\top)$ and $\Phi_{S^*}(u_{S^*})$ are invertible. The entry in row i and column j of the matrix $\Phi_{S^*}(u_{S^*})$ is $\phi_i(u_j)$, which is defined using (7) as

$$\phi_i(u_j) = \begin{cases} \kappa(u_i, u_j), & \|u_i - u_j\|_\infty \leq r \\ 0, & \|u_i - u_j\|_\infty > r \end{cases} \quad (26)$$

For all inducing input pairs i, j in the set $\mathcal{S}(x^*, r^*)$, it holds that $\|u_i - u_j\|_\infty \leq 2r^*$. If $r \geq 2r^*$, the condition $\|u_i - u_j\|_\infty \leq r$ holds for all $i, j \in \mathcal{S}^*$. This implies that $\phi_i(u_j) = \kappa(u_i, u_j)$ for all $i, j \in \mathcal{S}^*$ and hence $\Phi_{S^*}(u_{S^*}) = K(u_{S^*}, u_{S^*})$. Inserting this result into (24) gives $P_* = K(u_{S^*}, u_{S^*})^{-1}$.

DATA AVAILABILITY STATEMENT

All code and files required to reproduce the results can be found on <https://github.com/fridaviset/fastgpmapping>.

REFERENCES

- [1] F. Gustafsson, *Statistical Sensor Fusion*. Studentlitteratur, 2013, p. 554.
- [2] O. J. Woodman, “An Introduction to Inertial Navigation,” 2007. [Online]. Available: <https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-696.pdf>
- [3] M. Kok, J. Hol, and T. B. Schön, “Using Inertial Sensors for Position and Orientation Estimation,” *Foundations Trends Signal Process.*, Jan. 2017.
- [4] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, Nov. 2005.
- [5] N. Wahlström, “Modeling of magnetic fields and extended object for localization applications,” Ph.D. Thesis, Linköping University, Dec. 2015.
- [6] T. Bailey and H. Durrant-Whyte, “Simultaneous localization and mapping (SLAM): Part II,” *IEEE Robot. Automat. Mag.*, vol. 13, no. 3, pp. 108–117, Mar. 2006.
- [7] H. Durrant-Whyte and T. Bailey, “Simultaneous localization and mapping: Part I,” *IEEE Robot. Automat. Mag.*, vol. 13, no. 2, pp. 99–110, Feb. 2006.
- [8] M. Kok and A. Solin, “Scalable magnetic field SLAM in 3D using Gaussian process maps,” in *Proc. 21st Int. Conf. Inf. Fusion (FUSION)*, Jul. 2018, pp. 1353–1360.
- [9] P. Robertson et al., “Simultaneous localization and mapping for pedestrians using distortions of the local magnetic field intensity in large indoor environments,” in *Proc. Int. Conf. Indoor Positioning Indoor Navigation*, Oct. 2013, pp. 1–10.
- [10] J. Jung, S.-M. Lee, and H. Myung, “Indoor mobile robot localization and mapping based on ambient magnetic fields and aiding radio sources,” *IEEE Trans. Instrum. Meas.*, vol. 64, no. 7, pp. 1922–1934, Jul. 2015.
- [11] I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Röning, “Magnetic field-based SLAM method for solving the localization problem in mobile robot floor-cleaning task,” in *Proc. 15th Int. Conf. Adv. Robot. (ICAR)*, Jun. 2011, pp. 198–203.
- [12] I. Torroba, M. Cellà, A. Terán, N. Rolleberg, and J. Folkesson, “Online stochastic variational gaussian process mapping for large-scale bathymetric SLAM in real time,” *IEEE Robot. Automat. Lett.*, vol. 8, no. 6, pp. 3150–3157, Jun. 2023.
- [13] S. Barkby, S. B. Williams, O. Pizarro, and M. V. Jakuba, “Bathymetric SLAM with no map overlap using Gaussian Processes,” *Proc. Int. Conf. Intell. Robots Syst.*, pp. 1242–1248, Sep. 2011.
- [14] M. Kjaergaard, E. Bayramoglu, A. S. Massaro, and K. Jensen, “Terrain mapping and obstacle detection using Gaussian processes,” in *Proc. 10th Int. Conf. Mach. Learn. Appl. Workshops*, vol. 1, Dec. 2011, pp. 118–123.
- [15] H. Yu and B. Lee, “Terrain field SLAM and uncertainty mapping using Gaussian process,” in *Proc. 18th Int. Conf. Control, Automat. Syst.*, 2018, pp. 1077–1080.
- [16] A. Solin and S. Särkkä, “Hilbert space methods for reduced-rank Gaussian process regression,” *Statist. Comput.*, vol. 30, pp. 419–446, 2014.
- [17] J. Jung, S.-M. Lee, and H. Myung, “Indoor mobile robot localization using ambient magnetic fields and range measurements,” in *Robot Intelligence Technology and Applications 2: Results from the 2nd International Conference on Robot Intelligence Technology and Applications*, in Advances in Intelligent Systems and Computing, J.-H. Kim, E. T. Matson, H. Myung, P. Xu, and F. Karray, Eds. Cham: Springer International Publishing, 2014, pp. 137–143.
- [18] A. Viseras et al., “Decentralized multi-agent exploration with online-learning of Gaussian processes,” in *Proc. Int. Conf. Robot. Automat.*, May 2016, pp. 4222–4229.
- [19] Y. Ding, R. Kondor, and J. Eskreis-Winkler, “Multiresolution kernel approximation for Gaussian process regression,” in *Advances in Neural Inf. Process. Syst.*, vol. 30. Curran Associates, Inc., 2017.
- [20] F. Viset, R. Helmons, and M. Kok, “An extended Kalman filter for magnetic field SLAM using Gaussian process regression,” *Sensors*, vol. 22, no. 8, Jan. 2022, Art. no. 2833.
- [21] D. Jang, J. Yoo, C. Y. Son, D. Kim, and H. J. Kim, “Multi-robot active sensing and environmental model learning with distributed Gaussian process,” *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 5905–5912, Oct. 2020.
- [22] F. Viset, J. T. Gravdahl, and M. Kok, “Magnetic field norm SLAM using Gaussian process regression in foot-mounted sensors,” in *Proc. Eur. Control Conf. (ECC)*, Jun. 2021, pp. 392–398.
- [23] H. Bijl, J.-W. van Wingerden, T. B. Schön, and M. Verhaegen, “Online sparse Gaussian process regression using FITC and PITC approximations,” *IFAC-PapersOnLine*, vol. 48, no. 28, pp. 703–708, Jan. 2015.
- [24] J. Quiñonero-Candela and C. E. Rasmussen, “A unifying view of sparse approximate Gaussian Process Regression,” *J. Mach. Learn. Res.*, vol. 6, pp. 1939–1959, Dec. 2005.
- [25] J. Hensman, N. Durrande, and A. Solin, “Variational Fourier features for Gaussian processes,” *J. Mach. Learn. Res.*, vol. 18, Nov. 2016.
- [26] F. M. Viset, “Scalable magnetic field mapping and localization using gaussian process regression,” Ph.D. Thesis, Delft Univ. Technol., Nov. 2024.
- [27] A. Kullberg, I. Skog, and G. Hendeby, “Online joint state inference and learning of partially unknown state-space models,” *IEEE Trans. Signal Process.*, vol. 69, pp. 4149–4161, 2021.
- [28] S. J. Julier, “A sparse weight Kalman filter approach to simultaneous localisation and map building,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.. Expanding Societal Role Robot. Next Millennium (Cat. No. 01CH37180)*, vol. 3, 2001, pp. 1251–1256.
- [29] R. Karlsson and F. Gustafsson, “Bayesian surface and underwater navigation,” *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4204–4213, Nov. 2006.
- [30] S. Vasudevan, F. Ramos, E. Nettleton, H. Durrant-Whyte, and A. Blair, “Gaussian process modeling of large scale terrain,” in *Proc. Int. Conf. Robot. Automat.*, May 2009, pp. 1047–1053.
- [31] P. Tichavský, O. Straka, and J. Duník, “Grid-based Bayesian filters with functional decomposition of transient density,” *IEEE Trans. Signal Process.*, vol. 71, pp. 92–104, 2023.
- [32] F. Gustafsson et al., “Particle filters for positioning, navigation, and tracking,” *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 425–437, Feb. 2002.
- [33] M. Osman, F. Viset, and M. Kok, “Indoor SLAM using a foot-mounted IMU and the local magnetic field,” in *Proc. 25th Int. Conf. Inf. Fusion*, Jul. 2022, pp. 1–7.
- [34] R. B. Gramacy and D. W. Apley, “Local Gaussian process approximation for large computer experiments,” *J. Comput. Graph. Statist.*, vol. 24, no. 2, pp. 561–578, Feb. 2015.
- [35] E. Snelson and Z. Ghahramani, “Local and global sparse Gaussian process approximations,” in *Proc. 11th Int. Conf. Artif. Intell. Statist. PMLR*, Mar. 2007, pp. 524–531.
- [36] C. Park and J. Z. Huang, “Efficient computation of gaussian process regression for large spatial data sets by patching local Gaussian processes,” *J. Mach. Learn. Res.*, vol. 17, no. 174, pp. 1–29, 2016.
- [37] C. Park, J. Z. Huang, and Y. Ding, “Domain decomposition approach for fast Gaussian process regression of large spatial data sets,” *J. Mach. Learn. Res.*, vol. 12, no. 47, pp. 1697–1728, 2011.
- [38] A. Wilson and H. Nickisch, “Kernel interpolation for scalable structured gaussian processes (KISS-GP),” in *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 37, Jul. 2015, pp. 1775–1784.
- [39] M. Yadav, D. Sheldon, and C. Musco, “Faster Kernel Interpolation for Gaussian processes,” in *Proc. 24th Int. Conf. Artif. Intell. Statist.*, Mar. 2021, pp. 2971–2979.
- [40] A. G. O. Mutambra, *Decentralized Estimation and Control for Multi-sensor Systems*. Boca Raton, FL, USA: CRC Press, Jan. 1998.
- [41] N. Wahlström, M. Kok, T. B. Schön, and F. Gustafsson, “Modeling magnetic fields using Gaussian processes,” in *Proc. Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 3522–3526.
- [42] M. R. Walter, R. M. Eustice, and J. J. Leonard, “Exactly sparse extended information filters for feature-based SLAM,” *Int. J. Robot. Res.*, vol. 26, no. 4, pp. 335–359, Apr. 2007.
- [43] R. E. Turner, “Statistical models for natural sounds,” Ph.D. dissertation, Gatsby Computational Neuroscience Unit, UCL, 2010.
- [44] GEBCO Compilation Group, “Gridded bathymetry data GEBCO (General Bathymetric Chart of the Oceans) Grid.” 2021. [Online]. Available: gebco.net/data_and_products/gridded_bathymetry_data/
- [45] I. Skog, “OpenShoe Matlab Framework,” 2012.
- [46] I. Skog, J.-O. Nilsson, P. Händel, and A. Nehorai, “Inertial sensor arrays, maximum likelihood, and Cramér-Rao bound,” *IEEE Trans. Signal Process.*, vol. 64, no. 16, pp. 4218–4227, Aug. 2016.