Dense 3-D Mapping with Spatial Correlation via Gaussian Filtering

Ke Sun, Kelsey Saulnier, Nikolay Atanasov, George J. Pappas, and Vijay Kumar

Abstract—Constructing an occupancy representation of the environment is a fundamental problem for robot autonomy. Many accurate and efficient methods exist that address this problem but most assume that the occupancy states of different elements in the map representation are statistically independent. The focus of this paper is to provide a model that captures correlation of the occupancy of map elements. Correlation is important not only for improved accuracy but also for quantifying uncertainty in the map and for planning autonomous mapping trajectories based on the correlation among known and unknown areas. Recent work proposes Gaussian Process (GP) regression to capture covariance information and enable resolution-free occupancy estimation. The drawback of techniques based on GP regression (or classification) is that the computation complexity scales cubically with the length of the measurement history. Our main contribution is a new approach for occupancy mapping that models the binary nature of occupancy measurements precisely, via a Bernoulli distribution, and provides an efficient approximation of GP classification with complexity that does not scale with time. We prove that the error between the estimates provided by our method and those provided by GP classification is negligible. The proposed method is evaluated using both simulated data and real data collected using a Velodyne Puck 3-D range sensor.

I. INTRODUCTION

Constructing a geometric representation of the environment is a fundamental problem in robotics because it is essential for localization, reliable navigation, as well as occlusion and collision checking in tasks such as object recognition and manipulation. Of course, geometric mapping has been studied extensively and there exist many accurate and efficient methods [1]–[7]. While accuracy and efficiency are important requirements, in this paper we focus on an additional requirement that many existing methods do not address. In particular, we focus on maintaining information about covariance of different elements in the map representation. For example, occupancy grid mapping [1], [8, Ch.9], one of the early, very successful approaches to geometric mapping, represents the environment as a regular grid in which each cell is a Bernoulli random variable that captures the probability of the cell being occupied or free. Occupancy grid mapping relies on the assumption that the occupancy states of different grid cells are statistically independent and sensor data is correlated only with directly observed cells. We claim that maintaining a model of occupancy dependence

The authors are with GRASP Lab, University of Pennsylvania, Philadelphia, PA 19104, USA, {`sunke, saulnier, atanasov, pappasg, kumar`}`@seas.upenn.edu`.
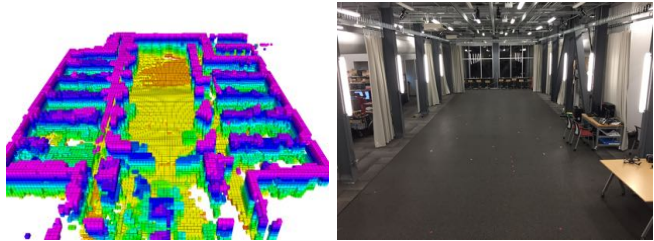
Fig. 1: A dense 3-D map (left) of a lab environment (right) produced using the occupancy grid filter, one new approach which approximates GP classification accurately with computational complexity that does not scale with time.

among the cells may improve the accuracy of the resulting maps. However, our main motivation for considering such a model with covariance information is that the correlation between known and unknown areas in the map enables planning informative robot trajectories for autonomous mapping [9]–[11]. More precisely, covariance statistics can be used to quantify the current map uncertainty (e.g., the via Shannon's entropy [12]–[14]) and to infer the probable occupancy of unobserved locations based on the occupancy of already observed cells.

The above has been recognized by several recent works [15]–[23] that propose the use of Gaussian Processes (GPs) [24] for occupancy mapping. In addition to capturing covariance information via a kernel function, these models provide resolution-free mapping in the form of a continuous occupancy function. The challenge for GP methods is to deal efficiently with the hybrid nature of the problem – discrete measurements (occupied or free) and continuous occupancy function. O'Callaghan et al. [15]–[17] proposed a probabilistic least-squares classification method in which a probit[1] function is used to convert a latent function estimated via GP regression to binary occupancy predictions. A more accurate but computationally demanding mapping method is to use GP classification [24, Ch.3.3], in which the latent function posterior is updated from the discrete occupancy measurements. Since the measurement likelihood in this case is not Gaussian, the posterior distribution cannot be analytically determined and has to be approximated with iterative methods. The major drawback of GP regression and classification is that their computational complexity is cubic in the number of measurements. In an online mapping setting where the number of measurements grows with time,

---

[1]The probit functions is defined as the culmulative distribution function of a standard normal, i.e. $\Phi(x; 0, 1) = \int_{-\infty}^{x} \phi(u; 0, 1)du$, where $\phi(u; 0, 1)$ is the standard normal pdf.

applicability of these methods is limited.

Many techniques [25]–[27] focus on improving the computational complexity of GPs in general, relying on sparse kernels and matrix factorization to enable efficient inversion of the covariance matrix (resulting from evaluating the kernel at the training examples). Specific to GP mapping, Kim and Kim [20]–[22] proposed the use of a sparse kernel and Bayesian Committee Machines to perform small GP regressions with subsets of training data before fusing them into the full map. Similarly, Wang and Englot [23] partition their sensor data amongst several GP regressions and then use Baysian Committee Machines to first combine the sensor-level regressions and then again to fuse the resulting regression with the map. The speed of this method is further improved with a new data-structure, the test-data octree, which improves access time and only considers correlations between grid cells within a specified range. Despite its advantages, this method does not capture the discrete nature of the measurements in its model and ignores some cross correlations in the grid.

Our key insight, compared to the existing work which uses GPs for metric mapping, is that the values of the latent function at a discrete set of points can be recovered exactly by a Kalman filter [28]. More precisely, the state of the Kalman filter can capture the continuous values of the latent occupancy function evaluated at each cell of a discrete map, while a probit observation model can convert these latent values into boolean measurements indicating the occupancy of the corresponding cell. The significance of this observation is that a Kalman filter can avoid the cubic scaling with the number of data points (observations of free/occupied space), while providing essentially the same estimates of the mean and covariance of the map. Our second idea is that, although the use of a probit measurement model makes the posterior distribution of the filter non-Gaussian, the posterior can be approximated via a Gaussian based on a single forward pass through the measurement data. We prove that this method results in an efficient approximation of GP classification [24, Ch.3.6]. We name the proposed approach an Occupancy Grid Filter (OGF) to emphasize that these two insights were inspired by filtering techniques. Our work makes the following contributions:

- We prove that the OGF is an accurate approximation of GP classification on a grid and show that in practice the classification error is negligible.
- The computational complexity of the OGF is constant in time while GP methods scale cubically with time as the number of measurements grows. This property makes the proposed occupancy grid filter suitable for online applications.

II. PROBLEM FORMULATION

Consider a robot whose task is to build a geometric map of a static environment. Let $\mathcal{S} \subset \mathbb{R}^3$ be a bounded connected set representing the environment and let $o : \mathcal{S} \mapsto \{-1, 1\}$ be a function such that $o(\mathbf{s}) := -1$, if $\mathbf{s} \in \mathcal{S}$ is unoccupied, and $o(\mathbf{s}) := 1$, otherwise. We restrict attention to occupancy grid

mapping [8, Ch.9]. More precisely, we consider a tessellation of $\mathcal{S}$ with an associated lattice $\mathcal{X} \subset \mathcal{S}$ and estimate the map occupancy $o(\mathbf{x})$ only at the locations $\mathbf{x} \in \mathcal{X}$. Suppose that the robot collects a sequence of occupancy measurements $Z_{0:t}$, e.g., via a laser scanner or a depth camera, along a known trajectory $(\mathbf{s}_{0:t}, R_{0:t}) \subset \mathbb{R}^3 \rtimes \mathrm{SO}(3)$. Each occupancy measurement Z_t is a matrix containing the 3-D positions of the cells $\mathbf{x}$ observed by the sensor and their occupancy $o(\mathbf{x})$, i.e., $Z_t := (X_t, \mathbf{y}_t)$, $\mathbf{x}_{i,t} \in \mathcal{X} \cap \mathrm{FoV}(\mathbf{s}_t, R_t)$, $y_{i,t} := o(\mathbf{x}_{i,t})$, and $\mathrm{FoV}(\mathbf{s}_t, R_t) \subset \mathcal{S}$ is the sensor's field of view.

Problem. *Given occupancy measurements $Z_{0:t}$, construct an estimate $\hat{o}_t \in \{-1, 0, 1\}^{|\mathcal{X}|}$ of the true environment occupancy $\{o(\mathbf{x}) | \mathbf{x} \in \mathcal{X}\}$.*

Note that one more state, 0, is added in order to allow the case that the occupancy of a cell is unknown.

III. GAUSSIAN PROCESS CLASSIFICATION

In this section, we give a brief overview of GPs and how they are applied to the occupancy mapping problem. As defined in [24], a GP is a collection of random variables, any finite number of which have a joint Gaussian distribution, represented as, incorporate

$$f(\mathbf{s}) \sim \mathcal{GP}\left(\hat{f}(\mathbf{s}), k(\mathbf{s}, \mathbf{s}')\right)$$

where $\hat{f}(\mathbf{s}) = \mathbb{E}(f(\mathbf{s}))$ is the mean of $f(\mathbf{s})$, and $k(\mathbf{s}, \mathbf{s}') = \mathbb{E}((f(\mathbf{s}) - \hat{f}(\mathbf{s}))(f(\mathbf{s}') - \hat{f}(\mathbf{s}')))$ is a kernel function used to construct the covariance matrix.

In a binary classification problem, $f(\mathbf{s})$ cannot directly model the class labels because of their discontinuous nature. Instead, $f(\mathbf{s})$ is treated as a latent function with GP prior, which is then squashed through the probit function to model the likelihood of a measurement, i.e. $p(y = 1|\mathbf{s})$ or $p(y = -1|\mathbf{s})$. The following two steps are required in a typical application of binary GP classification to mapping [16].

A. Training

In the training step, the posterior

$$p(\mathbf{f}_t | Z_{0:t}) = \frac{p(\mathbf{y}_{0:t} | \mathbf{f}_t) p(\mathbf{f}_t | S_{0:t})}{p(\mathbf{y}_{0:t} | S_{0:t})} \tag{1}$$

is used to estimate the latent values, $\mathbf{f}_t := f_{0:t}$, of all the captured samples. Note the slight abuse of notation, $Z_{0:t}$ in (1) should be more accurately represented as $Z_{0:t} := (S_{0:t}, \mathbf{y}_{0:t})$ instead of $(X_{0:t}, \mathbf{y}_{0:t})$, i.e. the training samples can be taken at any points $\mathbf{s} \in \mathcal{S}$. In (1), $p(\mathbf{f}_t | S_{0:t}) \sim \mathcal{N}(0, K)$ is the prior of the latent values of all training samples, where K is a covariance matrix constructed using a pre-defined kernel, $p(\mathbf{y}_{0:t} | S_{0:t})$ is a normalization factor independent of $\mathbf{f}_t$, and $p(\mathbf{y}_{0:t} | \mathbf{f}_t)$ is the joint likelihood function which can be represented as,

$$p(\mathbf{y}_{0:t} | \mathbf{f}_t) = \prod_i p(y_i | f_i) = \prod_i \Phi(f_i; 0, 1) \tag{2}$$

because the measurements are independent given the latent values. The posterior pdf in (1) cannot be obtained in closed form since $p(\mathbf{y}_{0:t} | \mathbf{f}_t)$ is not of normal distribution. There exist

different methods to approximate the posterior. The Laplace approximation [24, Ch.3] finds the mode of $p(\mathbf{f}_t|Z_{0:t})$ and the local negative inverse Hessian to construct a normal distribution that approximates $p(\mathbf{f}_t|Z_{0:t})$. Expectation Propagation (EP) [24, Ch.3] replaces each term in (2) with a (unnormalized) normal distribution, and tries to approximate the true posterior with product of normals in an iterative manner. More details on EP can be found in Sec. V.

B. Prediction

With the estimated latent values of the training samples, $\hat{\mathbf{f}}_t$, the latent values, $\mathbf{m}_t$, of a query position in $\mathcal{X}$ can be estimated as,

$$\hat{\mathbf{m}}_t = K^* \cdot K^{-1}\hat{\mathbf{f}}_t$$

where K^* represents the correlation between $\mathbf{m}$ and $\mathbf{f}$, which is constructed using the kernel function with query positions and the positions of the training samples. By squashing $\hat{\mathbf{m}}_t$ through the probit function, we are able to recover the occupancy of the grid,

$$\hat{o}(\mathbf{x}_i) = \begin{cases} 1, & \Phi(m_i; 0, 1) > r_o \\ -1, & \Phi(m_i; 0, 1) < r_f, \quad \forall \mathbf{x}_i \in \mathcal{X} \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

where r_o and r_f are the thresholds for determining occupied and free cells respectively.

IV. OCCUPANCY GRID FILTER

In this section, we propose a new solution to the mapping problem which we call the Occupancy Grid Filter. This method takes into account the spatial correlation and discontinuous nature of occupancy and, at the same time, has lower computational cost than GP classification methods. The OGF arises from two key ideas. First, instead of creating a continuous map which is then evaluated at query points, we preselect the query points and only maintain latent function values at those points. Second, we note that the training step of GP classification described in Sec. III is a batch optimization which must be re-evaluated when new measurements become available. We replace this with an incremental, Kalman-filter-like approach which greatly reduces the computational complexity.

By defining a grid and abstracting each cell with its center position $\mathbf{x}_i$, we only need keep track of the latent values $\mathbf{m}$ of the query positions $\mathcal{X} := \{\mathbf{x}_i\}$, where $\mathbf{m}$ can be thought as the state of the map. Following the idea of GP classification, the prior of the state is $\mathbf{m} \sim \mathcal{N}(\hat{\mathbf{m}}_0, \Sigma_0)$, where $\hat{\mathbf{m}}_0 = \mathbf{0}$ and Σ_0 is constructed by applying a kernel function on $\mathcal{X}$, i.e. $\Sigma_{0,ij} := k(\mathbf{x}_i, \mathbf{x}_j)$. Note by initializing the prior covariance with the kernel we are capturing the same correlation which would be captured by the associated GP. With the assumption that the measurement $\mathbf{z}_i := (\mathbf{x}_i^\top, y_i)$ is within the pre-defined tessellation $\mathcal{X}$, the binary measurement model can be represented as

$$y_i = h(\mathbf{x}_i) \in \{-1, 1\}$$

with the Bernoulli distribution,

$$p(y_i|\mathbf{x}_i, m_i) = \Phi(y_i m_i; 0, 1).$$

Assume that the distribution of the map follows $\mathbf{m} \sim \mathcal{N}(\hat{\mathbf{m}}_t, \Sigma_t)$ at time t. With a measurement for the ith cell at time $t+1$, the posterior distribution of the map can be computed based on Bayes' rule, i.e.,

$$p(\mathbf{m}|Z_{0:t}) = \frac{\phi(\mathbf{m}; \hat{\mathbf{m}}_t, \Sigma_t)\Phi(y_i m_i; 0, 1)}{p(\mathbf{z}_{i,t+1})}$$

Since the likelihood function is a cumulative normal function, the posterior of $\mathbf{m}$ at $t+1$ is no longer normal. To ensure the map follows a normal pdf, the posterior is estimated by minimizing the following Kullback-Leibler (KL) divergence[2]

$$\underset{\hat{\mathbf{m}}_{t+1}, \Sigma_{t+1}}{\arg\min} \ \text{KL}\Big(\phi(\mathbf{m}; \hat{\mathbf{m}}_{t+1}, \Sigma_{t+1}) \,\big\|\, p(\mathbf{m}|Z_{0:t})\Big) \quad (4)$$

The same objective function is used in [29] to enable the context-aware filter to incorporate binary measurements. We adapt the formulation of the discrete measurement update from the context-aware filter to solve (4) for the mapping problem. The mean and covariance estimation for the map at $t+1$ is

$$\hat{\mathbf{m}}_{t+1} = \hat{\mathbf{m}}_t + y_i \cdot \Sigma_t \mathbf{v}_i \frac{\phi\left(y_i \cdot \frac{\mathbf{v}_i^\top \hat{\mathbf{m}}_t}{\sqrt{\mathbf{v}_i^\top \Sigma_t \mathbf{v}_i + 1}}; 0, 1\right)}{\eta_{t+1}\sqrt{\mathbf{v}_i^\top \Sigma_t \mathbf{v}_i + 1}}$$

$$\Sigma_{t+1} = \Sigma_t - (\hat{\mathbf{m}}_{t+1} - \hat{\mathbf{m}}_t)(\hat{\mathbf{m}}_{t+1} - \hat{\mathbf{m}}_t)^\top - \quad (5)$$

$$y_i \cdot \Sigma_t \mathbf{v}_i \mathbf{v}_i^\top \Sigma_t^\top \frac{\phi\left(y_i \cdot \frac{\mathbf{v}_i^\top \hat{\mathbf{m}}_t}{\sqrt{\mathbf{v}_i^\top \Sigma_t \mathbf{v}_i + 1}}; 0, 1\right)\left(\mathbf{v}_i^\top \hat{\mathbf{m}}_t\right)}{\eta_{t+1}\left(\mathbf{v}_i^\top \Sigma_t \mathbf{v}_i + 1\right)^{3/2}}$$

where $\mathbf{v}_i$ is a selection vector containing all 0s but a single 1 corresponding to the ith cell, η_{t+1} is the normalization factor,

$$\eta_{t+1} = p(\mathbf{z}_{i,t+1}) = \Phi\left(y_i \cdot \frac{\mathbf{v}_i^\top \hat{\mathbf{m}}_t}{\sqrt{\mathbf{v}_i^\top \Sigma_t \mathbf{v}_i + 1}}; 0, 1\right)$$

The occupancy of each cell can then be recovered through (3). Unlike GP classification, the occupancy grid filter merges both training and prediction steps, and incrementally updates the map only keeping track of the latent values on the grid.

V. THEORETICAL PROPERTIES OF OGF

A mentioned in Sec. III, GP classification cannot be solved in closed form. In this section we briefly describe one of the conventional iterative solutions, EP, in the context of the mapping problem. We then show that the occupancy grid filter introduced in Sec. IV is a streamwise approximation of EP and can solve the occupancy mapping problem with computation complexity linear in the number of measurements at each time step and quadratic in the size of the map. The idea of EP is to approximate the posterior distribution of $\mathbf{m}$, $p(\mathbf{m}|Z_{0:t})$, by

$$q(\mathbf{m}|Z_{0:t}) = \frac{1}{\eta_{ep}} p(\mathbf{m}|X_{0:t}) \prod_i t_i(m_i|\tilde{\eta}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2),$$

---

[2]The KL operator is defined as $\text{KL}(p||q) = \int p(x) \log(p(x)/q(x))dx$. Note that the KL operator is not associative. The order in (6) is chosen so that (6) can be solved analytically.

Algorithm 1 Expectation Propagation

Initialize $\tilde{\mu}_i = 0$, $\tilde{\sigma}_i = \infty$, and $\tilde{\eta}_i = 1$.
repeat
    for all i **do**
        Find the cavity parameters for m_i with $q_{-i}(m_i)$.
        Update the site parameters for t_i using (7) and (8).
        Update the mean and covariance for $q(\mathbf{m}|Z_{0:t})$.
until convergence

---

where η_{ep} is a normalization factor and

$$t_i(m_i|\tilde{\eta}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) = \tilde{\eta}_i \phi(m_i; \tilde{\mu}_i, \tilde{\sigma}_i^2),$$

is an unnormalized normal pdf. For convenience, we follow the naming convention in [24, Ch.3] calling $\tilde{\mu}$, $\tilde{\sigma}$, and $\tilde{\eta}$ site parameters. Conceptually, EP approximates the original posterior distribution with a product of normals, which itself is also a normal pdf. The site parameters are estimated iteratively, and the final approximation, $q(\mathbf{m}|Z_{0:t})$, can be easily recovered by combining each term after convergence. At each iteration, the cavity distribution of m_i, defined as

$$q_{-i}(m_i) = \int_{j,j \neq i} q \cdot t_i^{-1} dm_j \sim \mathcal{N}(\mu_{-i}, \sigma_{-i}^2),$$

is found with μ_{-i} and σ_{-i} called cavity parameters. Then the site parameters for t_i are updated by minimizing the KL divergence between $q_{-i}(m_i)t_i$ and $q_{-i}(m_i)\Phi(y_i m_i)$, i.e.

$$\underset{\tilde{\eta}_i, \tilde{\mu}_i, \tilde{\sigma}_i}{\arg\min} \mathrm{KL}\Big(q_{-i}(m_i)t_i \,\big\|\, q_{-i}(m_i)\Phi(y_i m_i; 0, 1)\Big) \quad (6)$$

Since $q_{-i}(m_i)t_i$ is in the exponential family, (6) can be solved by matching the first two moments of the distributions. As given in [24, Ch.3], the solution to (6) is,

$$z_i = \frac{y_i \mu_{-i}}{\sqrt{1 + \sigma_{-i}^2}}, \quad \hat{\eta}_i = \Phi(z_i; 0, 1)$$

$$\hat{\mu}_i = \mu_{-i} + \frac{y_i \sigma_{-i}^2 \phi(z_i; 0, 1)}{\hat{\eta}_i \sqrt{1 + \sigma_{-i}^2}} \quad (7)$$

$$\hat{\sigma}_i^2 = \sigma_{-i}^2 - \frac{\sigma_{-i}^4 \phi(z_i; 0, 1)}{(1 + \sigma_{-i}^2)\hat{\eta}_i}\left(z_i + \frac{\phi(z_i; 0, 1)}{\hat{\eta}_i}\right)$$

and,

$$\tilde{\sigma}_i^2 = \left(\hat{\sigma}_i^{-2} - \sigma_{-i}^{-2}\right)^{-1}, \quad \tilde{\mu}_i = \tilde{\sigma}_i^2\left(\hat{\sigma}_i^{-2}\hat{\mu}_i - \sigma_{-i}^{-2}\mu_{-i}\right)$$

$$\tilde{\eta}_i = \hat{\eta}_i \sqrt{2\pi\left(\sigma_{-i}^2 + \tilde{\sigma}_i^2\right)} \exp\left(\frac{(\mu_{-i} - \tilde{\mu}_i)^2}{2\left(\sigma_{-i}^2 + \tilde{\sigma}_i^2\right)}\right) \quad (8)$$

In (7), $\hat{\mu}_i$ and $\hat{\sigma}_i^2$ are the first and second moments of the distribution $p_{-i}(m_i)\Phi(y_i m_i; 0, 1)$. A complete description of EP can be found in Algorithm 1.

A. Accuracy

Theorem 1. *Given a prior of a random variable* $\mathbf{m}$ *with* $\mathbf{m} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ *and a measurement* $y_i \in \{-1, 1\}$ *with Bernoulli likelihood* $y_i \sim \mathcal{B}(\Phi(y_i m_i; 0, 1))$. *If the posterior of* $\mathbf{m}$, *i.e.* $\phi(\mathbf{m}; \boldsymbol{\mu}, \Sigma)\Phi(y_i m_i; 0, 1)$ *(the normalization factor is ignored here and onwards in this theorem), is to be approximated*

by another normal pdf $\mathbf{m} \sim \mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\Sigma})$, *the following two approaches will give the same results.*

$$\min_{\hat{\boldsymbol{\mu}}_c, \hat{\Sigma}_c} KL\Big(\phi(\mathbf{m}; \hat{\boldsymbol{\mu}}_c, \hat{\Sigma}_c) \,\big\|\, \phi(\mathbf{m}; \boldsymbol{\mu}, \Sigma)\Phi(y_i m_i; 0, 1)\Big) \quad (9)$$

and,

$$\min_{\tilde{\mu}_i, \tilde{\sigma}_i} KL\Bigg(\int_{m_{j, j \neq i}} \phi(\mathbf{m}; \boldsymbol{\mu}, \Sigma)\phi(m_i; \tilde{\mu}_i, \tilde{\sigma}_i^2)dm_j \,\Big\| \\ \int_{m_{j, j \neq i}} \phi(\mathbf{m}; \boldsymbol{\mu}, \Sigma)\Phi(y_i m_i; 0, 1)dm_j\Bigg) \quad (10)$$

i.e.

$$\phi(\mathbf{m}; \hat{\boldsymbol{\mu}}_c, \hat{\Sigma}_c) \sim \phi(\mathbf{m}; \boldsymbol{\mu}, \Sigma)\phi(m_i; \tilde{\mu}_i, \tilde{\sigma}_i^2)$$

Proof. Recall that (9) is in the same form of (4), which is the objective function of the occupancy grid filter at the measurement update step. Therefore, the solutions for $\hat{\boldsymbol{\mu}}_c$ and $\hat{\Sigma}_c$ follows the form of (5).

One critical observation for (10) is that the approximated distribution $\phi(\mathbf{m}; \boldsymbol{\mu}, \Sigma)\phi(m_i; \tilde{\mu}_i, \tilde{\sigma}_i^2)$ can be thought of as the product of a Gaussian prior and a measurement with a normal likelihood. The measurement model is this case is linear and can be represented as,

$$h_i(\mathbf{m}) = \mathbf{v}_i^\top \mathbf{m} \sim \mathcal{N}(\tilde{\mu}_i, \tilde{\sigma}_i^2),$$

where $\mathbf{v}_i$ is defined similarly as in (5). Then the resulting posterior distribution $\mathcal{N}(\hat{\boldsymbol{\mu}}_e, \hat{\Sigma}_e)$ can be readily computed by following the measurement update step of a Kalman filter,

$$K = \Sigma\mathbf{v}_i\left(\mathbf{v}_i^\top \Sigma\mathbf{v}_i + \tilde{\sigma}_i^2\right)^{-1}$$
$$\hat{\boldsymbol{\mu}}_e = \boldsymbol{\mu} + K\left(\tilde{\mu}_i - \mathbf{v}_i^\top \boldsymbol{\mu}\right) \quad (11)$$
$$\hat{\Sigma}_e = (I - K\mathbf{v}_i^\top)\Sigma$$

Using the results for $\tilde{\mu}_i$ and $\tilde{\sigma}_i$ in (7) and (8),

$$\hat{\boldsymbol{\mu}}_c = \hat{\boldsymbol{\mu}}_e, \quad \hat{\Sigma}_c = \hat{\Sigma}_e$$

i.e. the two optimization methods result in the same approximation for posterior of $\mathbf{m}$. The details for computing $\hat{\boldsymbol{\mu}}_e$ and $\hat{\Sigma}_e$ can be found in Appendix A. $\qquad \square$

Corollary 1.1. *The proposed method is a streamwise approximation of Expectation Propagation to solve Gaussian process occupancy mapping, where "streamwise" means processing the measurements sequentially without iterations.*

Proof. We prove this corollary by applying Mathematical Induction. The initial prior for OGF is $\mathbf{m} \sim \mathcal{N}(\hat{\mathbf{m}}_0, \Sigma_0)$ where $\hat{\mathbf{m}}_0 = \mathbf{0}$ and $\Sigma_0 = K$ is constructed by the kernels as in Gaussian processes. The initial approximation provided EP is also $\mathbf{m} \sim \mathcal{N}(\mathbf{0}, K)$, since $t_i = 1$ for all i given the initial settings in Algorithm 1. Assume after processing the ith measurement at time step t, both OGF and EP have an approximation for the posterior of $\mathbf{m}$ with $\mathbf{m} \sim \mathcal{N}(\hat{\mathbf{m}}_t, \hat{\Sigma}_t)$. With the $(i + 1)^{\text{th}}$ measurement at time $t + 1$, the OGF and EP generates new approximations for the posterior with $\mathbf{m} \sim \mathcal{N}(\hat{\mathbf{m}}_{t+1,c}, \hat{\Sigma}_{t+1,c})$ and $\mathbf{m} \sim \mathcal{N}(\hat{\mathbf{m}}_{t+1,e}, \hat{\Sigma}_{t+1,e})$ by solving (6) and (4) respectively. Note the correspondence between (6) and (10), as well as (4) and (9). A direct

application of Theorem 1 shows that $\phi(\mathbf{m}; \hat{\mathbf{m}}_{t+1,c}, \hat{\Sigma}_{t+1,c}) = \phi(\mathbf{m}; \hat{\mathbf{m}}_{t+1,e}, \hat{\Sigma}_{t+1,e})$. Therefore, OGF is a streamwise approximation of the EP algorithm. □

Conceptually, EP can be considered as a smoothing method requiring multiple iterations to converge. The proposed OGF is its corresponding filter approach which process each measurement only once. Assume both EP[3] and OGF converge in the sense that with new measurements, the estimated mean $\hat{\mathbf{m}}$ and uncertainty $\hat{\Sigma}$ no longer changes. Based on (5), (7), and (8), this happens if and only if for every cell in the map, $\phi(m_i; 0, 1) = 0$, which implies all cells are well classified. In practice, m_i may converge to different values for EP and OGF because the gradient of the normal pdf approaches 0 out of the three deviations. However, the difference should not affect the final classification result since $\Phi(m_i; 0, 1)$ is almost at its extreme values when $\phi(m_i; 0, 1) \to 0$. This behavior will be more clear with the experiment results in the 2-D simulation environment in Sec. VI-A.

B. Complexity

Representing the map size as N, we consider the computational complexity at time step T with the assumption that a constant number of measurements, M, is taken at each time step. Then we have the following remark about the computation complexity.

Remark 1. *For a single time step, the computation complexities for Laplace approximation, EP, and OGF are $\mathcal{O}(s_n M^3 T^3 + N M^2 T^2)$, $\mathcal{O}(s_e M^3 T^3 + N M^2 T^2)$, and $\mathcal{O}(M N^2)$ respectively.*

The cubic complexity of Laplace approximation comes from the inversion of the Hessian in Newton's method, and s_n is the average number of iterations for Newton's method to converge. For EP, the cubic complexity results from the iterative update where the update for the site parameters of each t_i has complexity $\mathcal{O}(M^2 T^2)$. For OGF, no prediction step is required. The computation complexity only results from the measurement update. A direct observation is that, unlike conventional methods, the computation complexity of our approach does not scale with time. Also, in practical applications, the total number of measurements over time is usually in the same order with the dimension of the map. In this case, OGF has much lower computation complexity compared to Laplace approximation and EP.

VI. Experimental Results

Two kinds of experimental results are shown in this section. In Section VI-A, we compare the accuracy of OGF with EP and the occupancy grid mapping algorithm [1] in a 2-D simulated environment. In Section VI-B, we compare the maps built using OGF and Octomap [2] using 3-D point cloud measurements from a Velodyne. In both simulated and real world experiments, we used the standard normal pdf as the kernel function, i.e.

$$k(\mathbf{x}, \mathbf{x}') = \phi(\|\mathbf{x} - \mathbf{x}'\|_2; 0, \sigma^2), \quad \mathbf{x}, \mathbf{x}' \in \mathcal{X}$$

[3]As has been mentioned in [24], EP is not guaranteed to converge. However, EP shows good convergence property in practical application.

A. 2-D Simulation Environment

We compare the proposed OGF with EP and occupancy grid mapping algorithm in a 2-D simulated environment with known ground truth. Noise-free samples were taken randomly from the map without repetition. The 2-D ground truth map and an example of 300 random samples are shown in Figure 2.
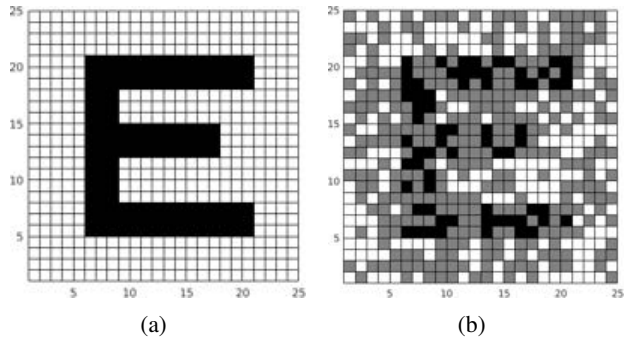

(a)   (b)

Fig. 2: (a) Ground truth map of dimension 25×25 used in the 2-D simulation. (b) 300 samples were taken randomly from the map in (a) without repetition.

Figure 3 shows the map estimated by both EP and OGF, as well as the difference between the two maps, with the 300 measurements shown in Figure 2b. As can be observed directly in Figure 3c, the differences were mainly concentrated at the well-classified regions where the absolute latent values are relatively large. This echoes our analysis in Section V.

We used (12) to convert the difference of the latent values into a scalar metric, which is the difference as a percentage of the map estimated by EP.

$$d(\hat{\mathbf{m}}_c, \hat{\mathbf{m}}_e) = \frac{\|\hat{\mathbf{m}}_c - \hat{\mathbf{m}}_e\|_2}{\|\hat{\mathbf{m}}_e\|_2} \tag{12}$$

Figure 3d shows $d(\hat{\mathbf{m}}_c, \hat{\mathbf{m}}_e)$ with respect to the number of measurements. Based on the plot, the difference $d(\hat{\mathbf{m}}_c, \hat{\mathbf{m}}_e)$ was generally less than $0.04\|\hat{\mathbf{m}}_e\|_2$. It can also be observed that $d(\hat{\mathbf{m}}_c, \hat{\mathbf{m}}_e)$ increased with the number of measurements, which was due to the fact that more cells in the map should be well-classified as the measurement increases. As shown in the later examples, this minor error in latent values does not affect the classification results.

Figure 4a shows the accuracy of the map generated by the occupancy grid mapping algorithm, EP, and OGF with the accuracy defined as,

$$acc = \frac{1}{|\mathcal{X}|} \sum_i \mathbf{1}(\hat{o}(\mathbf{x}_i) = o(\mathbf{x}_i)), \quad \mathbf{x}_i \in \mathcal{X} \tag{13}$$

where $|\mathcal{X}|$ is the map size, while $o(\mathbf{x})$ and $\hat{o}(\mathbf{x})$ are defined in Sec. II and III. It should be noted that occupancy grid mapping was trivial in this case; since the measurements were noise-free, the map generated by the occupancy grid algorithm was the same as marking sample positions as whatever the measurements were.

As can be seen from Figure 4a, the map generated using the proposed method has the same accuracy as EP for various numbers of measurements. This supports our previous claim
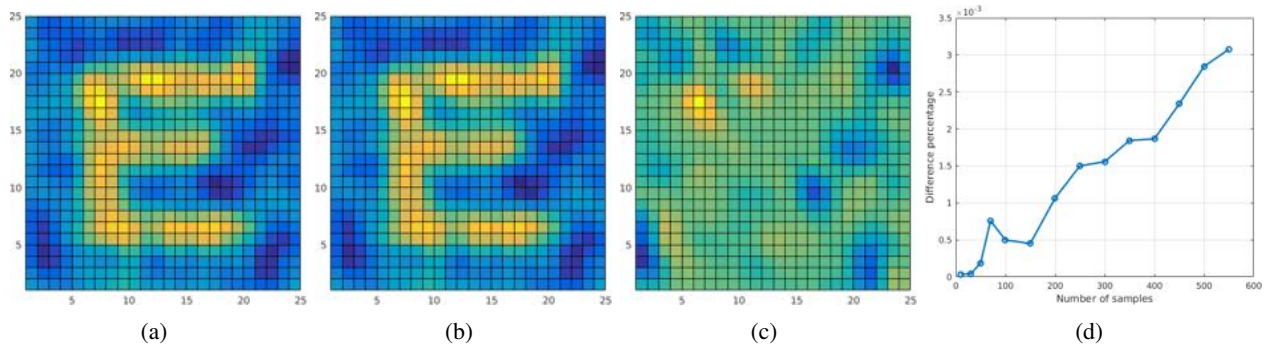
Fig. 3: (a) Map difference percentage $d(\hat{\mathbf{m}}_c, \hat{\mathbf{m}}_e)$, as defined in (12), with respect to the number of samples. (b) Map estimated with EP. (c) Map estimated OGF. (d) Difference between the two estimated maps. Both of the maps were estimated using the 300 training samples shown in Figure 2b with the standard deviation of the normal pdf kernel set to 1 grid unit.
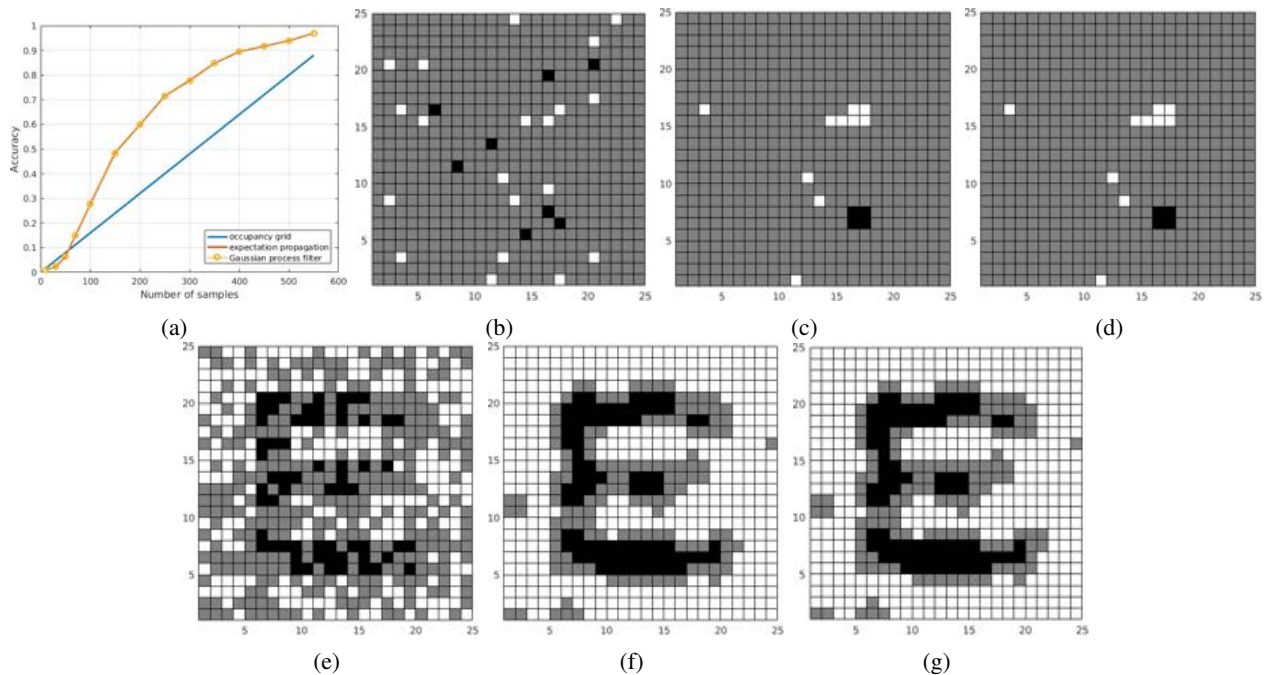

Fig. 4: (a) The accuracy, as defined in (13), of the map generated by the occupancy grid mapping algorithm, EP, and OGF. (b), (c), and (d) are the maps generated by the three algorithms with 30 samples. (e), (f), and (g) are the maps generated by the same algorithms with 300 samples. For EP and OGF, r_o and r_f were set to 0.65 and 0.35 respectively.

that the minor difference in latent value estimation does not affect the classification. It is also indicated in Figure 4a that both EP and OGF have lower accuracy than the occupancy grid mapping when there are few samples, but outperform it once the samples get reasonably dense. The reason for this is that sparse measurements will not alter the latent value of a cell far from its prior mean so the occupancy status of most of the cells remain unknown. However, when the density of samples increases the OGF is able to infer the occupancy of the cells in the neighborhood of the measured samples. The required density depends on the type and the parameters of the kernel in use. Figure 4 shows the map generated by the three algorithms with 30 and 300 samples respectively.

B. Real World Data

In order to evaluate the effectiveness of OGF with real world data we implemented the proposed algorithm using

C++ with ROS[4]. A Velodyne Puck (VLP-16)[5] was used to provide 3-D point cloud measurements. The lidar-based odometry method (LOAM) [30] was used to estimate the position and orientation of the sensor.

To enable the algorithm to deal with the dense point clouds provided by Velodyne, we applied the following downsampling method that reduced the number of samples taken at each measurement update step, but at the same time kept all the samples uniformly distributed across the 3-D space. For each lidar ray, we used the ray tracing algorithm introduced in [31] to identify which cells that ray has travelled through. A measurement was only taken in a cell if no measurement had been taken within this cell before. As has been mentioned in [2], a cell measured as free by one lidar ray may be

[4]www.ros.org
[5]velodynelidar.com/vlp-16.html
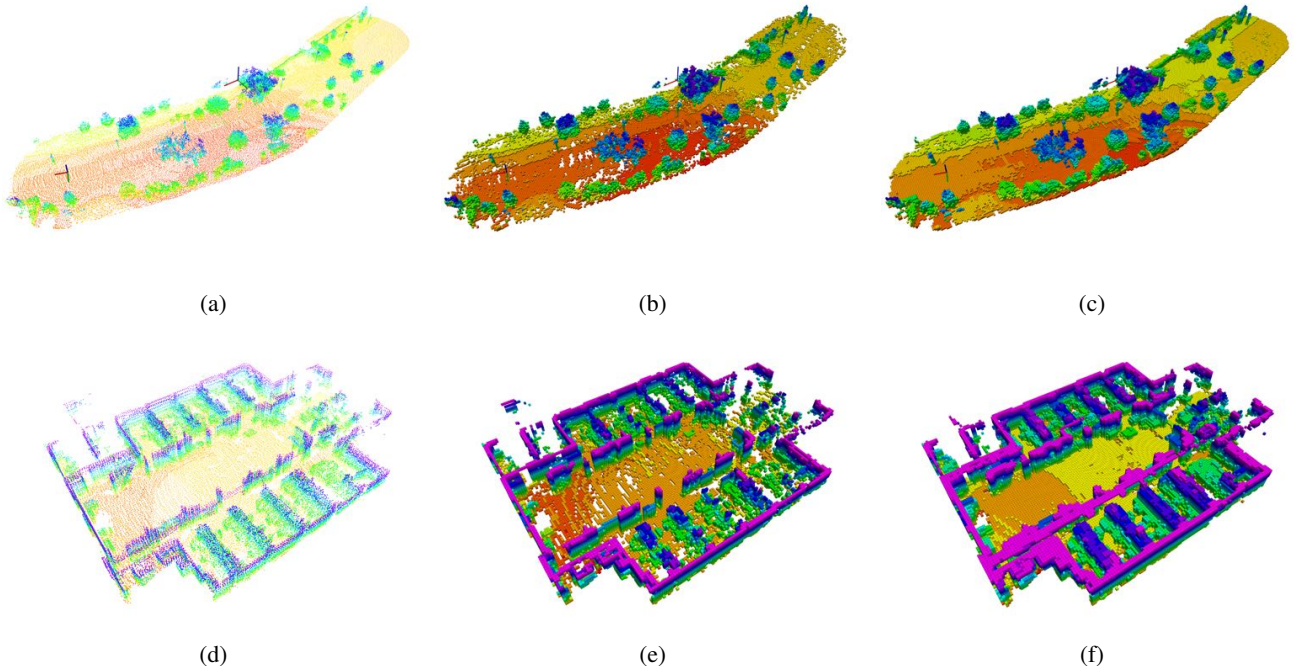
|(a)|(b)|(c)|
|(d)|(e)|(f)|

Fig. 5: The first column is the raw point cloud directly registered using the Velodyne measurements. The point cloud was filtered with the PCL voxel grid filter with $0.5m$ leaf size. The second and third columns are the occupied cells in the map generated by the Octomap and OGF respectively. The dataset of the first row was taken in a outdoor environment with dimension $150m \times 60m \times 8m$. In the outdoor dataset, the cell size used for both Octomap and the OGF was $0.4m$. The second row shows a dataset from an indoor environment with dimension $35m \times 30m \times 2m$ and cell size $0.2m$. In both datasets, the standard deviation used in the kernel function was defined as half the size of the corresponding cell, i.e. $0.2m$ and $0.1m$. The threshold values, r_o and r_f, were set to 0.65 and 0.35 respectively.

reported as occupied in others due to the discretization of the 3-D space. This effect is especially obvious when the map resolution is coarse. To overcome this problem, we prioritized occupied measurements of a cell over free measurements of the same cell, i.e a cell previously measured as free could be re-measured as occupied, but the opposite was not allowed.

Figure 5 shows a comparison between the map generated by Octomap and OGF in both outdoor and indoor environments. In the experiment with real world data, OGF demonstrated its capability of generating a denser map than Octomap by filling the gaps, which is especially obvious comparing the reconstructed ground.

VII. CONCLUSION

This paper presented an incremental approach for solving the occupancy estimation problem. We proved that the proposed Occupancy Grid Filter is a streamwise approximation of EP, one of the conventional methods to solve GP classification, with constant computation complexity in time. With the 2-D simulation environment, we demonstrated that OGF generates almost the same classification results as EP, albeit with minor differences in the latent values only for the well-classified cells. Also, the accuracy outperforms the occupancy grid mapping algorithm with reasonable number of measurements. We also evaluated the performance of OGF on real world data, and compared the results with Octomap. It was demonstrated that, because of the more accurate

representation of the uncertainty of the map, OGF is able to fill the gaps and produce a map that is reasonably denser.

As the computation complexity of our method is quadratic in the map size, for the future work, we would like to explore the possibility of developing a map representation with an adaptive resolution that is compatible with the current framework. Furthermore, we would like to understand if the covariance information provided by OGF, which drops the independence assumption between the cells in the grid, can be a better guidance for autonomous exploration and mapping compared to the classical occupancy grid map.

APPENDIX A
COMPLEMENTARY PROOF FOR THEOREM 1

In this section, we complete the intermediate steps of (11), which is the measurement update step of a Kalman filter with prior $\mathbf{m} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ and a linear measurement model $h_i(\mathbf{m}) = \mathbf{v}_i^\top \mathbf{m} \sim \mathcal{N}(\tilde{\mu}_i, \tilde{\sigma}_i^2)$. Define,

$$\Delta \boldsymbol{\mu}_e := \hat{\boldsymbol{\mu}}_e - \boldsymbol{\mu}, \qquad \Delta \Sigma_e := \hat{\Sigma}_e - \Sigma$$

Then change of mean is,

$$\Delta \boldsymbol{\mu}_e = \Sigma \mathbf{v}_i \cdot \frac{\tilde{\mu}_i - \mu_i}{\sigma_i^2 + \tilde{\sigma}_i^2}$$

Using the results from (7) and (8), $\Delta \boldsymbol{\mu}_e$ can be computed as in (14). Note that μ_i and σ_i are used in (14) to represent the

cavity parameters instead of μ_{-i} and σ_{-i}.

$$\Delta\boldsymbol{\mu}_e = \Sigma\mathbf{v}_i \cdot \frac{\tilde{\sigma}_i^2\left(\hat{\sigma}_i^{-2}\hat{\mu}_i - \sigma_i^{-2}\mu_i\right) - \mu_i}{\sigma_i^2 + \tilde{\sigma}_i^2}$$

$$= \Sigma\mathbf{v}_i \cdot \frac{\hat{\sigma}_i^{-2}\hat{\mu}_i - \sigma_i^{-2}\mu_i - \tilde{\sigma}_i^{-2}\mu_i}{\sigma_i^2\tilde{\sigma}_i^{-2} + 1}$$

$$= \Sigma\mathbf{v}_i \cdot \frac{\hat{\sigma}_i^{-2}\hat{\mu}_i - \left(\sigma_i^{-2} + \tilde{\sigma}_i^{-2}\right)\mu_i}{\sigma_i^2\left(\hat{\sigma}_i^{-2} - \sigma_i^{-2}\right) + 1} \qquad (14)$$

$$= \Sigma\mathbf{v}_i \cdot \frac{\hat{\sigma}_i^{-2}\hat{\mu}_i - \hat{\sigma}_i^{-2}\mu_i}{\sigma_i^2\hat{\sigma}_i^{-2}}$$

$$= \Sigma\mathbf{v}_i \cdot \frac{\hat{\mu}_i - \mu_i}{\sigma_i^2}$$

$$= \Sigma\mathbf{v}_i \cdot \frac{y_i\phi(z_i; 0, 1)}{\Phi(z_i; 0, 1)\sqrt{1 + \sigma_i^2}}$$

Similarly, the change of covariance is,

$$\Delta\Sigma_e = -\Sigma\mathbf{v}_i\mathbf{v}_i^\top\Sigma \cdot \frac{1}{\sigma_i^2 + \tilde{\sigma}_i^2}$$

$$= -\Sigma\mathbf{v}_i\mathbf{v}_i^\top\Sigma \cdot \frac{\tilde{\sigma}_i^{-2}}{\sigma_i^2\tilde{\sigma}_i^{-2} + 1}$$

$$= -\Sigma\mathbf{v}_i\mathbf{v}_i^\top\Sigma \cdot \frac{\hat{\sigma}_i^{-2} - \sigma_i^{-2}}{\sigma_i^2\left(\hat{\sigma}_i^{-2} - \sigma_i^{-2}\right) + 1}$$

$$= -\Sigma\mathbf{v}_i\mathbf{v}_i^\top\Sigma \cdot \frac{\hat{\sigma}_i^{-2} - \sigma_i^{-2}}{\sigma_i^2\hat{\sigma}_i^{-2}}$$

$$= -\Sigma\mathbf{v}_i\mathbf{v}_i^\top\Sigma \cdot \frac{\phi(z_i; 0, 1)}{(1 + \sigma_i^2)\Phi(z_i; 0, 1)}\left(z_i + \frac{\phi(z_i; 0, 1)}{\Phi(z_i; 0, 1)}\right)$$

$$= -\Sigma\mathbf{v}_i\mathbf{v}_i^\top\Sigma \cdot \frac{y_i\mu_i \cdot \phi(z_i; 0, 1)}{(1 + \sigma_i^2)^{3/2}\Phi(z_i; 0, 1)} - \Delta\boldsymbol{\mu}_e\Delta\boldsymbol{\mu}_e^\top$$

The change of mean and covariance matches the measurement update step in the occupancy grid filter in (5).

REFERENCES

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

[2] 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.

[3] 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.

[4] 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.

[5] 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.

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

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

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

[9] M. Ghaffari Jadidi, J. Valls Miro, R. Valencia, and J. Andrade-Cetto, "Exploration on continuous Gaussian process frontier maps," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 6077–6082.

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

[11] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, and V. Kumar, "Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping," in *Robotics: Science and Systems*, 2015.

[12] A. Krause, "Optimizing Sensing," Ph.D. dissertation, Carnegie Mellon University, December 2008.

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

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

[15] 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.

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

[17] ——, "Continuous Occupancy Mapping with Integral Kernels," 2011.

[18] F. Ramos and L. Ott, "Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent," in *Robotics: Science and Systems*, 2015.

[19] 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.

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

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

[22] ——, *GPmap: A Unified Framework for Robotic Mapping Based on Sparse Gaussian Processes*. Springer International Publishing, 2015, pp. 319–332.

[23] 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.

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

[25] S. Ambikasaran, D. Foreman-Mackey, L. Greengard, D. Hogg, and M. O'Neil, "Fast Direct Methods for Gaussian Processes," *ArXiv:1403.6015*, 2014.

[26] 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.

[27] 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, pp. 221–238, 2015.

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

[29] R. Ivanov, N. Atanasov, M. Pajic, G. Pappas, and I. Lee, "Robust estimation using context-aware filtering," in *Allerton Conference on Communication, Control, and Computing*, Sept 2015, pp. 590–597.

[30] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, 2014.

[31] J. Amanatides, A. Woo *et al.*, "A fast voxel traversal algorithm for ray tracing," in *Eurographics*, vol. 87, no. 3, 1987, pp. 3–10.