Building Continuous Occupancy Maps with Moving Robots

Ransalu Senanayake and Fabio Ramos
School of Information Technologies, The University of Sydney, Australia

Abstract

Mapping the occupancy level of an environment is important for a robot to navigate in unknown and unstructured environments. To this end, continuous occupancy mapping techniques which express the probability of a location as a function are used. In this work, we provide a theoretical analysis to compare and contrast the two major branches of Bayesian continuous occupancy mapping techniques—Gaussian process occupancy maps and Bayesian Hilbert maps—considering the fact that both utilize kernel functions to operate in a rich high-dimensional implicit feature space and use variational inference to learn parameters. Then, we extend the recent Bayesian Hilbert maps framework which is so far only used for stationary robots, to map large environments with moving robots. Finally, we propose convolution of kernels as a powerful tool to improve different aspects of continuous occupancy mapping. Our claims are also experimentally validated with both simulated and real-world datasets.

Introduction

Spatial awareness is of fundamental importance for artificial intelligent systems to operate safely and robustly in real world applications. The recent development of autonomous cars, which is revolutionizing transportation and urban design practices, brings unprecedented need for accurate, robust, fast, and, adaptive spatial representations that can be directly utilized in decision making systems. One type of such spatial representation that has been very popular in robotics is occupancy mapping, in which points in 2D or 3D space receive a label on whether or not it is occupied by an object. The seminal work by Elfes (Elfes 1989) introduced the occupancy grid map which discretizes the environment into cells and computes a probability distribution on the occupancy level. Due to its simplicity, occupancy grid maps have been the standard spatial representation particularly for indoor environments where the dimensions of the space to be mapped are known in advance. However, strong independent assumptions between the occupancy level of each cell and the need to define the level of discretization or resolution of the map at the beginning of the mapping process constitute major limitations for the technique.

In a different approach, occupancy mapping has been treated as a supervised learning problem in Gaussian process occupancy maps (GPOMs) (O'Callaghan and Ramos 2014) and Hilbert maps (HMs) (Ramos and Ott 2015). Both of these representations are continuous and exploit the spatial proximity of occupied and non-occupied observations, usually from a laser range finder, to predict the occupancy property of unseen regions. Because the representations are continuous they can be applied to indoor and outdoor environments. Common to both methods is the use of kernels (Smola and Schlkopf 2003; Kivinen, Smola, and Williamson 2014) to capture spatial closeness between observations. Another important advantage of continuous occupancy maps is the fact that they provide a functional representation of space where derivatives with respect to model parameters and occupancy levels are directly available. This is important for simultaneous localization and mapping problems (Stachniss, Leonard, and Thrun 2016) where the robot needs to localize within a map being constructed, and path planning (Norouzii, Miro, and Dissanayake 2016) where the robot attempts to find a trajectory that minimizes the chance of collisions (Marinho et al. 2016) (Mukadam et al. 2017).

Within the context of these recent developments, the paper brings the following contributions:

1. An analysis of Bayesian Hilbert maps (BHMs) and Gaussian process occupancy maps considering the fact that both use kernels and variational inference;
2. The use of convolution of kernels in robotic mapping;
3. Proposing the BHMs framework to map the occupancy of large environments using moving robots.

The paper is organized as follows. Firstly, we provide an overview of continuous occupancy mapping techniques. Then, in the next section, the BHMs framework is introduced for moving robots to map large areas. We discuss the similarities and differences between BHMs and GPOMs highlighting that both are Bayesian models with a similar likelihood that uses kernels to automatically capture complex spatial patterns using a few sparse data points. Next, we propose a variety of techniques to use convolution within the BHMs framework. Finally, we report experimental results based on simulated environments and real-world benchmark datasets.

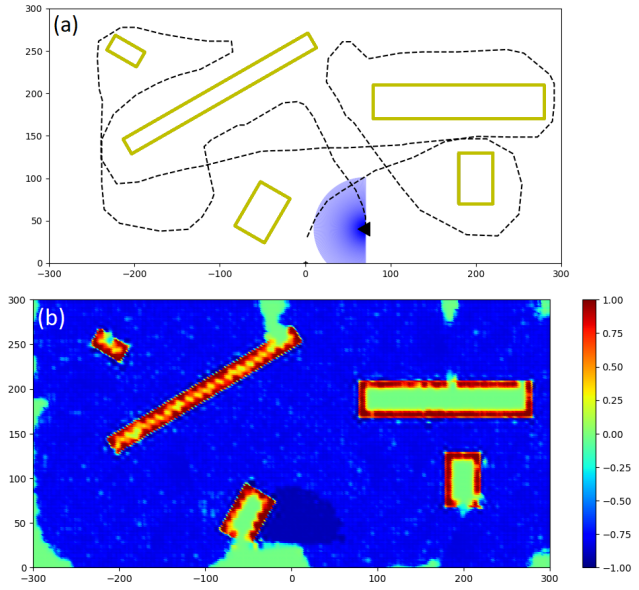

Figure 1: Bayesian Hilbert mapping (BHMs) with a moving robot (a) the obstacles of the environment are shown in yellow, the robot as a black triangle with blue rays to indicate lidar beams, and the path the robot traverses in black dotted lines (b) the occupancy map learned using the BHM model. The occupancy levels are indicated within the range $[\text{unoccupied} = -1, \text{occupied} = 1]$. Observe that the areas the robot has not seen at all and cannot either be accurately inferred from neighborhood information take an intermediate occupancy level (i.e. unknown).

Terminology and notations: In order to agree with the context and literature, we interchangeably use pseudo-inputs, supports, hinged points, and inducing points to mean the same thing. I indicates the identity matrix.

Continuous Occupancy Mapping

Conventionally, occupancy maps are built based on a fixed-sized grid (Elfes 1989) and they are typically called occupancy grid maps. Considering the disadvantages of grid maps (Senanayake, O'Callaghan, and Ramos 2017), Gaussian process occupancy maps (GPOMs) (O'Callaghan and Ramos 2014; Wang and Englot 2016; Jadidi, Miró, and Dissanayake 2017) and Hilbert maps (HMs) (Ramos and Ott 2015; Doherty, Wang, and Englot 2016) modeled the occupancy probability as a continuous function of longitude-latitude locations rather than individual probabilities associated with different cells of the grid. By considering neighborhood information, GPOMs and HMs were able to infer occupancy probabilities in even occluded or partially observed areas (Guizilini and Ramos 2017) which would otherwise be challenging with grid-based approaches. As opposed to grid maps, GPOMs and HMs can generate maps with any resolution merely by evaluating the continuous function at any location, and hence they are called continuous occupancy mapping techniques. This way it is possible to capture the continuity of the space from data itself in a principled manner rather than interpolating a discrete map as a post-processing step.

For both continuous occupancy mapping techniques, training data are $\mathcal{D} = \{\mathbf{x}_n, y_n\}_{n=1}^N$ where $\mathbf{x} \in \mathbb{R}^2$ are longitude-latitude pairs and $y \in \{-1, 1\} =: \{\text{unoccupied}, \text{occupied}\}$ are occupancy levels. Laser hit points are considered as occupied and points randomly drawn from a uniform distribution of the laser beam $\mathcal{U}(\text{robot's position, laser hit point})$ are considered as unoccupied points. The objective is to learn the occupancy probability $p(y_*|\mathbf{x}_*, \mathcal{D})$ of an unknown point in the space $\mathbf{x}_*$ based on a Bernoulli likelihood (occupied and unoccupied) using training data $\mathcal{D}$.

In GPOMs, the predictive probability distribution $p(y_*|\mathbf{x}_*, \mathcal{D}) = \mathcal{N}(\mathbf{x}_*; \mu_*, \sigma_*)$ is obtained by marginalizing the posterior distribution $p(\mathbf{f}|\mathbf{y})$ over latent functions $\mathbf{f}$, given all observations $\mathbf{y}$. In order to capture spatial dependencies, a prior distribution over latent functions $\mathbf{f}$ is introduced. This prior is a Gaussian process $p(\mathbf{f}) = \mathcal{GP}(\mathbf{0}, K)$—a collection of random variables which jointly follows a Gaussian distribution (Rasmussen and Williams 2006). Here, K is the covariance matrix whose elements can be calculated using a kernel function $k(x, x'; \theta)$ with hyperparameters θ. Typically, these hyperparameters are learned by either minimizing the negative marginal log-likelihood or through cross validation (Rasmussen and Williams 2006). Covariance matrix needs to be inverted when learning hyperparameters as well as querying the occupancy probability at unknown locations. This inversion has a computational complexity of $\mathcal{O}(N^3)$ where N is the number of data points collected by the robot. Since the size of the covariance matrix grows as the robot collects more data, GPOMs are prohibitive when mapping with large datasets. Although this work has later been scaled up (Senanayake, O'Callaghan, and Ramos 2017) using big data Gaussian process (Hensman, Matthews, and Ghahramani 2015), the computational complexity still grows as new data are collected.

The key to the success of GPOMs is using kernels to capture neighborhood information. Based on this property, (Ramos and Ott 2015) proposed to evaluate kernels $k(\mathbf{x}, \tilde{\mathbf{x}})$ between data $\mathbf{x}$ and some pseudo-inputs $\tilde{\mathbf{x}}$ hinged in different locations of the environment rather than evaluating the kernel for all pairs of data. Although this essentially reduces the computational complexity, this can no more be represented as a process which follows a joint probability distribution. Taking a frequentists approach, HMs minimizes the negative log-likelihood to learn weight parameters of the model $y = \sigma\big(w_0 + \sum_{m=1}^M w_m k(\mathbf{x}, \tilde{\mathbf{x}}_m)\big)$ where $\sigma(.)$ is the sigmoidal function and M is the number of hinged features. Since the model can be easily over-fitted due to the enormous number of features, the objective function is regularized using a combination of Lasso and Tikhonov penalties $\lambda_1\|\mathbf{w}\|_1 + \lambda_2\|\mathbf{w}\|_2^2$, where regularization parameters λ_1 and λ_2 have to be chosen heuristically.

Bayesian Hilbert Maps for Moving Robots

By taking a Bayesian approach, Bayesian Hilbert maps (BHM) (Senanayake and Ramos 2017) eliminate the requirement of heuristically tuning regularization parameters which could especially be difficult in dynamic environments. Assuming the robot is stationary, (**?**) demonstrated mapping small areas within the vicinity of the robot e.g. 60 m radius. In this section, we further discuss the streaming setting of BHMs which can accommodate a moving robot to map large areas.

A feature vector is calculated for any input $\mathbf{x}$ using M pseudo-inputs $\tilde{\mathbf{x}}$ hinged in different locations of the environment,

$$\Phi(\mathbf{x}) = \big(1, k(\mathbf{x}, \tilde{\mathbf{x}}_1), k(\mathbf{x}, \tilde{\mathbf{x}}_2), ..., k(\mathbf{x}, \tilde{\mathbf{x}}_M)\big), \quad (1)$$

where $k(\mathbf{x}, \tilde{\mathbf{x}}_1)$ is a kernel function to measures the similarity between its input points. A discussion on how to hinge these points will be discussed in detail.

The probability that a point $\mathbf{x}$ is unoccupied is given by $\sigma\big(-\mathbf{w}^\top \Phi(\mathbf{x})\big)$. However, unlike in HMs where the weights are scalars, the weights in BHMs are probability distributions $\mathbf{w} \sim \mathcal{N}(\mu_0, \Sigma_0)$. In fact, because of the sigmoidal likelihood[1], the posterior distribution is intractable (Bishop 2006; Jaakkola and Jordan 1997). Therefore, the likelihood is approximated using the Taylor expansion at local perturbation points ξ and then use variational inference is used to find an approximate posterior distribution $q(\mathbf{w}) \sim \mathcal{N}(\mu, \Sigma)$. Variational inference has been shown effective over Laplace approximation for mode matching or Markov chain Monte Carlo (McMC) techniques when the number of dimensions is high (Blei, Kucukelbir, and McAuliffe 2017). In variational inference, the log-marginal likelihood is decomposed as,

$$\ln \underbrace{\mathrm{p}(\mathbf{y}|\mathbf{x})}_{\substack{\text{marginal}\\ \text{likelihood}}} = \mathcal{L}\big(\underbrace{\mathrm{q}(\mathbf{w})}_{\substack{\text{approx.}\\ \text{posterior}}}\big) + \mathbb{KL}\big(\underbrace{\mathrm{q}(\mathbf{w})}_{\substack{\text{approx.}\\ \text{posterior}}} \,\|\, \underbrace{\mathrm{p}(\mathbf{w}|\mathbf{y})}_{\text{posterior}}\big), \quad (2)$$

where,

$$\mathbb{KL} = \int \mathrm{q}(\mathbf{w}) \ln\left(\frac{\mathrm{q}(\mathbf{w})}{\mathrm{p}(\mathbf{w}|\mathbf{y})}\right) \mathrm{d}\mathbf{w}, \quad (3)$$

is the Kullback-Leibler divergence between the approximate posterior and the true posterior. Since this is not evaluable, the lower bound (negative variational free energy),

$$\mathcal{L} = \int \mathrm{q}(\mathbf{w}) \ln\left(\frac{\mathrm{p}(\mathbf{w}, \mathbf{y})}{\mathrm{q}(\mathbf{w})}\right) \mathrm{d}\mathbf{w}, \quad (4)$$

is maximized w.r.t. μ, Σ, and ξ.

Since the robot is moving in a large area to collect new data, it is not feasible to store all data and using all data to learn parameters. In stead, we can use the posterior evaluation of the previous update as the prior distribution for the current update,

$$\begin{aligned} \text{Posterior}(t) &\propto \text{Likelihood}(t) \times \text{Prior}(t) \\ &\propto \text{Likelihood}(t) \times \text{Posterior}(t-1) \end{aligned} \quad (5)$$

This can be thought of as we implicitly carry forward all useful information extracted from the previous lidar scans into the present by means of probability distributions.

Denoting the current time step with t and the number of data points in the current scan with N_t, for each scan, the parameters can be sequentially learned in an EM-fashion,

E-step:

$$\boldsymbol{\mu}_t = \Sigma_t\left(\Sigma_{t-1}^{-1}\mu_{t-1} + \sum_{n_t=1}^{N_t}(y_{n_t} - 0.5)\Phi(\mathbf{x}_{n_t})\right) \quad (6)$$

$$\Sigma_t^{-1} = \Sigma_{t-1}^{-1} + 2\sum_{n_t=1}^{N_t}\lambda(\xi_{n_t})\Phi(\mathbf{x}_{n_t})\Phi^\top(\mathbf{x}_{n_t}) \quad (7)$$

M-step:

$$\xi_{n_t}^2 = \Phi^\top(\mathbf{x}_{n_t})(\Sigma_t + \mu_t\mu_{n_t}^\top)\Phi(\mathbf{x}_{n_t}) \quad (8)$$

Here, $\lambda(\xi) = 0.5\xi^{-1}\big(\sigma(\xi) - 0.5\big)$, with ξ as a local parameter used to linearize the function using the Taylor expansion.

GPOMs versus BHMs

Kernel methods lie at the heart of both continuous mapping techniques—GPOMS and BHMs. Kernels have been used to capture spatial dependencies. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for a non-empty set $\mathcal{X}$ is a kernel if there exists an $\mathbb{R}$-Hilbert space with a feature map $\varphi : \mathcal{X} \to \mathcal{H}$ s.t. $k(\mathbf{x}, \tilde{\mathbf{x}}) = \langle\varphi_{\mathbf{x}}, \varphi_{\tilde{\mathbf{x}}}\rangle_{\mathcal{H}}$, $\forall \mathbf{x}, \tilde{\mathbf{x}} \in \mathcal{X}$. For N inputs, the kernel matrix $K_{NN} \in \mathbb{R}^{N\times N}$ containing elements $K_{NN}[i,j] = k(\mathbf{x}_i, \mathbf{x}_j)$ is positive semi-definite. This is also the covariance matrix in a Gaussian process.

For mapping techniques, a kernel represents how close two points in the space are. Due to attractive theoretical properties, simplicity, and success in practice (Guizilini and Ramos 2016), we consider the squared-exponential kernel,

$$k(\mathbf{x}, \tilde{\mathbf{x}}) = \exp\big(-(\mathbf{x} - \tilde{\mathbf{x}})G^{-1}(\mathbf{x} - \tilde{\mathbf{x}})^\top\big), \quad (9)$$

throughout the paper. Here, $G \in \mathbb{R}^{2\times 2}$ controls the strength of similarity for longitude and latitude separately.

Consider the Bayesian linear regression model $y(\mathbf{x}) = \mathbf{w}^\top\Phi(\mathbf{x})$ with prior $\mathbf{w} \sim \mathcal{N}(0, \Sigma_0)$. When the feature vector of (1) is infinite, i.e. $M \to \infty$, and the $\Sigma_0 = \mathrm{diag}(\lambda_1, \lambda_2, \cdots)$, the covariance between two evaluations can be calculated as $\mathrm{Cov}[y(\mathbf{x}), y(\tilde{\mathbf{x}})] = \Phi(\mathbf{x})^\top\Sigma_0\Phi(\tilde{\mathbf{x}})$. With relevance to the Mercer's theorem (Hofmann, Schlkopf, and Smola 2008), observe that this is also the Eigendecomposition of a real symmetric Gaussian process covariance matrix with kernel $k(\mathbf{x}, \tilde{\mathbf{x}}) = \sum_{i=1}^{\infty}\lambda_i\Phi_i(\mathbf{x})\Phi_i(\tilde{\mathbf{x}})$. Therefore, a BHM with finitely many pseudo-points have inherent similarities to a GPOM.

The variational approximation to GPOMs (VSDGPOMs) proposed by (Senanayake, O'Callaghan, and Ramos 2017) uses Nyström approximation to make GPOMs computationally feasible. The covariance matrix is factorized as

[1]
$$\underbrace{\mathrm{q}(\mathbf{w})}_{\substack{\text{approx.}\\ \text{posterior}}} \approx \underbrace{\mathrm{p}(\mathbf{w}|\mathbf{y}, \mathbf{x})}_{\text{posterior}} \propto \underbrace{\mathrm{p}(\mathbf{y}|\mathbf{w}, \mathbf{x})}_{\text{likelihood}} \times \underbrace{\mathrm{p}(\mathbf{w})}_{\text{prior}}$$

Table 1: A comparison of the two Bayesian continuous mapping techniques: VSDGPOM vs. BHM

	VSDGPOM	BHM
Prior	$p(\tilde{\mathbf{f}}) = \mathcal{GP}(\tilde{\mathbf{f}}; \mathbf{0}, K_{MM})$	$\mathbf{w} \sim \mathcal{N}(\mathbf{w}; \mu_{t-1}, \Sigma_{t-1})$
Approx. posterior	$p(\mathbf{f}\|\mathbf{y})$	$\mathbf{w} \sim \mathcal{N}(\mathbf{w}; \mu_t, \Sigma_t)$
Kernel matrices	$M \times M$ matrix with elements $\{k(\tilde{\mathbf{x}}_{m'}, \tilde{\mathbf{x}}_m)\}_{m',m=1}^{M}$	$1 \times M$ matrix with elements $\{k(\mathbf{x}, \tilde{\mathbf{x}}_m)\}_{m=1}^{M}$
	$N \times M$ matrix with elements $\{\{k(\mathbf{x}_n, \tilde{\mathbf{x}}_m)\}_{n=1}^{N}\}_{m=1}^{M}$	The matrix does not grow with N
Runtime	$\mathcal{O}(M^2N); \quad M << N$	$\mathcal{O}(M^3)$

$K_{NN} \approx K_{NM}\tilde{K}_{MM}^{-1}K_{NM}^{\top}$ with $K_{NN} \in \mathbb{R}^{N\times N}$ and $\tilde{K}_{MM} \in \mathbb{R}^{M\times M}$ for N inputs and $M(<< N)$ pseudo-inputs (inducing inputs). These pseudo-inputs should be able to represent the dataset, and hence they can be chosen as a subset of input points or by a clustering technique such as a density based clustering technique such as DBSCAN. In a similar fashion, some pseudo-inputs to the kernel, named hinged features, should be chosen in both HMs (Guizilini and Ramos 2017) and BHMs (Senanayake and Ramos 2017). Therefore, the accuracy of both models depends on the choice of these pseudo-inputs. However, for an input $\mathbf{x}$, the kernels of VSDGPOMs are computed using $\{k(\mathbf{x}, \tilde{\mathbf{x}}_m), k(\tilde{\mathbf{x}}_{m'}, \tilde{\mathbf{x}}_m)\}_{m',m=1}^{M}$ where as the kernels of BHMs are computed using $\{k(\mathbf{x}, \tilde{\mathbf{x}}_m)\}_{m=1}^{M}$. Note that since the kernel is evaluated for all possible combinations of pseudo-input pairs $(\tilde{\mathbf{x}}_{m'}, \tilde{\mathbf{x}}_m)$ in VSDGPOMs, it is a matrix, where as it is a feature vector in BHMs.

The Nyström approximation used in VSDGPOMs is,

$$\tilde{\Phi}(\mathbf{x}) = \tilde{D}^{-1/2}\tilde{V}^{\top}\big(k(\mathbf{x}, \tilde{\mathbf{x}}_1), k(\mathbf{x}, \tilde{\mathbf{x}}_2), \cdots, k(\mathbf{x}, \tilde{\mathbf{x}}_M)\big)^{\top}, \tag{10}$$

where $\tilde{D} = diag(\tilde{\lambda}_1, \tilde{\lambda}_2, \cdots, \tilde{\lambda}_M)$ and $\tilde{V} = (\tilde{\mathbf{v}}_1, \tilde{\mathbf{v}}_2, \cdots, \tilde{\mathbf{v}}_M)$ are Eigenvalues and Eigenvectors, respectively. Observe that the kernel vector of (10) has the same form as that of BHMs given in (1). It can be easily shown that the (n, n') element of K_{NN} is $\tilde{\Phi}^{\top}(\mathbf{x}_n)\tilde{\Phi}(\mathbf{x}_{n'})$.

Although pseudo-inputs in VSDGPOM being a subset of input is not required, they should represent the entire dataset (Hensman, Matthews, and Ghahramani 2015; Bauer, van der Wilk, and Rasmussen 2016), possibly as the cluster centers of a clustering algorithm. This is because the kernel is evaluated for all possible combinations of pseudo-input pairs $(\tilde{\mathbf{x}}_{m'}, \tilde{\mathbf{x}}_m)$, and they are directly used to define the prior as $p(\tilde{\mathbf{f}}) := p\big(f(\tilde{\mathbf{x}}_1), f(\tilde{\mathbf{x}}_2), \cdots, f(\tilde{\mathbf{x}}_M)\big)$. Hence, pseudo-inputs that do not represent the dataset, i,e. an erroneous prior, would result in a fallacious posterior distribution. In contrast, pseudo-inputs to the BHMs can consist of points that represent the dataset as well as any other point in the space. Moreover, unlike in VSDGPOM, as shown in Table 1, BHMs neither require calculating the kernel (correlation) between pseudo-inputs nor they are used to compute the prior. Instead, as described previously, the prior is a distribution over weights of the kernel. As we will show in experiments, this property of being able to capture the spatial correlation between an input point and any other point in the space is useful for making inference in areas of the environment where few or no data point are available.

One other important factor to use continuous occupancy mapping techniques in real world is the run time of algorithms. For N inputs (the total number of points over all lidar scans) and $M(<< N)$ pseudo-inputs, GPOM has a runtime computational complexity of $\mathcal{O}(N^3)$ and VSDGPOM has a computational complexity of $\mathcal{O}(NM^2)$, while the streaming setting of the BHMs has a computational complexity of $\mathcal{O}(M^3)$. Therefore, BHMs are efficient for $M < N$ which indeed is the case. On the other hand, since the computational complexity of streaming BHMs is not dominated by N, as opposed to Gaussian process based techniques, learning time does not increase as more data are captured. Here, we assume that the number of data points in each scan is considerably smaller than the entire dataset, i.e. $N_t << N$. Data gathered in each scan can be simply discarded after updating the model. A comparison of the two models is given in Table 1.

Convolution for Occupancy Mapping

Convolution of functions is ubiquitous in signal processing and control theory for applications from signal smoothing to systems identification. Convolution has also been successfully used for image recognition using convolutional neural networks (CNNs). In this section, we present a variety of techniques to use convolution of kernels as a powerful tool to improve the streaming setting of BHMs and map large areas with moving robots.

Sampling laser beams

As discussed in the Continuous Occupancy Mapping Section, laser hit points are considered as occupied ($y = +1$) and an arbitrary number of points drawn randomly from a uniform distribution between the robot and the laser hit point are considered as unoccupied points ($y = -1$). However, this could result in having fewer points away from the sensor more points close to the sensor. Although it is possible to sample finitely many data points along the laser beam, it would then require evaluating finitely many kernels and increasing model update time. As illustrated in figure 2, we propose to use line segments between the sample points rather than points itself.

Consider a line segment is given by a vector $\vec{\mathbf{x}} := \vec{\mathbf{x}}_a + \zeta\vec{\mathbf{x}}_b$ parameterized by $\zeta \in [0, 1]$. In the two dimensional case, it is $(x_1, x_2) = (x_{1,a}, x_{2,a}) + \zeta(x_{1,b}, x_{2,b})$. We compute the convolution between between a vector $\vec{\mathbf{x}}$ and a

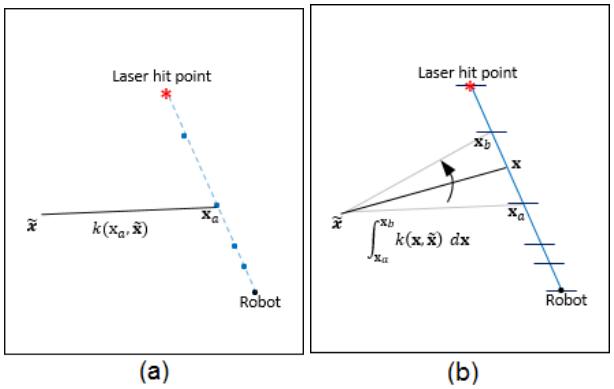

Figure 2: (a) points are used in typical continuous occupancy mapping (b) using a collection of line segments instead of points

pseudo-input $\tilde{\mathbf{x}}$ as,

$$k(\vec{\mathbf{x}}_\zeta, \tilde{\mathbf{x}}) = \oint_0^1 \exp\big(-(\vec{\mathbf{x}}_\zeta - \tilde{\mathbf{x}})G^{-1}(\vec{\mathbf{x}}_\zeta - \tilde{\mathbf{x}})^\top\big)\mathrm{d}\zeta = \int_{\mathbf{x}_a}^{\mathbf{x}_b} \exp\big(-(\mathbf{x} - \tilde{\mathbf{x}})G^{-1}(\mathbf{x} - \tilde{\mathbf{x}})^\top\big)\mathrm{d}\mathbf{x} \tag{11}$$

Note that this results in averaging effects. Though there is no closed form solution, this can be easily evaluated using approximations to error functions $\mathrm{erf}(x) := \pi^{-\frac{1}{2}}\int_{-x}^{x}\exp(-t^2)\mathrm{d}t$ which are readily available in many scientific computing libraries. Another method approximate is to consider Riemann sum or considering a drawing N_{ab} samples from a uniform distribution $\mathbf{x}_{ab} \sim \mathcal{U}(\mathbf{x}_a, \mathbf{x}_b)$,

$$k(\vec{\mathbf{x}}_\zeta, \tilde{\mathbf{x}}) \approx \frac{1}{N_{ab}}\sum_{i=1}^{N_{ab}} \exp\big(-(\mathbf{x}_{ab} - \tilde{\mathbf{x}})G^{-1}(\mathbf{x}_{ab} - \tilde{\mathbf{x}})^\top\big) \tag{12}$$

This can be understood as having sub-line segments. Consider the σ-algebra formed by $\mathcal{S}$ having all intervals $[a', b')$ in $[a, b)$. Then $[a', b')$ is also a Borel set. For the Lebesgue measure $\mathbb{P}$ of pairwise disjoint sets $A_i = [a_i, b_i)$, from σ-additivity, $\mathbb{P}\big(\cup_{i=1}^{\infty} A_i\big) = \sum_{i=1}^{\infty}\mathbb{P}(A_i)$.

The method proposed in (O'Callaghan and Ramos 2011) related to GPOMs integrates out the entire laser beam—from the robot to the laser hit point—result in merely one feature which could not be representative enough for complex and dynamic environments in Hilbert mapping.

Regionalized supports

So far the supports we considered are points in the space (Figure 3(a)). Ideally, there should be infinite number of such supports. The farther the points are to each other, the poorer the representation is. However, the computational complexity of BHMs cubically increases with the number of supports—$\mathcal{O}(M^3)$. Therefore, we propose to convolve data points with areas rather than points to cover the entire realm. The following techniques can be used in order to perform convolution in a tractable and computationally efficient way.

Mixture of Gaussians As shown in Figure 3(b), points can be replaced with Gaussian distributions $\mathcal{N}(\mathbf{m}, S)$. Now, the effective kernel becomes a convolution over probability measures $\mathbb{P}$. Using 9, the kernel between the input $\mathbf{x}$ and m^{th} measurable support $\tilde{\mathbf{x}}_m$,

$$\begin{aligned} &k\big(\mathbf{x}, \mathbb{P}(\tilde{\mathbf{x}}_m)\big) \\ &= \int k(\mathbf{x}, \tilde{\mathbf{x}}_m)\mathrm{d}\mathbb{P}(\tilde{\mathbf{x}}_m) \\ &= \int k(\mathbf{x}, \tilde{\mathbf{x}}_m; G)\mathcal{N}(\tilde{\mathbf{x}}_m; \mathbf{m}_m, S_m)\mathrm{d}\tilde{\mathbf{x}}_m \\ &= |G^{-1}S + 2I|^{-\frac{1}{2}} \\ &\quad \exp\big(-(\mathbf{x} - \tilde{\mathbf{x}}_m)(G + S)^{-1}(\mathbf{x} - \tilde{\mathbf{x}}_m)^\top\big) \end{aligned} \tag{13}$$

The derivation is straightforward.

Rather than assuming isotropy as in Figure 3(b), it is possible to use the full covariance matrix S as in Figure 3(c). To this end, a Gaussian mixture model $f(\mathbf{x}) = \sum_{m=1}^{M}\mathcal{N}(\mathbf{x}; \mathbf{m}_m, S_m)$ can be used (Bishop 2006). However, as a straightforward modification to the standard algorithm, means should be kept stationary in a regular grid while learning the covariances. Using k-means to cluster input data, (Guizilini and Ramos 2016) have shown that having different variances along different directions is useful for obtaining smoother edges in 3D surface modeling. However, unlike GMMs, k-means does not provide covariances and hence a k-means based method cannot be readily incorporated into (13).

Tessellation Another straightforward method to cover the entire mapping area is tessellation. Note that this is completely different to occupancy grid maps (Elfes 1987) as the objective of tessellation here is to build supports which are then used to compute kernels. Denoting an area with $s(\cdot)$, the kernel is,

$$\begin{aligned} k\big(\mathbf{x}, s(\tilde{\mathbf{x}}_m)\big) &= \int_{a_{m1}}^{b_{m2}}\int_{a_{m1}}^{b_{m2}} k\big(\mathbf{x}, (\tilde{x}_{m1}, \tilde{x}_{m2})\big)\mathrm{d}\tilde{x}_{m1}\mathrm{d}\tilde{x}_{m2} \\ &\approx \frac{1}{|\tilde{\mathcal{X}}_m|}\sum_{\tilde{\mathbf{x}}'_m \in \tilde{\mathcal{X}}_m} k(\mathbf{x}, \tilde{\mathbf{x}}'_m) \end{aligned} \tag{14}$$

where $\tilde{\mathcal{X}}_m$ are samples drawn from a bivariate uniform distribution on the longitude-latitude plane $(a_{m1}, b_{m1}] \times (a_{m2}, b_{m2}]$. For simplicity, we restrict the tessellation to be rectangles as in Figure 3(d). Rather than drawing samples from a uniform distribution, as in (Reid, Ramos, and Sukkarieh 2009), it is also possible to use error functions to make the kernel computationally evaluable.

Experiments and Discussions

The program was developed using the Python programming language and an Intel corei7 laptop (no GPUs) with 8 GB RAM was used to conduct experiments. Three different datasets were used to carry out experiments to show different aspects of the proposed techniques,

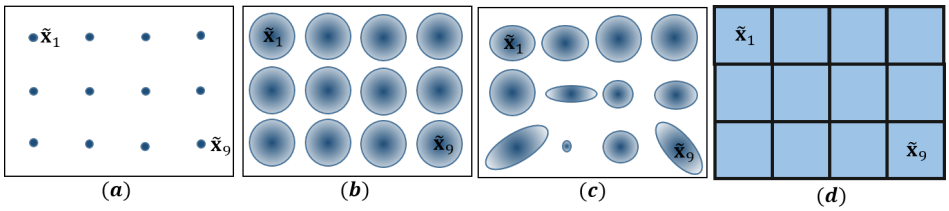

Figure 3: Regionalized supports (a) points (b) Gaussians (c) Gaussians with relevance detection (d) rectangular supports

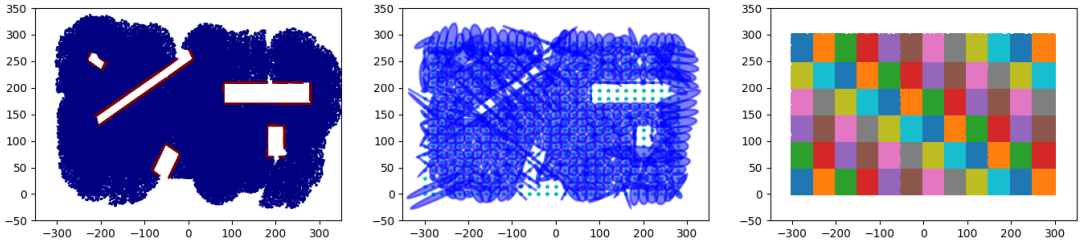

Figure 4: (a) Dataset 2 (b) a mixture of Gaussians (c) rectangular tessellation. Observe that the Gaussians are aligned with borders, indicating high correlation while Gaussians are almost isotropic when data are identically distributed. Inside objects where there are no data, Gaussians have a negligible covariance, making them equivalent to having points.

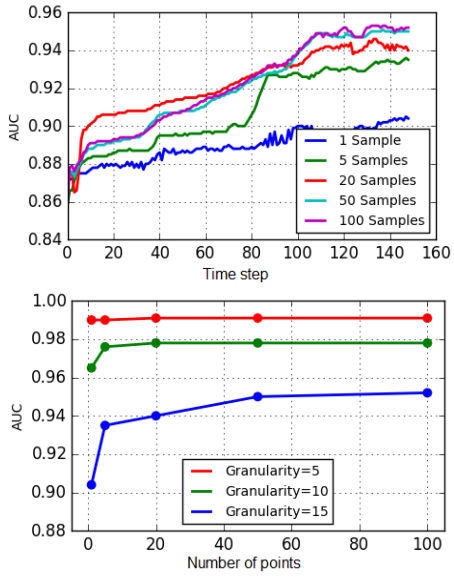

Figure 5: The effect of regionalization

1. Dataset 1: A publicly available simulated environment with moving obstacles used in (Senanayake, O'Callaghan, and Ramos 2017). The robot is stationary and LiDAR sensor can see 180^0 within a $80\,m$ radius (figures in supplementary materials).
2. Dataset 2: A large ($600 \times 300\ m^2$) simulated environment to represent outdoor. A robot with a $60\ m$ LiDAR moves in the environment (Figure 1).
3. Dataset 3: A real world dataset to represent an indoor environment at Intel Labs (publicly available: http://radish.sourceforge.net/). The robot moves in a passage multiple loops (Figure 7).

Table 2: Average AUC and NLL ($\mu \pm 2\sigma$) calculated using the test datasets. Large AUC values and small NLL values indicate accurate models.

	AUC		NLL	
Method	Dataset 2	Dataset 3	Dataset 2	Dataset 3
BHM	0.83 ± 0.12	0.92 ± 0.17	0.47 ± 0.14	0.36 ± 0.19
VSDGPOM	0.80 ± 0.20	0.79 ± 0.16	0.37 ± 0.16	0.53 ± 0.12

In all experiments, we assumed the position of the robot is known from an external sensor and hence we only focused on mapping. $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, 10^4 I)$ was used as the diffused prior and G matrix of the kernel were $16I$, $13I$, and $0.15I$ for datasets 1,2, and 3, respectively. Note that these values control the smoothness and can be set using grid search or by maximizing the objective function.

The area under ROC curve (AUC) and negative log-loss (NLL) are used to evaluate the accuracy of the models as in (Bishop 2006; Senanayake, O'Callaghan, and Ramos 2017). 10% of randomly chosen data that were never used for training were used to compute these metrics i.e. testing dataset.

Convolution of kernels

Figure 4 (b) and (c) were obtained usig dataset 2 to validate the method illustrated in Figure 3 (c) and (d), respectively. In this experiment, we consider the effect of considering areas over points. Dataset 1 was used to see the temporal effect as it is a dynamic environment with moving vehicles. As shown in Figure 5 (a), we draw different number of samples and

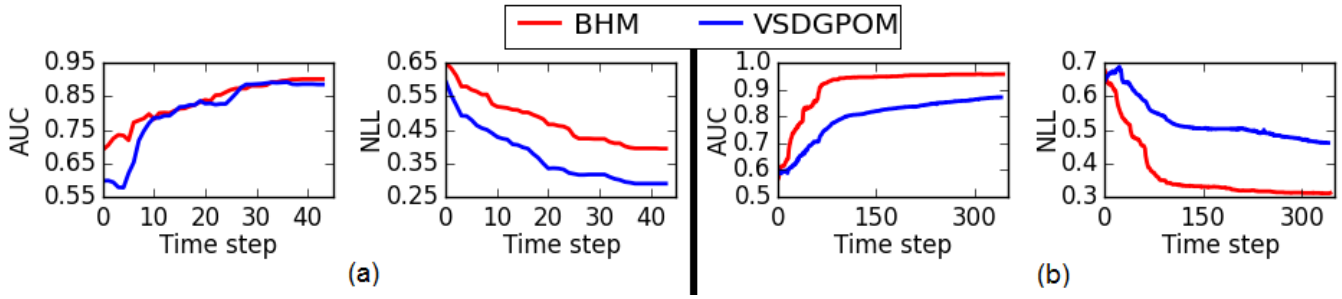

Figure 6: AUC and NLL (a) Dataset 2 (b) Dataset 3

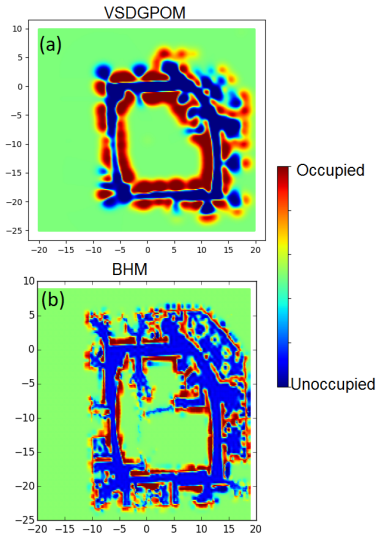

Figure 7: Dataset 3 (a) VSDGPOM (b) BHM

computed the convolution as in (14), keeping the center-to-center distance of areas (granularity) at 15. The sample size of 1 is equivalent to having point supports rather than region supports as in Figure 3 (a). As the number of samples increases up to a certain limit, the accuracy also increases. As shown in Figure 5 (b), this limit depends on the granularity. In conclusion, regionalization is important especially when the supports are farther apart (i.e. bigger granularity values). This occurs when we want to map large environments as we do not want to set M to a large number.

Moving robot

(Senanayake and Ramos 2017) assumed that the robot is stationary. However, for real world applications, it is required to move the robot and map since the distance a commercial lidar can see does not typically exceed 100 m. In this experiment, moving robots of datasets 2 and 3 were used. A RBF was used as the kernel.

Table 2 shows the average AUC and NLL for the two datasets while Figure 6 illustrates how the the accuracy improves as the robot captures more data. The AUC of BHM is always higher. However, for dataset 3, NLL for VSDGPOM of dataset 2 is lower than (the lower the NLL, the better the model is) that of BHMs. This is due to two reasons: i) the supports of BHM were placed sparsely at 5 m, and ii) the NLL calculation using the 10% of data used in the test set does not take into account the unseen areas which are in fact erroneously predicted in VSDGPOM. This is illustrated in Figure 8 (b). Comparing Figures 7 (a) and (b), although similar squared-exponential kernels used in both methods, BHMs can easily capture sharp edges and boundaries with much fine details while VSDGPOMs result in a dilated map.

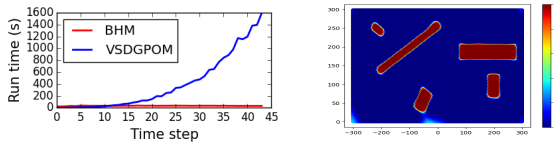

Figure 8: (a) Run time for dataset 2. BHMs have an almost constant update time while it cubically increases in VSDGPOM. (b) VSDGPOM map for dataset 3. In comparison with Figure 1, observe that values inside the obstacles are erroneously predicted as occupied instead of unknown.

On the other hand, as shown in Figure 8 (a), the runtime of VSDGPOM grows as more data are captured. In contrast, as discussed, the runtime of BHMs is almost constant for every update. Practically, BHMs are efficient as the convergence of the EM algorithm occurs in 1-2 iterations. Note that (6)-(8) require inverting the covariance matrix of the prior distribution. For M supports, this would cost $\mathcal{O}(M^3)$. When mapping even larger areas of several square miles, it is straightforward to partition the environment and have a collection of maps.

Conclusions

The two main Bayesian continuous occupancy mapping techniques were analyzed and maps were built using moving robots using the streaming setting of the Bayesian Hilbert maps algorithm. These maps can be improved by using convolution of kernels in different ways. Since only squared-exponential kernels have been considered in Hilbert mapping literature, it would be worthwhile to investigate using a combination of kernels (Thomas et al. 2002; Smits and E.M. 2002) to further improve the quality of maps.

References

Bauer, M.; van der Wilk, M.; and Rasmussen, C. E. 2016. Understanding probabilistic sparse gaussian process approximations. In *Advances in Neural Information Processing Systems*, 1533–1541.

Bishop, C. 2006. *Pattern Recognition and Machine Learning*. Springer.

Blei, D.; Kucukelbir, A.; and McAuliffe, J. D. 2017. Variational inference: A review for statisticians. In *arXiv:1601.00670v5*.

Doherty, K.; Wang, J.; and Englot, B. 2016. Probabilistic map fusion for fast, incremental occupancy mapping with 3d hilbert maps. In *IEEE International Conference on Robotics and Automation (ICRA)*, 0–0.

Elfes, A. 1987. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation* RA-3(3):249–265.

Elfes, A. 1989. *Occupancy grids: a probabilistic framework for robot perception and navigation*. Ph.D. Dissertation, Carnegie Mellon University.

Guizilini, V., and Ramos, F. 2016. Large-scale 3d scene reconstruction with hilbert maps. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Guizilini, V., and Ramos, F. 2017. Learning to reconstruct 3d structures for occupancy mapping. In *Proceedings of Robotics: Science and Systems (RSS)*.

Hensman, J.; Matthews, A.; and Ghahramani, Z. 2015. Scalable variational gaussian process classification. In *the Eighteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Hofmann, T.; Schlkopf, B.; and Smola, A. 2008. Kernel methods in machine learning. *The Annals of Statistics* 36(3):1171–1220.

Jaakkola, T., and Jordan, M. 1997. A variational approach to bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, volume 82.

Jadidi, M. G.; Miró, J. V.; and Dissanayake, G. 2017. Warped gaussian processes occupancy mapping with uncertain inputs. *CoRR* abs/1701.00925.

Kivinen, J.; Smola, A.; and Williamson, R. 2014. Online learning with kernels. 2165 – 2176.

Marinho, Z.; Boots, B.; Dragan, A.; Byravan, A.; Gordon, G. J.; and Srinivasa, S. 2016. Functional gradient motion planning in reproducing kernel hilbert spaces. In *Proceedings of Robotics: Science and Systems (RSS)*.

Mukadam, M.; Dong, J.; Yan, X.; Dellaert, F.; and Boots, B. 2017. Continuous-time gaussian process motion planning via probabilistic inference. *arXiv:1707.07383*.

Norouzii, M.; Miro, J. V.; and Dissanayake, G. 2016. Probabilistic stable motion planning with stability uncertainty for articulated vehicles on challenging terrains. *Autonomous Robots* 40(2):361–381.

O'Callaghan, S. T., and Ramos, F. T. 2011. Continuous occupancy mapping with integral kernels. In *AAAI Conference on Artificial Intelligence*, 1494–1500.

O'Callaghan, S., and Ramos, F. 2014. Gaussian Process Occupancy Maps for Dynamic Environment. In *Proceedings of the International Symposium of Experimental Robotics (ISER)*.

Ramos, F., and Ott, L. 2015. Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent. In *Proceedings of Robotics: Science and Systems (RSS)*.

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

Reid, A.; Ramos, F.; and Sukkarieh, S. 2009. Bayesian fusion for multi-modal aerial images. In *Robotics: Science and Systems (RSS)*.

Senanayake, R., and Ramos, F. 2017. Bayesian hilbert maps for continuous occupancy mapping in dynamic environments. In *Workshop on Machine Learning for Autonomous Vehicles at the 34th International Conference on Machine Learning (ICML)*.

Senanayake, R.; O'Callaghan, S.; and Ramos, F. 2017. Learning highly dynamic environments with stochastic variational inference. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Smits, G., and E.M., J. 2002. Improved svm regression using mixtures of kernels. In *International Joint Conference on Neural Networks (IJCNN)*.

Smola, A., and Schlkopf, B. 2003. Bayesian kernel methods. In *Proceedings of the Summer School, Australian National University*. Springer-Verlag.

Stachniss, C.; Leonard, J. J.; and Thrun, S. 2016. Simultaneous localization and mapping. In *Springer Handbook of Robotics*. Springer. 1153–1176.

Thomas, G.; Flach, P.; Kowalczyk, A.; and Smola, A. 2002. Multi-instance kernels. In *International conference on machine learning*, 179–186.

Wang, J., and Englot, B. 2016. Fast, accurate gaussian process occupancy maps via test-data octrees and nested bayesian fusion. In *IEEE International Conference on Robotics and Automation (ICRA)*, 1003–1010.