Information Filter Occupancy Mapping using Decomposable Radial Kernels

Siwei Guo and Nikolay A. Atanasov

***Abstract*— Building occupancy maps of the environment is a fundamental problem for robot autonomy. A common assumption in early work was that the occupancy states of different map elements are independent. Recently, Gaussian Process (GP) techniques were proposed to capture correlation, which is important not only for improved accuracy but also for uncertainty quantification and autonomous exploration based on the predicted occupancy of nearby unexplored areas. Despite these desirable properties, current GP mapping techniques are limited to small maps and slow inference speeds. This paper proposes an information space formulation of the GP mapping problem. If a decomposable radial kernel is evaluated over a latent grid of pseudo-input points, the resulting kernel matrix has a Kronecker-product-of-Toeplitz-matrices structure that allows very efficient representation of the occupancy distribution. We utilize this structure to design an information filter occupancy mapping algorithm with linear time and memory complexity that still permits continuous space observations and predictions.**

I. Introduction

Modern robots are equipped with sensors able to provide overwhelming amounts of data within seconds. Interpreting the data and extracting a concise representation of a robot's surroundings is a key problem in robot autonomy. In this paper, we focus on building large occupancy maps using point cloud observations from LIDAR or depth camera sensors. A common approach is to describe the environment as a collection of volumes, each associated with a binary variable, indicating whether the volume is occupied or free. There exist efficient methods for dense occupancy mapping, including mesh-based [1]–[3], voxel-based [4]–[6], and surface-based [7]–[10] techniques. All of these techniques make the assumption that the occupancy states of different map elements are independent. A primary objectives of this work is to model correlation among map elements, not only to improve accuracy but also to predict the visibility and geometric structure of nearby unexplored areas, which is important for autonomous exploration [11]–[13].

Maintaining correlation in occupancy maps has been considered by several recent works relying on Gaussian Process (GP) classification and regression [14]–[17]. GP occupancy mapping approaches rely on a kernel function to model correlation and allow resolution-free occupancy estimation. GP classification [18, Ch. 3.3] deals accurately with the hybrid nature of the problem, involving discrete measurements (occupied or free) and a continuous occupancy function. However, due to the non-Gaussian measurement likelihood, the posterior distribution of the occupancy function cannot be determined analytically and has to be approximated with an assumed density [19], [20] or iterative methods such as Laplace approximation [21] or expectation propagation [22]. O'Callaghan et al. [14], [23], [24] recognized that GP regression can be used to estimate the latent occupancy function using Gaussian measurements and its posterior can be squashed to a binary (free-occupied) observation model only afterwards. The resulting probabilistic least-squares method is simpler and more efficient than GP classification with negligible loss in accuracy. Despite this, the computational complexity of GP mapping scales cubically with the number of sensor measurements since the matrix of kernel correlations among different measurement locations needs to be inverted in the inference process.

Several fundamental techniques [25], [26], involving sparse kernels and matrix factorization, have been proposed to enable efficient inversion of the covariance matrix. Specific to occupancy mapping, Kim and Kim [16], [27], [28] use a sparse kernel and Bayesian Committee Machines (BCM) to perform small GP regressions with subsets of the training data. Similarly, Wang and Englot [17] partition the measurement data among several GP regressions and use BCM to fuse the sensor-level regressions into a full map. Ramos et al. [29] proposed fast kernel approximations to project the occupancy data into a Hilbert space where a logistic regression classifier can distinguish occupied and free space. This idea has been extended to dynamic maps [30] as well as into a variational autoencoder formulation [31] that compresses the local spatial information into a latent low-dimensional feature representation and then decodes it to infer the occupancy of a scene. These GP techniques have demonstrated a key ability to propagate a joint occupancy distribution but still have time and memory complexity limitations when large maps are considered. Variational inference techniques [32], [33] to choose a sparse set of inducing points [34], [35] that summarize the full GP model have been proposed but have not been applied to occupancy mapping.

The main **contribution** of this paper is an approach for storing and updating a joint map occupancy distribution in terms of a Gaussian information vector and information matrix over a latent grid structure of inducing points. Similar to other GP occupancy mapping techniques, our representation can be updated from *continuous-space* occupancy observations and can predict the occupancy values of *continuous-space* query points. In contrast to BCM techniques that

We gratefully acknowledge support from ARL DCIST CRA W911NF-17-2-0181 and NSF NRI CNS-1830399.

S. Guo is with Brain Corp, San Diego, CA 92093, USA guo@braincorporation.com. N. Atanasov is with the Department of Electrical and Computer Engineering, University of California San Diego, CA 92093, USA natanasov@ucsd.edu

consider independent decompositions of space, our approach avoids independence assumptions by computing kernel correlations from the inputs and query points to a global latent grid structure. Maintaining the joint distribution over a large grid is possible due to two key insights. First, the kernel matrix, associated with kernels that decompose across dimensions, and its inverse can be computed over a grid as the Kronecker product of kernel matrices of one-dimensional kernels. Second, approximating the latent occupancy values over a grid using GP regression is exactly equivalent to information filtering with particularly simple parameter updates. These observations allow us to design an Information Filter Occupancy Mapping (IFOM) algorithm whose memory complexity of storing and time complexity of updating the information space parameters of the joint occupancy distribution is linear both in the number of grid cells and in the number of sensor observations. The structure of decomposable kernels has recently been exploited by [36]–[38] to design scalable GP inference but these ideas have not been applied to occupancy mapping and have not been associated with information filtering.

II. Problem Formulation

Let $\mathcal{X}^{(d)} := [\underline{\mathbf{x}}^{(d)}, \bar{\mathbf{x}}^{(d)}]$ be a closed interval in $\mathbb{R}$ and let $\mathcal{X} := \mathcal{X}^{(1)} \times \cdots \times \mathcal{X}^{(D)} \subset \mathbb{R}^D$ be a closed rectangle in D-dimensional Euclidean space that we are interested in mapping. The occupancy of a location $\mathbf{x} \in \mathcal{X}$ is specified by a latent function $f : \mathcal{X} \to \mathbb{R}$ that assigns a free label $y = -1$ or an occupied label $y = 1$ according to a Bernoulli probability mass function (pmf):

$$p(y \mid f(\mathbf{x})) = \Phi\left(\frac{y f(\mathbf{x})}{\sigma}\right), \qquad y \in \{-1, 1\} \tag{1}$$

where $\Phi(z)$ is the *probit* function, i.e., the cumulative distribution function of a standard Gaussian, and σ is a scaling parameter. The probit[1] serves to squash the continuous range of $f(\mathbf{x})$ into a range $[0, 1]$ representing a valid pmf.

Problem (Occupancy Mapping)**.** Given a set of occupancy measurements, $\mathcal{D} := \{(\mathbf{x}(t), y(t)) \mid \mathbf{x}(t) \in \mathcal{X}, y(t) \in \{-1, 1\}, t = 1, \ldots, T\}$, generated from the observation model (1), construct an approximation $\hat{f} : \mathcal{X} \to \mathbb{R}$ of the latent occupancy function f.

III. Background

Consider a linear model $f(\mathbf{x}) := \boldsymbol{\omega}^T \mathbf{x}$, where $\boldsymbol{\omega} \in \mathbb{R}^D$ is a vector parameterizing f. The occupancy $y_* \in \{-1, 1\}$ of an arbitrary location $\mathbf{x}_* \in \mathcal{X}$ can be predicted based on the available observations $\mathcal{D}$ via:

$$p(y_* \mid \mathbf{x}_*, \mathcal{D}) = \int p(y_* \mid \mathbf{x}_*, \boldsymbol{\omega}) p(\boldsymbol{\omega} \mid \mathcal{D}) d\boldsymbol{\omega}, \tag{2}$$

[1]Another usual choice of a squashing function is the sigmoid $\left(1 + e^{-z}\right)^{-1}$. We chose the probit $\Phi(z)$ because it allows us to establish a clear connection between classification and regression in Sec. III and to compute integrals $\int \Phi(z)\phi(z)dz$ with respect to the Gaussian density function in closed form.

where $p(y_* | \mathbf{x}_*, \boldsymbol{\omega}) = \Phi(y_* \boldsymbol{\omega}^T \mathbf{x}_* / \sigma)$ based on (1). Thus, the classification problem reduces to approximating the posterior parameter distribution $p(\boldsymbol{\omega}|\mathcal{D}) \propto \prod_{t=1}^{T} \Phi(y(t)\boldsymbol{\omega}^T \mathbf{x}(t)/\sigma) p(\boldsymbol{\omega})$ based on the data $\mathcal{D}$ and a given (usually Gaussian) prior $p(\boldsymbol{\omega})$ over the parameters.

Instead of a linear model, state-of-the-art classification techniques use more complex models for f such as neural networks [39] or Gaussian processes [18]. In this paper, we focus on GP models, which use a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ to capture correlation among the environment locations. A common kernel choice is the Gaussian radial basis function:

$$k_{RBF}(\mathbf{x}, \mathbf{x}') := b \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^T S^{-1} (\mathbf{x} - \mathbf{x}')\right) \tag{3}$$

with hyperparameters $b \in \mathbb{R}_{>0}$ and diagonal positive-semidefinite $S \in \mathbb{R}^{D \times D}$ controlling the amplitude and wiggliness. Other kernel choices include Matérn kernels, γ-exponential kernels, piecewise polynomial kernels, etc. [18, Ch. 4.2]. The kernel is used to place a GP prior on the latent occupancy function $f \sim \mathcal{GP}(0, k)$, which means that its values $\mathbf{f} := [f(\mathbf{x}(1)) \ \cdots \ f(\mathbf{x}(T))]^T$ over the training set, before taking the observations $\mathbf{y} := [y(1) \ \cdots \ y(T)]^T$ into account, have a Gaussian distribution $p(\mathbf{f}) = \phi(\mathbf{f}; \mathbf{0}, K)$ with mean $\mathbf{0}$ and covariance matrix $K \in \mathbb{R}^{T \times T}$ with elements $K_{ij} := k(\mathbf{x}(i), \mathbf{x}(j))$. As in the linear model case, since the observation model (1) is non-Gaussian, the main challenge is to compute the posterior distribution $p(f|\mathcal{D}) \propto \prod_{t=1}^{T} p(y(t) \mid f(\mathbf{x}(t))) p(f)$ and subsequently the integral in (2). The posterior is usually approximated via a Gaussian distribution using Laplace approximation [21] or expectation propagation [22] and, in turn, the integral in (2) can be computed in closed form (see eq. (15)). While GP classification uses a theoretically sound model and leads to accurate results, the posterior approximation may be very computationally demanding for large training sets.

Several authors [14], [16], [17], [28] proposed the use of GP regression instead of GP classification in the context of occupancy mapping. This can be justified by interpreting the observation model (1) as follows:

$$y = \mathbf{sgn}(f(\mathbf{x}) + \epsilon), \qquad \epsilon \sim \mathcal{N}(0, \sigma^2) \tag{4}$$

where $\mathbf{sgn}$ is the sign function. This model is equivalent to the model in (1) since:

$$\mathbb{P}(y = 1 \mid f(\mathbf{x})) = \mathbb{P}(\epsilon > -f(\mathbf{x})) = \Phi(f(\mathbf{x})/\sigma). \tag{5}$$

Hence, instead of using classification, one can pretend that the occupancy measurements $y(t)$ are direct observations of $f(\mathbf{x}(t))$ perturbed by Gaussian noise ϵ, apply GP regression to estimate f, and squash its value to $\Phi(f(\mathbf{x}_*)/\sigma)$ only after the inference process. More precisely, for an arbitrary query $\mathbf{x}_*(1), \ldots, \mathbf{x}_*(M) \in \mathcal{X}$, the joint distribution of $\mathbf{y}$ and $\mathbf{f}_* := [f(\mathbf{x}_*(1)) \ \cdots \ f_*(\mathbf{x}(M))]^T$ is:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K + \sigma^2 I & K_* \\ K_*^T & K_{**} \end{bmatrix}\right) \tag{6}$$

where $K_* \in \mathbb{R}^{M\times T}$ is a matrix with elements $[K_*]_{ij} := k(\mathbf{x}_*(i), \mathbf{x}(j))$ and $K_{**} \in \mathbb{R}^{M\times M}$ is a matrix with elements $[K_{**}]_{ij} := k(\mathbf{x}_*(i), \mathbf{x}_*(j))$ [18]. The posterior computed via GP regression is the distribution of $\mathbf{f}_*$ conditioned on $\mathbf{y}$ and can be obtained from (6) via Schur complementation:

$$\mathbf{f}_*|\mathcal{D} \sim \mathcal{N}\left(K_*(K+\sigma^2 I)^{-1}\mathbf{y}, K_{**} - K_*(K+\sigma^2 I)^{-1}K_*^T\right).$$

IV. Technical Approach

GP regression is a non-parametric method and inference requires storing all training data $\mathcal{D}$ as seen in the expression for $\mathbf{f}_*|\mathcal{D}$ above. We propose the following steps to make the inference in occupancy mapping problems significantly less computationally and memory demanding with negligible accuracy loss. First, instead of storing all training data $\mathcal{D}$, we introduce a latent finite grid $\mathcal{C} \subset \mathcal{X}$ and only keep a distribution over the N values $\mathbf{f}_\# \in \mathbb{R}^N$ of the grid points $\mathbf{x}_\# \in \mathcal{C}$. This is similar to the idea of pseudo-inputs in sparse GP regression [40] but instead of optimizing the pseudo-input positions we keep a fixed grid. A latent grid is well-suited for mapping because the kernel matrix resulting from a decomposable radial kernel evaluated over the grid $\mathcal{C}$ has a special Kronecker-product-of-Toeplitz-matrices structure (Sec. IV-A). This allows very efficient storage of the map distribution even for large environments. Second, approximating the finite set of values $\mathbf{f}_\#$ using GP regression is equivalent to Kalman filtering and, in turn, to information filtering [41]. Hence, one can propagate the latent function distribution in information form, which is *exact*, memory and computationally efficient, and can still be updated from continuous-space measurements (Sec. IV-B). The trade-off is that to compute the predictive pmf in (2), one needs to recover the latent mean and covariance of $\mathbf{f}_\#$, which is the main computational challenge for our approach. Our motivation is that *recovering the whole map mean and covariance is not necessary during online mapping* because map uncertainty can be evaluated using the information matrix, while collision checking can be performed locally by only recovering small portions of the mean (Sec. IV-C). These observations leads to our IFOM algorithm (Sec. IV-D).

A. Kronecker and Toeplitz kernel structure

To motivate the choice of a latent pseudo-point grid $\mathcal{C}$, we analyze the structure of the kernel matrix that arises when a decomposable radial kernel is evaluated over the grid. We make the following assumption throughout the paper.

Assumption 1. The kernel function is *decomposable* into a product of one-dimensional kernels,

$$k(\mathbf{x}, \mathbf{x}') = \prod_{d=1}^{D} k^{(d)}\left(\mathbf{x}^{(d)}, \mathbf{x}'^{(d)}\right), \tag{7}$$

and each $k^{(d)}\left(\mathbf{x}^{(d)}, \mathbf{x}'^{(d)}\right)$ is *radial*, i.e., its value depends only on the distance $|\mathbf{x}^{(d)} - \mathbf{x}'^{(d)}|$ between its inputs.

This assumption means that the correlation between the occupancy of two points in $\mathcal{X}$ is independent along the axes in 3-D space, while the per-axis correlation depends only on distance. This is a weak assumption that only states that strong correlation along one direction does not imply correlation along another direction. It is general enough to capture correlations along different orientations of structures in the environment but one set of kernel hyperparameters might not capture different parts of the environment well (e.g., vertical hallways vs diagonal hallways). Many commonly used kernels satisfy this assumption. For example, the Gaussian kernel in (3) with $b = \prod_{d=1}^{D} \mathbf{b}^{(d)}$ and $S = \mathbf{diag}\left(\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(D)}\right)$ is radial and decomposable:

$$k_{RBF}(\mathbf{x}, \mathbf{x}') = \prod_{d=1}^{D} \mathbf{b}^{(d)} \exp\left(-\frac{\left(\mathbf{x}^{(d)} - \mathbf{x}'^{(d)}\right)^2}{2\mathbf{s}^{(d)}\mathbf{s}^{(d)}}\right). \tag{8}$$

Remark. We emphasize that Assumption 1 *does not* preclude the use of *automatic relevance determination* [18] for optimization of the kernel hyperparameters. For example, it permits the use of a Gaussian kernel with ellipsoidal covariance. Rather than hyperparameter optimization, this paper focuses on efficient online training, storage, and inference.

Let $\mathcal{X}^{(d)}$ be discretized into a finite grid $\cup_{k=1}^{\mathbf{n}^{(d)}}[\underline{\mathbf{x}}^{(d)} + (k-1)\mathbf{r}^{(d)}, \underline{\mathbf{x}}^{(d)} + k\mathbf{r}^{(d)}]$ with $\mathbf{n}^{(d)}$ cells and resolution $\mathbf{r}^{(d)} := (\bar{\mathbf{x}}^{(d)} - \underline{\mathbf{x}}^{(d)})/\mathbf{n}^{(d)}$. Let $\mathcal{C}^{(d)} := \{\mathbf{c}_k^{(d)} \mid \mathbf{c}_k^{(d)} = \underline{\mathbf{x}}^{(d)} + \mathbf{r}^{(d)}(k-1/2), k = 1, \ldots, \mathbf{n}^{(d)}\}$ be the set of cell centers. Finally, let $\mathcal{C} := \mathcal{C}^{(1)} \times \cdots \times \mathcal{C}^{(D)} = \{\mathbf{x}_i \mid \mathbf{x}_i^{(d)} = \mathbf{c}_{s(i,d)}^{(d)}, d = 1, \ldots D\}$ be the set of voxel centers in $\mathbb{R}^D$, where the function $s(i,d)$ maps an index $i \in \{1, \ldots, N\}$ with $N := \prod_{d=1}^{D} \mathbf{n}^{(d)}$ to subindices in each dimension. Due to Assumption 1, the matrix $K_{\#\#} \in \mathbb{R}^{N\times N}$ resulting from evaluating a decomposable kernel k over all pairs of grid points $\mathbf{x}_\#, \mathbf{x}'_\# \in \mathcal{C}$ has a Kronecker product structure:

$$K_{\#\#} = \bigotimes_{d=1}^{D} K_{\#\#}^{(d)}, \quad \textit{i.e.,} \quad [K_{\#\#}]_{ij} = \prod_{d=1}^{D} \left[K_{\#\#}^{(d)}\right]_{s(i,d),s(j,d)} \tag{9}$$

where $K_{\#\#}^{(d)} \in \mathbb{R}^{\mathbf{n}^d \times \mathbf{n}^d}$ is the matrix associated with $k^{(d)}$. This observation has been exploited by [36], [37], [42] to design GP learning and inference algorithms with $O(DN^{1+1/D})$ time and $O(DN^{2/D})$ memory complexity. The Kronecker structure of $K_{\#\#}$ carries over to LDLT and eigendecompositions[2] and hence $K_{\#\#}^{-1} = \bigotimes_{d=1}^{D}\left(K_{\#\#}^{(d)}\right)^{-1}$. If in addition $k^{(d)}$ is radial, the matrices $K_{\#\#}^{(d)}$ are Toeplitz with constant diagonals, $[K_{\#\#}^{(d)}]_{ij} = [K_{\#\#}^{(d)}]_{i+1,j+1}$. The Toeplitz structure can be exploited for GP inference [43], [44] with $O(\mathbf{n}^{(d)} \log \mathbf{n}^{(d)})$ computational complexity. For our purposes, it allows efficient storage and inversion of $K_{\#\#}^{(d)}$. A symmetric Toeplitz matrix and its inverse are also persymmetric (symmetric with respect to the northeast-to-southwest diagonal) and hence only $\frac{1}{4}\mathbf{n}^{(d)}\mathbf{n}^{(d)}$ elements need to be stored. Also, a symmetric Toeplitz matrix can be inverted using $2\mathbf{n}^{(d)}\mathbf{n}^{(d)}$ operations via the Trench algorithm [45]. For a Gaussian kernel matrix, this can be improved to $\frac{1}{2}\mathbf{n}^{(d)}\mathbf{n}^{(d)}$ operations [46].

[2]One can compute $K_{\#\#} = QVQ^T$ by combining the eigendecompositions of the small $K_{\#\#}^{(d)}$ via $Q = \bigotimes_{d=1}^{D} Q^{(d)}$ and $V = \bigotimes_{d=1}^{D} V^{(d)}$

Finally, we note that for a radial and monotone decreasing kernel, such as the Gaussian RBF, the correlation, $k(\mathbf{x}, \mathbf{x}_\#)$, between a query point $\mathbf{x} \in \mathcal{X}$ and a grid point $\mathbf{x}_\# \in \mathcal{C}$ is approximately zero when $\|\mathbf{x} - \mathbf{x}_\#\|$ is larger than a threshold. In this case, both the vector $\mathbf{k}_\#^T := [k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_N)]$ for $\mathbf{x}_i \in \mathcal{C}$ and the matrix $K_{\#\#}$ can be assumed *sparse*.

B. Equivalence between Gaussian Processes and Kalman filtering on finite spaces

The special structure of the kernel matrix for a radial decomposable kernel discussed in Sec. IV-A, motivates an occupancy mapping approach which uses the continuous-space online observations $(\mathbf{x}(t), y(t))$ to efficiently update and store a distribution over the latent grid values $\mathbf{f}_\#$ and subsequently predict the labels $\mathbf{y}_*$ of continuous-space query points $\mathbf{x}_*(1), \ldots, \mathbf{x}_*(M)$ based on $\mathbf{f}_\#$. Consider the relationship between an observation $(\mathbf{x}, y)$ and $\mathbf{f}_\#$. Since $f(\mathbf{x})$ and $\mathbf{f}_\#$ are jointly Gaussian with zero mean and covariance given by the kernel matrix:

$$\begin{aligned} p(y \mid \mathbf{x}, \mathbf{f}_\#) &= \int p(y \mid f(\mathbf{x})) p(f(\mathbf{x}) \mid \mathbf{f}_\#) df(\mathbf{x}) \\ &= \int \Phi\left(\frac{yf}{\sigma}\right) \phi\left(f; \mathbf{k}_\#^T K_{\#\#}^{-1} \mathbf{f}_\#, k(\mathbf{x}, \mathbf{x}) - \mathbf{k}_\#^T K_{\#\#}^{-1} \mathbf{k}_\#\right) df \\ &= \Phi\left(\frac{y \mathbf{k}_\#^T K_{\#\#}^{-1} \mathbf{f}_\#}{\sqrt{k(\mathbf{x}, \mathbf{x}) - \mathbf{k}_\#^T K_{\#\#}^{-1} \mathbf{k}_\# + \sigma^2}}\right). \end{aligned} \tag{10}$$

Using the interpretation in (4), we can pretend that the label y is a direct observations of the latent values:

$$y = \mathbf{k}_\#^T K_{\#\#}^{-1} \mathbf{f}_\# + \epsilon(\mathbf{x}) \qquad \epsilon(\mathbf{x}) \sim \mathcal{N}(0, \lambda(\mathbf{x}) + \sigma^2) \tag{11}$$

where $\lambda(\mathbf{x}) := k(\mathbf{x}, \mathbf{x}) - \mathbf{k}_\#^T K_{\#\#}^{-1} \mathbf{k}_\#$. Then, GP regression and Kalman filtering are equivalent.

Proposition 1. *Suppose that the observations $(\mathbf{x}(t), y(t)) \in \mathcal{D}$ are obtained according to the observation model in (11). Then, the posterior distribution $p(\mathbf{f}_\# \mid \mathcal{D})$ of the latent values over the grid points $\mathbf{x}_\# \in \mathcal{C}$ computed via GP regression with zero prior mean and kernel k is exactly equal to the posterior distribution computed by a Kalman filter with prior mean $\mathbf{0}$ and prior covariance $K_{\#\#} \in \mathbb{R}^{N \times N}$.*

Proof. The joint distribution of $\mathbf{y}$ and $\mathbf{f}_\#$ is:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_\# \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K + \Lambda + \sigma^2 I & K_\# \\ K_\#^T & K_{\#\#} \end{bmatrix}\right) \tag{12}$$

where $K_\# \in \mathbb{R}^{T \times N}$ is a matrix with elements $[K_\#]_{t,j} := k(\mathbf{x}(t), \mathbf{x}_\#(j))$ and $\Lambda \in \mathbb{R}^{T \times T}$ is a diagonal matrix with $\Lambda_{tt} := \lambda(\mathbf{x}(t))$. The distribution of $\mathbf{f}_\# | \mathcal{D}$ computed via GP regression is obtained via Schur complementation:

$$\mathcal{N}\left(K_\#(K + \Lambda + \sigma^2 I)^{-1} \mathbf{y}, K_{\#\#} - K_\#(K + \Lambda + \sigma^2 I)^{-1} K_\#^T\right).$$

Let $H \in \mathbb{R}^{T \times N}$ be a matrix whose t-th row is $\mathbf{k}_\#^T K_{\#\#}^{-1}$. Given observations $\mathbf{y} = H\mathbf{f}_\# + \boldsymbol{\epsilon}$ with noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \Lambda + \sigma^2 I)$, the Kalman filter posterior over $\mathbf{f}_\#$ is:

$$\mathbf{f}_\# \mid \mathcal{D} \sim \mathcal{N}\left(G\mathbf{y}, K_{\#\#} - GHK_{\#\#}\right) \tag{13}$$

where $G := K_{\#\#} H^T \left(HK_{\#\#} H^T + \Lambda + \sigma^2 I\right)^{-1}$ is the Kalman gain. Note that $K = HK_{\#\#}H^T$ and $K_\# = K_{\#\#}H^T$, which shows that (13) is equivalent to the GP posterior. □

Since Kalman filtering is also equivalent to information filtering, Prop. 1 allows us to maintain the latent function distribution in information form $\mathbf{f}_\# | \mathcal{D} \sim \mathcal{N}(\Omega^{-1}\boldsymbol{\nu}, \Omega^{-1})$:

$$\begin{aligned} \boldsymbol{\nu} &= K_{\#\#}^{-1} K_\#^T (\Lambda + \sigma^2 I)^{-1} \mathbf{y} \\ \Omega &= K_{\#\#}^{-1} + K_{\#\#}^{-1} K_\#^T (\Lambda + \sigma^2 I)^{-1} K_\# K_{\#\#}^{-1}. \end{aligned} \tag{14}$$

Once the parameters $\boldsymbol{\nu}, \Omega$ of the posterior of $\mathbf{f}_\#$ are computed based on the data $\mathcal{D}$, we can predict the label y_* at an arbitrary *continuous-space* location $\mathbf{x}_* \in \mathcal{X}$, as follows:

$$\begin{aligned} p(y_* \mid \mathbf{x}_*, \mathcal{D}) &= \int p(y_* \mid \mathbf{x}_*, \mathbf{f}_\#) p(\mathbf{f}_\# \mid \mathcal{D}) d\mathbf{f}_\# \\ &= \int \Phi\left(\frac{y_* \mathbf{k}_\#^T K_{\#\#}^{-1} \mathbf{f}_\#}{\sqrt{\lambda(\mathbf{x}_*) + \sigma^2}}\right) \phi\left(\mathbf{f}_\#; \Omega^{-1}\boldsymbol{\nu}, \Omega^{-1}\right) d\mathbf{f}_\# \\ &= \Phi\left(\frac{y_* \mathbf{k}_\#^T K_{\#\#}^{-1} \Omega^{-1} \boldsymbol{\nu}}{\sqrt{\lambda(\mathbf{x}_*) + \sigma^2 + \mathbf{k}_\#^T K_{\#\#}^{-1} \Omega^{-1} K_{\#\#}^{-1} \mathbf{k}_\#}}\right) \end{aligned} \tag{15}$$

Close inspection of (14) and (15) reveals that it is sufficient to represent the distribution of $\mathbf{f}_\# | \mathcal{D}$ using simpler parameters:

$$\begin{aligned} \boldsymbol{\gamma} &:= K_\#^T (\Lambda + \sigma^2 I)^{-1} \mathbf{y} \\ \Gamma &:= K_\#^T (\Lambda + \sigma^2 I)^{-1} K_\# \end{aligned} \tag{16}$$

whose relationship to the information space parameters $\boldsymbol{\nu}$, Ω, the regular parameters $\boldsymbol{\mu}$, Σ, and the pmf $p(y_* | \mathbf{x}_*, \mathcal{D})$ is:

$$\begin{aligned} \boldsymbol{\nu} &= K_{\#\#}^{-1} \boldsymbol{\gamma} & \Omega &= K_{\#\#}^{-1} \left(K_{\#\#} + \Gamma\right) K_{\#\#}^{-1} \\ \boldsymbol{\mu} &= K_{\#\#} \left(K_{\#\#} + \Gamma\right)^{-1} \boldsymbol{\gamma} & \Sigma &= K_{\#\#} \left(K_{\#\#} + \Gamma\right)^{-1} K_{\#\#} \end{aligned}$$

$$p(y_* \mid \mathbf{x}_*, \mathcal{D}) = \Phi\left(\frac{y_* \mathbf{k}_\#^T \left(K_{\#\#} + \Gamma\right)^{-1} \boldsymbol{\gamma}}{\sqrt{\lambda(\mathbf{x}_*) + \sigma^2 + \mathbf{k}_\#^T \left(K_{\#\#} + \Gamma\right)^{-1} \mathbf{k}_\#}}\right)$$

Instead of a batch update with all measurements at once, the $\boldsymbol{\gamma}$ and Γ parameters can be updated sequentially:

$$\begin{aligned} \boldsymbol{\gamma}_{t+1} &= \boldsymbol{\gamma}_t + \frac{\mathbf{k}_{\#,t} y(t)}{\lambda(\mathbf{x}(t)) + \sigma^2} \\ \Gamma_{t+1} &= \Gamma_t + \frac{\mathbf{k}_{\#,t} \mathbf{k}_{\#,t}^T}{\lambda(\mathbf{x}(t)) + \sigma^2} \end{aligned} \quad \text{for } t = 1, \ldots, T \tag{17}$$

with $\gamma_1 = \mathbf{0} \in \mathbb{R}^N$ and $\Gamma_1 = \mathbf{0} \in \mathbb{R}^{N \times N}$. We emphasize that $\boldsymbol{\gamma}_t$ and Γ_t can be stored and updated with linear complexity in T and N as long as the kernel function k is decomposable, radial, and monotone decreasing. In detail, $\mathbf{k}_{\#,t}$ is sparse as mentioned in Sec. IV-A and can be computed in $O(1)$ by looking up the grid cells within a distance threshold from $\mathbf{x}(t)$. The inverse kernel matrix satisfies:

$$\left[K_{\#\#}^{-1}\right]_{ij} = \prod_{d=1}^{D} \left[\left(K_{\#\#}^{(d)}\right)^{-1}\right]_{s(i,d), s(j,d)} \tag{18}$$

and $\lambda(\mathbf{x}(t))$ can be computed in $O(1)$ by precomputing the small kernel matrices $\left(K_{\#\#}^{(d)}\right)^{-1}$. Hence, computing γ_{T+1}

and Γ_{T+1} has time complexity $O(T)$. Storing γ_t and Γ_t has memory complexity $O(N)$ because Γ_t has the same sparisty structure as $K_{\#\#}$ for a radial, monotone decreasing kernel. Storing the inverse kernel matrices $\left(K_{\#\#}^{(d)}\right)^{-1}$ has complexity $O(\max_d \mathbf{n}^d)$ which is dominated by $O(N)$. Thus, storing and updating the distribution of $\mathbf{f}_\#$ in terms of $\boldsymbol{\gamma}_t$ and Γ_t has *linear memory complexity in the number of grid cells N and linear time complexity in the number of observations T.*

C. Occupancy prediction

If predicted occupancy values are needed, the expression for $p(y_*|\mathbf{x}_*, \mathcal{D})$ above needs to be evaluated. This is the main computational challenge for our method since it requires computing $(K_{\#\#} + \Gamma)^{-1} \boldsymbol{\gamma}$ and $(K_{\#\#} + \Gamma)^{-1} \mathbf{k}_\#$. While $(K_{\#\#} + \Gamma)$ is sparse it may be giant for large maps and using Cholesky or QR factorization is not feasible. Instead, we use the conjugate gradient (CG) method [47] – an iterative algorithm for solving linear systems defined by symmetric positive-definite matrices. The advantage of CG is that it maintains only matrix-vector products during its iterations, making it scalable to very large systems. The main CG computation is repeated multiplication of $(K_{\#\#} + \Gamma)$ with a conjugate direction vector $\mathbf{p}$, which may be computed efficiently due to the sparsity of $(K_{\#\#} + \Gamma)$. For completeness, we include the CG method in Alg. 1.

If instead of the occupancy likelihood $p(y_*|\mathbf{x}_*, \mathcal{D})$, it is only necessary to tell if a query point $\mathbf{x}_*$ is occupied, i.e., if $p(1|\mathbf{x}_*, \mathcal{D}) > \frac{1}{2}$, then it is sufficient to compute only $(K_{\#\#} + \Gamma)^{-1} \boldsymbol{\gamma}$ because $\Phi(x) > \frac{1}{2} \Leftrightarrow x > 0$ and since the denominator in the expression for $p(y_*|\mathbf{x}_*, \mathcal{D})$ is positive:

$$y_* = 1 \quad \Leftrightarrow \quad \mathbf{k}_\#^T (K_{\#\#} + \Gamma)^{-1} \boldsymbol{\gamma} > 0 \tag{19}$$

As a result, a complete occupancy grid map at *an arbitrary resolution* can be obtained by a single call to Alg. 1 to compute $(K_{\#\#} + \Gamma)^{-1} \boldsymbol{\gamma}$. Given $\boldsymbol{\gamma} \in \mathbb{R}^N$, $\Gamma \in \mathbb{R}^{N \times N}$, due to the sparsity of $(K_{\#\#} + \Gamma)$, obtaining a map with M elements via the CG algorithm has complexity $O(LMN)$ where L is the number of CG iterations. Computing the occupancy likelihood $p(y_*|\mathbf{x}_*, \mathcal{D})$ requires a second call to Alg. 1 to obtain $\mathbf{k}_\#^T (K_{\#\#} + \Gamma)^{-1} \mathbf{k}_\#$ but due to sparsity in $\mathbf{k}_\#$, a much smaller matrix and vector can be provided.

Predicting occupancy values over the whole latent grid is much more computationally demanding than propagating the information space distribution of $\mathbf{f}_\#$. Our motivation for using an information space representation is that recovering the occupancy values of the *whole map* is not necessary during online mapping. Instead, only the occupancy values of a small set of query points may be needed for collision checking, while the complete map can be recovered offline. We propose an approximation to $K_{*\#}(K_{\#\#} + \Gamma)^{-1}\gamma$ that may be used to predict a subset of the occupancy values for the purpose of collision checking much more efficiently than CG. Exploiting the kernel matrix sparsity, suppose that the points in the query set $\mathbf{x}_*(1), \ldots, \mathbf{x}_*(M)$ are close only to a small subset $\mathcal{A}$ of the latent grid $\mathcal{C}$, while the remaining grid points $\mathcal{B} \subset \mathcal{C}$ are far enough that $k(\mathbf{x}_*(i), \mathbf{x}_\#) \approx 0$ for

Algorithm 1 Conjugate Gradient Method

1: **Input**: symmetric positive-definite $A \in \mathbb{R}^{N \times N}$, $\mathbf{b} \in \mathbb{R}^N$
2: **Output**: $\boldsymbol{\mu} = A^{-1}\mathbf{b} \in \mathbb{R}^N$
3: $\boldsymbol{\mu} = \mathbf{0}, \quad \mathbf{r} = \mathbf{p} = \mathbf{b}$
4: **loop**
5: $\quad \mathbf{q} = A\mathbf{p}, \quad \rho = \mathbf{r}^T\mathbf{r}, \quad \alpha = \frac{\rho}{\mathbf{p}^T\mathbf{q}}$
6: $\quad \boldsymbol{\mu} = \boldsymbol{\mu} + \alpha\mathbf{p}, \quad \mathbf{r} = \mathbf{r} - \alpha\mathbf{q}, \quad \mathbf{p} = \mathbf{r} + \frac{\mathbf{r}^T\mathbf{r}}{\rho}\mathbf{p}$

$i \in \{1, \ldots, M\}$ and $\mathbf{x}_\# \in \mathcal{B}$. Let $A := (K_{\#\#} + \Gamma)$ and decompose $\boldsymbol{\mu} := A^{-1}\boldsymbol{\gamma}$ as follows:

$$\begin{bmatrix} \boldsymbol{\mu}_\mathcal{A} \\ \boldsymbol{\mu}_\mathcal{B} \end{bmatrix} = \begin{bmatrix} A_{\mathcal{A},\mathcal{A}} & A_{\mathcal{A},\mathcal{B}} \\ A_{\mathcal{A},\mathcal{B}}^T & A_{\mathcal{B},\mathcal{B}} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\gamma}_\mathcal{A} \\ \boldsymbol{\gamma}_\mathcal{B} \end{bmatrix}. \tag{20}$$

We only need $\boldsymbol{\mu}_\mathcal{A}$ to predict the occupancy values of the query set. Using the block matrix inversion lemma:

$$\boldsymbol{\mu}_\mathcal{A} = \left(A_{\mathcal{A},\mathcal{A}} - A_{\mathcal{A},\mathcal{B}} A_{\mathcal{B},\mathcal{B}}^{-1} A_{\mathcal{A},\mathcal{B}}\right)^{-1} \left(\boldsymbol{\gamma}_\mathcal{A} - A_{\mathcal{A},\mathcal{B}} A_{\mathcal{B},\mathcal{B}}^{-1} \boldsymbol{\gamma}_\mathcal{B}\right)$$

Assuming that $\mathcal{A}$ is a small subset of $\mathcal{C}$, the above expression can be computed efficiently as long as $A_{\mathcal{B},\mathcal{B}}^{-1}$ is available. We use a diagonal-matrix approximation to $A_{\mathcal{B},\mathcal{B}}^{-1}$.

Proposition 2. *Let A be a symmetric positive definite matrix. The diagonal matrix D that best approximates A^{-1} according to the Frobenius norm ($\min_D \|AD - I\|_F^2$) satisfies $D_{ii} = \frac{A_{ii}}{\sum_{j=1}^M A_{ij}^2}$.*

Proof. Taking the gradient of and equating it to 0 leads to:

$$0 = \nabla_\Lambda \, \mathbf{tr}\left((AD - I)(AD - I)^T\right) = 2\,\mathbf{tr}(A^2 D + A)$$

which is satisfied as long as $D_{ii} = \frac{A_{ii}}{\sum_{j=1}^M A_{ij}^2}$. □

Based on Prop. 2, we can check the occupancy of a query point $\mathbf{x}_*$ approximately via:

$$\mathbf{k}_\mathcal{A}^T \boldsymbol{\mu}_\mathcal{A} \approx \mathbf{k}_\mathcal{A}^T \left(A_{\mathcal{A},\mathcal{A}} - A_{\mathcal{A},\mathcal{B}} D_{\mathcal{B},\mathcal{B}} A_{\mathcal{A},\mathcal{B}}\right)^{-1} \left(\boldsymbol{\gamma}_\mathcal{A} - A_{\mathcal{A},\mathcal{B}} D_{\mathcal{B},\mathcal{B}} \boldsymbol{\gamma}_\mathcal{B}\right)$$

where $\mathbf{k}_\mathcal{A}$ is a vector with elements $k(\mathbf{x}_*, \mathbf{x}_\#)$ for $\mathbf{x}_\# \in \mathcal{A}$.

D. Information filter occupancy mapping

In this section, we use the main results to design an information filter occupancy mapping algorithm (Alg. 2). Given the grid discretization and the kernel hyperparameters, the algorithm is initialized by precomputing the Toeplitz matrices $K^{(d)}$ and their inverses (line 3). Online, the distribution of the latent function $\mathbf{f}_\#$ is propagated in information form by accumulating occupancy measurements (line 4). When the occupancy states of query points $\mathbf{x}_*$ need to be recovered, the the CG method is used to compute the latent mean $\boldsymbol{\mu}$ (Alg. 1) and to obtain the label y_* according to (19) (line 7). As mentioned earlier, if $p(y_*|\mathbf{x}_*, \mathcal{D})$ is desired another call to Alg. 1 or an approximation based on Prop. 2 is needed to compute $\mathbf{k}_\#^T (K_{\#\#} + \Gamma)^{-1} \mathbf{k}_\#$. As discussed in Sec. IV-C, the occupancy state and likelihood computations can be approximated efficiently if the query set is small and localized in the same portion of the map, as it is the case for collision checking. As discussed at the end of Sec. IV-B, as long as the kernel k is decomposable, radial, and monotone decreasing, the overall memory and time complexities of Alg. 2 are $O(N)$ and $O(LMN + T)$, where T is the number

Algorithm 2 Information Filter Occupancy Mapping

Input: latent grid size $\mathbf{n} \in \mathbb{R}^D$ and resolution $\mathbf{r} \in \mathbb{R}^D$, RBF kernel amplitudes $\mathbf{b} \in \mathbb{R}^D$ and standard dev. $\mathbf{s} \in \mathbb{R}^D$, observation noise standard dev. $\sigma \in \mathbb{R}$, occupancy observations $\mathcal{D} = \{x(t), y(t)\}_{t=1}^T$
Output: occupancy states $\{y_*(i)\}$ for a set $\{\mathbf{x}_*(i)\}$ of M query points
$\left\{K_{\#\#}, \left(K_{\#\#}^{(d)}\right)^{-1}\right\}$ = INITIALIZE($\mathbf{n}$, $\mathbf{r}$, $\mathbf{b}$, $\mathbf{s}$)
$\boldsymbol{\gamma}$, Γ = ADDOBSERVATIONS($\mathcal{D}$, σ)
$\boldsymbol{\mu}$ = CONJUGATEGRADIENTMETHOD($K_{\#\#} + \Gamma$, $\boldsymbol{\gamma}$) ▷ Alg. 1
for $i = 1, \ldots, M$ **do**
  $y_*(i)$ = PREDICTOCCUPANCY($\boldsymbol{\mu}$, $\mathbf{x}_*$)
return $\{y_*(i)\}$

function INITIALIZE($\mathbf{n}$, $\mathbf{r}$, $\mathbf{b}$, $\mathbf{s}$)
  for $d = 1, \ldots, D$ **do**
    $\mathbf{a}^{(d)} := \exp\left(\frac{-\mathbf{r}^{(d)}\mathbf{r}^{(d)}}{2\mathbf{s}^{(d)}\mathbf{s}^{(d)}}\right)$
    Compute $K_{\#\#}^{(d)}$ and $\left(K_{\#\#}^{(d)}\right)^{-1}$
  $K_{\#\#} = \bigotimes_{d=1}^{D} K_{\#\#}^{(d)}$ ▷ sparse matrix

function ADDOBSERVATIONS($\mathcal{D}$, σ)
  $\boldsymbol{\gamma}_1 = \mathbf{0}$, $\Gamma_1 = \mathbf{0}$
  for $t = 1, \ldots, T$ **do**
    $\boldsymbol{\gamma}_{t+1} = \boldsymbol{\gamma}_t + \frac{\mathbf{k}_{\#,t}\, y(t)}{\lambda(\mathbf{x}(t)) + \sigma^2}$
    $\Gamma_{t+1} = \Gamma_t + \frac{\mathbf{k}_{\#,t}\mathbf{k}_{\#,t}^T}{\lambda(\mathbf{x}(t)) + \sigma^2}$

function PREDICTOCCUPANCY($\boldsymbol{\mu}$, $\mathbf{x}_*$)
  Compute $\mathbf{k}_\#$ with elements $k(\mathbf{x}_*, \mathbf{x}_\#)$ for $\mathbf{x}_\# \in \mathcal{C}$
  return 1 **if** $\mathbf{k}_\#^T\boldsymbol{\mu} > 0$ **and** -1 **otherwise**

of observations, N is the number of latent grid cells, L is the number of CG iterations for occupancy prediction, and M is the number of query points. In the worst case, if the occupancy states of the whole map are needed at each iteration, the overall time complexity would be $O(LMNT)$, which is similar to GP regression with pseudo-inputs [40]. The latent grid of IFOM allows it to scale to large maps while maintaining correlation efficiently. In the best case, when only small sets of query points need to be evaluated in local regions, IFOM allows very efficient predictive inference.

V. Evaluation

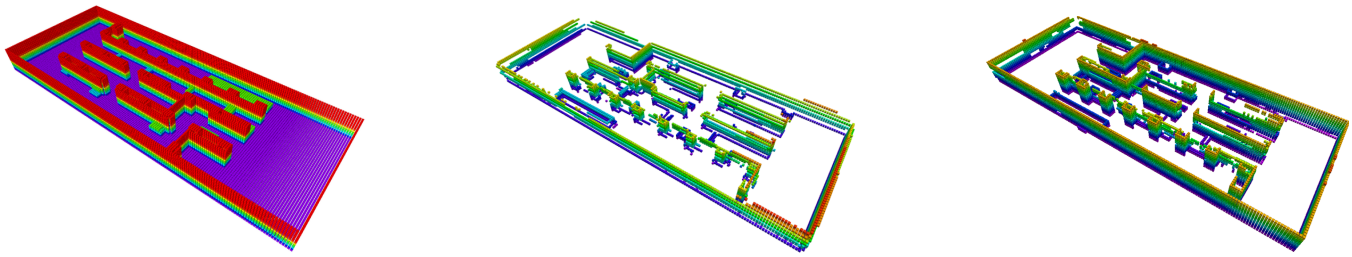
Our experiments compare the performance of IFOM (Alg. 2) and the GPOctoMap method of [17] using simulated data from a 2-D LIDAR and real data from a Microsoft Kinect depth camera. In both cases and for both methods, the Gaussian RBF kernel (3) was used. The sensor motion was assumed known and the focus was on building an occupancy map. To convert the range measurements collected by the LIDAR and depth camera to occupancy measurements $y(t) \in \{-1, 1\}$ at continuous-space points $\mathbf{x}(t) \in \mathcal{X}$, the endpoint of each sensor ray was taken as occupied, while free points were sampled using a Poisson distribution along each ray.

Simulated LIDAR data: The first set of experiments was carried out using a simulated 2-D Hokuyo LIDAR to obtain range measurements of a Gazebo environment, represented as a 3-D mesh in STL format. A ground truth occupancy grid was obtained from the mesh and is shown in Fig. 1a. The map had dimensons $85.5m \times 31.5m \times 6.5m$ and was discretized with resolution $\mathbf{r}^{(d)} = 0.5m$ in each dimension, creating a grid with size $\mathbf{n} = [171,\ 63,\ 13]^T$ and a total number of cells of $N = 140049$. We collected 5443 LIDAR scans, each providing roughly 556 occupancy measurements for a total of $T = 3.0 \times 10^6$ measurements. The kernel hyperparameters were $\mathbf{s} = [0.1,\ 0.1,\ 0.3]^T$ and $b = 1.16$, the scaling parameter was $\sigma = 0.1$. The occupancy maps recovered by GPOctoMap and IFOM are shown in Fig. 1. The time taken to obtain these results is shown in Table I. The table also shows the accuracy of the reconstructed maps evaluated with respect to the ground truth via acc $= \frac{1}{M}\sum_{i=1}^{M} \mathbb{1}_{\{q_i = \hat{q}_i\}}$, where q_i and $\hat{q}_i$ are the true and estimated occupancy states of cell i, respectively. When calculating accuracy, unobserved cells were considered free. The results show that our method is slower than GPOctoMap but results in higher accuracy. Note that the provided observations did not cover the whole map, so it was impossible to obtain a 100% accuracy. Our implementation is not yet optimized and we expect that the timing results can be improved. Recovering the whole map takes additional time for our method.

Real depth camera data: The algorithms were also evaluated on three sequences, fr1/teddy, fr3/cabinet and fr3/large_cabinet, of the TUM RGBD dataset [48] using only depth images. The depth images were filtered and downsampled after conversion to 3-D pointclouds to reduce noise. Both GPOctoMap and IFOM used the same preprocessed pointclouds. The maps had dimensions listed in Table I. Each was discretized with resolution $\mathbf{r}^{(d)} = 0.1m$ in each dimension, creating a latent grid with size $\mathbf{n} = [105,\ 105,\ 65]^T$ for fr1/teddy, $\mathbf{n} = [93,\ 77,\ 9]^T$ for fr3/cabinet, and $\mathbf{n} = [145,\ 169,\ 45]^T$ for fr3/large_cabinet. This resulted in a total number of $N = 7.2 \times 10^5$, $N = 6.4 \times 10^4$, and $N = 1.1 \times 10^6$ cells, respectively. The datasets contain about 5000 RGBD images published at 30 Hz. A depth measurement was used to update the map distribution only when the camera pose changed significantly ($0.1m$ position and 0.2 rad axis-angle change). About 220 pointclouds each providing roughly 46500 occupancy measurements were used for a total of $T = 11.5 \times 10^6$ measurements. The kernel hyperparameters were $\mathbf{s}^{(d)} = 0.15$ and $b = 1.09$, the scaling parameter was $\sigma = 0.1$. The estimated occupancy maps are shown in Fig. 2, while the computation time is shown in Table I. Fig. 2 also shows the information vector and information matrix maintained by IFOM in each case. While recovering a complete occupancy map using IFOM is slow, the information matrix can be maintained efficiently and provides a measure of uncertainty that can be used to generate sensing trajectories in the context of autonomous exploration and mapping. The figures show that the middle areas of the maps, where many observations are made, have high certainty, while frindge areas would benefit from additional observations.

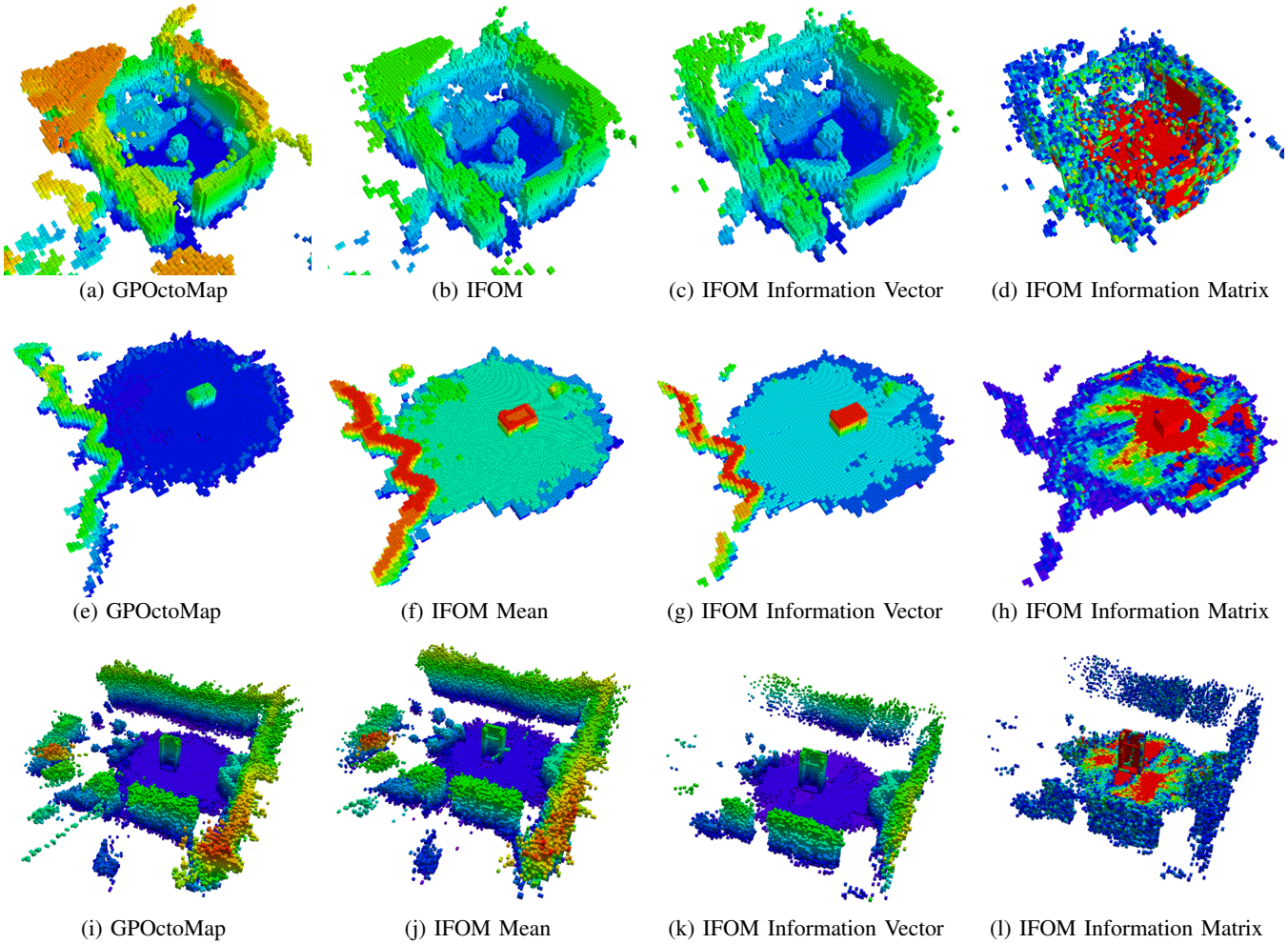
VI. Conclusion

We showed that GP regression with a grid of pseudo inputs is equivalent to information filtering and that kernel matrices corresponding to decomposable radial kernels evaluated over the grid can be computed and stored as Kronecker products of Toeplitz matrices. These observations were used to design IFOM, an efficient Bayesian occupancy mapping algorithm. Using both simulated and real data, we



(a) Ground Truth (b) GPOctoMap [17] (c) IFOM (Alg. 2)

Fig. 1: Ground truth occupancy map (a) and maps recovered by GPOctoMap (b) and IFOM (c) using 2-D LIDAR measurements in a simulated Gazebo environment. The color represents height. The map dimensions are listed in Table I.



(a) GPOctoMap (b) IFOM (c) IFOM Information Vector (d) IFOM Information Matrix

(e) GPOctoMap (f) IFOM Mean (g) IFOM Information Vector (h) IFOM Information Matrix

(i) GPOctoMap (j) IFOM Mean (k) IFOM Information Vector (l) IFOM Information Matrix

Fig. 2: Maps of 0.1m resolution recovered by GPOctoMap (a), (e), (i) and IFOM (b), (f), (j) on the TUM Teddy, Cabinet, and Large Cabinet sequences, respectively. The figures shows the information vector (c), (g), (k) and the information matrix (d), (h), (l) maintained by IFOM in each case. The values in the information matrix vary from certain (red) to uncertain (purple).

TABLE I: Comparison of the accuracy and computation time of GPOctoMap [17] and IFOM (Alg. 2)

Dataset	Map Size (m^3)	Res (m)	Algorithm	Accuracy (%)	Avg. Scan Time (s)	Avg. CG Iteration Time (s) (for map recovery)
Simulation	$85.5 \times 31.5 \times 6.5$	0.5	GPOctoMap	75.6	0.0055	0
			IFOM	79.1	0.074	0.119
TUM Teddy	$10.5 \times 10.5 \times 6.5$	0.1	GPOctoMap	—	0.039	0
			IFOM	—	0.52	0.486
TUM Cabinet	$9.3 \times 7.5 \times 0.9$	0.1	GPOctoMap	—	0.023	0
			IFOM	—	0.54	0.029
TUM Large Cabinet	$14.3 \times 16.7 \times 4.4$	0.1	GPOctoMap	—	0.15	0
			IFOM	—	0.52	0.942

demonstrated that IFOM has equivalent accuracy to state-of-the-art GP occupancy mapping techniques while providing computational and storage advantages when recovering the whole occupancy map online is not necessary. Future work will focus on map representations with adaptive and hierarchical structure and on estimating a signed distance field instead of occupancy values.

REFERENCES

[1] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera," in *ACM Sym. on User Interface Software and Technology (UIST)*, 2011, pp. 559–568.

[2] L. Teixeira and M. Chli, "Real-time mesh-based scene estimation for aerial inspection," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4863–4869.

[3] E. Piazza, A. Romanoni, and M. Matteucci, "Real-time cpu-based large-scale three-dimensional mesh reconstruction," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1584–1591, 2018.

[4] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

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

[6] A. Hornung, K. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: an efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[7] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments," *The International Journal of Robotics Research (IJRR)*, vol. 31, no. 5, pp. 647–663, 2012.

[8] T. Whelan, R. Salas-Moreno, B. Glocker, A. Davison, and S. Leutenegger, "ElasticFusion: Real-Time Dense SLAM and Light Source Estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.

[9] R. Newcombe, "Dense Visual SLAM," Ph.D. dissertation, Imperial College London, 2012.

[10] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *European Conference on Computer Vision (ECCV)*, 2014.

[11] M. Ghaffari Jadidi, J. Valls Miro, and G. Dissanayake, "Gaussian Process Autonomous Mapping and Exploration for Range Sensing Mobile Robots," *ArXiv: 1605.00335*, 2016.

[12] N. Atanasov, "Active Information Acquisition with Mobile Robots," Ph.D. dissertation, University of Pennsylvania, 2015.

[13] B. Charrow, "Information-Theoretic Active Perception for Multi-Robot Teams," Ph.D. dissertation, University of Pennsylvania, 2015.

[14] S. O'Callaghan and F. Ramos, "Gaussian process occupancy maps," *The International Journal of Robotics Research*, vol. 31, no. 1, pp. 42–62, 2012.

[15] C. Vido and F. Ramos, "From grids to continuous occupancy maps through area kernels," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016, pp. 1043–1048.

[16] S. Kim and J. Kim, *GPmap: A Unified Framework for Robotic Mapping Based on Sparse Gaussian Processes*. Springer International Publishing, 2015, pp. 319–332.

[17] J. Wang and B. Englot, "Fast, accurate gaussian process occupancy maps via test-data octrees and nested bayesian fusion," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016, pp. 1003–1010.

[18] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[19] M. Opper and O. Winther, "A Bayesian Approach to On-line Learning," in *On-line Learning in Neural Networks*, 1998, pp. 363–378.

[20] K. Sun, K. Saulnier, N. Atanasov, G. Pappas, and V. Kumar, "Dense 3-d mapping with spatial correlation via gaussian filtering," in *American Control Conference (ACC)*, 2018.

[21] T. Minka, "A family of algorithms for approximate Bayesian inference," Ph.D. dissertation, Massachusetts Institute of Technology, 2001.

[22] C. Williams and D. Barber, "Bayesian classification with Gaussian processes," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 20, no. 12, pp. 1342–1351, 1998.

[23] S. O'Callaghan, F. Ramos, and H. Durrant-Whyte, "Contextual occupancy maps using gaussian processes," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2009, pp. 1054–1060.

[24] S. O'Callaghan and F. Ramos, "Continuous Occupancy Mapping with Integral Kernels," in *AAAI Conference on Artificial Intelligence*, 2011.

[25] S. Ambikasaran, D. Foreman-Mackey, L. Greengard, D. W. Hogg, and M. O'Neil, "Fast direct methods for gaussian processes," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 38, no. 2, pp. 252–265, 2016.

[26] S. Anderson, T. Barfoot, C. H. Tong, and S. Särkkä, "Batch nonlinear continuous-time trajectory estimation as exactly sparse gaussian process regression," *Autonomous Robots*, vol. 39, no. 3, 2015.

[27] S. Kim and J. Kim, "Recursive Bayesian Updates for Occupancy Mapping and Surface Reconstruction," in *Australasian Conference on Robotics and Automation (ACRA)*, 2014.

[28] S. Kim and J. Kim, "Occupancy Mapping and Surface Reconstruction Using Local Gaussian Processes With Kinect Sensors," *IEEE Transactions on Cybernetics*, vol. 43, no. 5, pp. 1335–1346, 2013.

[29] F. Ramos and L. Ott, "Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1717–1730, 2016.

[30] R. Senanayake and F. Ramos, "Bayesian Hilbert Maps for Continuous Occupancy Mapping in Dynamic Environments," in *Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 78, 2017, pp. 458–471.

[31] V. Guizilini and F. Ramos, "Learning to Reconstruct 3D Structures for Occupancy Mapping," in *Robotics: Science and Systems*, 2017.

[32] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian Processes for Big Data," in *Conference on Uncertainty in Artificial Intelligence*, 2013.

[33] C. Cheng and B. Boots, "Variational inference for gaussian process models with linear complexity," in *Advances in Neural Information Processing Systems*, 2017.

[34] C. E. R. Joaquin Quiñonero-Candela, "A Unifying View of Sparse Approximate Gaussian Process Regression," *Journal of Machine Learning Research (JMLR)*, vol. 6, no. Dec, pp. 1939–1959, 2005.

[35] M. Titsias, "Variational Learning of Inducing Variables in Sparse Gaussian Processes," in *Int. Conf. on Artificial Intelligence and Statistics*, ser. Proc. of ML Research, vol. 5, 2009, pp. 567–574.

[36] Y. Saatci, "Scalable Inference for Structured Gaussian Process Models," Ph.D. dissertation, University of Cambridge, 2011.

[37] A. G. Wilson, C. Dann, and H. Nickisch, "Thoughts on Massively Scalable Gaussian Processes," *arXiv*, vol. 1511.01870, 2015.

[38] T. Evans and P. Nair, "Scalable Gaussian Processes with Grid-Structured Eigenfunctions (GP-GRIEF)," in *International Conference on Machine Learning*, 2018, pp. 1417–1426.

[39] Y. Gal, "Uncertainty in Deep Learning," Ph.D. dissertation, University of Cambridge, 2016.

[40] E. Snelson and Z. Ghahramani, "Sparse Gaussian Processes using Pseudo-inputs," in *Advances in Neural Information Processing Systems 18*, Y. Weiss, B. Schölkopf, and J. C. Platt, Eds., 2006, pp. 1257–1264.

[41] S. Reece and S. Roberts, "An Introduction to Gaussian Processes for the Kalman Filter Expert," in *Conference on Information Fusion*, 2010.

[42] A. G. Wilson, E. Gilboa, A. Nehorai, and J. P. Cunningham, "Fast Kernel Learning for Multidimensional Pattern Extrapolation," in *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[43] Y. Zhang, W. E. Leithead, and D. J. Leith, "Time-series Gaussian Process Regression Based on Toeplitz Computation of $O(N^2)$ Operations and $O(N)$-level Storage," in *IEEE Conference on Decision and Control (CDC)*, 2005, pp. 3711–3716.

[44] J. P. Cunningham, K. V. Shenoy, and M. Sahani, "Fast gaussian process methods for point process intensity estimation," in *International Conference on Machine Learning (ICML)*, 2008, pp. 192–199.

[45] S. Zohar, "Toeplitz Matrix Inversion: The Algorithm of W. F. Trench," *Journal of the ACM*, vol. 16, no. 4, pp. 592–601, 1969.

[46] M. Gover, "Properties of the Inverse of the Gaussian Matrix," *SIAM Journal on Matrix Analysis and Applications*, vol. 12, no. 3, 1991.

[47] M. R. Hestenes and E. Stiefel, "Methods of Conjugate Gradients for Solving Linear Systems," *Journal of Research of the National Bureau of Standards*, vol. 49, no. 6, 1952.

[48] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A Benchmark for the Evaluation of RGB-D SLAM Systems," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2012.