Mapping Occupancy of Dynamic Environments using Big Data Gaussian Process Classification

Ransalu Senanayake
School of Information Technologies
University of Sydney
rsen4557@uni.sydney.edu.au

Simon O'Callaghan
Machine Learning and Analytics Team
Data61/CSIRO, Australia
simon.ocallaghan@data61.csiro.au

Fabio Ramos
School of Information Technologies,
Australian Centre for Field Robotics
University of Sydney
fabio.ramos@sydney.edu.au

Abstract

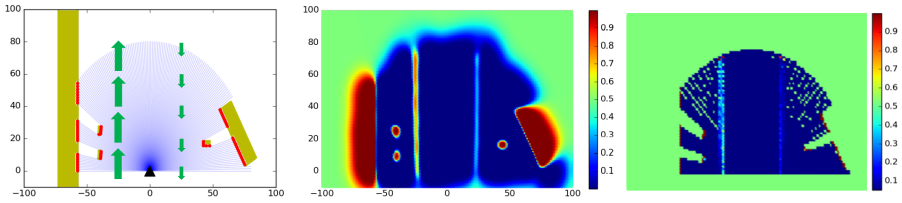
Understanding the dynamics of urban environments is crucial for path planning and safe navigation. However, the dynamics might be extremely complex making mapping a non-trivial problem. Within the methods available for learning dynamic environments, Dynamic Gaussian process occupancy maps (DGPOM) are attractive because they can produce spatially-continuous occupancy maps taking into account neighborhood information, and provide probabilistic estimates, naturally inferring the uncertainty of predictions. Despite these properties, they are extremely slow, especially in dynamic mapping where the parameters of the map have to be updated as new data arrive from range sensors such as LiDARs. In this work, we leverage recent advancements in stochastic variational inference (SVI) to quickly learn dynamic areas in an online fashion. Further, we propose an information-driven technique to "intelligently" select *inducing points* required for SVI without relying on any object tracker or velocity information. Our experiments with both simulation and real robot data on road intersections show a significant improvement in speed while maintaining a comparable or better accuracy as DGPOM. video: https://youtu.be/RItH8HH82ss

1 INTRODUCTION

Autonomous vehicles will be present in most major cities within the next ten years. These vehicles will be required to navigate among people, bicycles, and other vehicles, while attempting to maximize the transport efficiency and reducing the chance of accidents. However, urban environments can be very challenging to model due to the multitude of objects and complex dynamics such as intersections where cars move in opposite directions, at different speeds, and potentially also turning. Hence, in order to develop planning algorithms that are both safe and robust, it is important to learn which areas are safe to maneuver. Additionally, a long-term model of the dynamics of urban environments can improve traffic minimizing travel times and battery consumption.

The majority of the methods used to represent occupancy assumes a static environment. In occupancy grid maps [3], the world is divided into a grid with a fixed cell size and a Bayes filter is used to estimate the occupancy probability. It has three main limitations: 1) the cell size has to be predetermined heuristically (cannot be very large or small) and hence a map with varying resolutions cannot be rendered, 2) the world is discretized and hence the map is not continuous and, importantly, 3) the

29th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.



(a) Motion field (b) VSDGPOM (c) DGrid

Figure 1: The occupancy map produced using the proposed algorithm (VSDGPOM). The robot, indicated by the black arrow head, resides at the middle of the two roads. Its field of view is shown in blue laser beams with red laser hit points, when there are no moving vehicles. Static objects such as buildings and parked vehicles are shown in yellow and the traffic flow in green arrows. Vehicles moving in the upward direction are more frequent than that of downward. Therefore, after several laser observations, the occupancy probability of the left road shown in (b), the results obtained from the proposed method, is higher than that of the right road. The occupancy probability of unseen outlying areas is almost 0.5. Since the model captures neighborhood relationships, areas around $(-50, 15)$, $(-50, 35)$ and $(60, 20)$ are correctly mapped, regardless of occlusions due to the three parked vehicles. (c) is the equivalent map obtained from dynamic occupancy grid maps (DGrid).

cells are assumed to be independent and hence loose the interpolation power which makes the map susceptible for occlusions and invalid laser reflections (Fig. 1).

Considering these disadvantages of grid maps, Gaussian process occupancy maps (GPOM) [11, 17] and Hilbert maps [12] were built for purely static environments. The key to the success of these methods was using kernel machines to capture spatial relationships and dependencies. Though it is not clear how Hilbert maps can be used in dynamic environments, [10] proposed an extension to GPOM called dynamic GPOM (DGPOM), for mapping long-term dynamics. Although DGPOMs are appealing, they have an $\mathcal{O}(N^3)$ computational cost as in any conventional Gaussian process based model where N is the number of data points. In the dynamic setting, where the map has to be updated as new laser scans arrive, N grows unwieldy and hence updating the map at least in near real-time is prohibitive beyond a few hundred data points. The other conventional approach to use occupancy grid maps to build long-term dynamic occupancy grid maps (DGrid) is assigning a memory unit for each cell and updating individual cells as new data arrives without considering any spatial relationships [1]. The majority of other works that use grid maps in non-static environments have been dedicated for other aspects of dynamic environments such as detecting short-term patterns [8, 18] and removing dynamic objects from static maps [5] [16].

Despite Gaussian processes [13] massive success in machine learning and geostatistics for spatial interpolation problems, they have been less appealing for robotics applications mainly because of the scalability issues which in turn became the major bottleneck of DGPOM. In this paper, we utilize state-of-the-art stochastic optimization techniques to build dynamic occupancy maps, significantly ameliorating the scalability issues. Rather than using the entire data set for optimization, our framework "intelligently" select *inducing input points* to sparsely represent denser areas without discarding[1] any data.

While entertaining all advantages of GPOM — continuous, considers spatial dependencies and provides mean and variance of estimations — our model[2] has the following advantages compared to existing methods;

1. It can build long-term occupancy maps in large and highly dynamic environments with thousands of data points within minutes, which would otherwise take several days with existing methods such as vanilla DGPOM.
2. It can sequentially update the long-term occupancy map as new laser scans are captured.
3. It learns all key parameters, including inducing points, by the model itself and hence the accuracy does not rely on heuristic parameter choices.
4. It does not require any underlying motion model or object trackers.

[1]Note that discarding informative data is not desirable for any learning technique.

[2]Python code: `https://goo.gl/VvlF6f`

2 GAUSSIAN PROCESS OCCUPANCY MAPS

Consider a robot with known localization and equipped with a 2D laser scanner in a static environment. The end-point of each laser reflection is considered as occupied $y = 1$ and a randomly sampled points between the end-point and the sensor are considered as unoccupied $y = 0$. The corresponding 2D longitude-latitude locations are given by $\mathbf{x} = (x_{\text{longi}}, x_{\text{lati}})$. The robot collects such N input-output pairs $\{(\mathbf{x}_n, y_n)\}_{n=0}^{N}$ over time.

A Gaussian process prior is introduced over the latent functions $\mathbf{f} := \big(f(\mathbf{x}_1), f(\mathbf{x}_2), f(\mathbf{x}_3), \ldots, f(\mathbf{x}_N)\big)$ as $p(\mathbf{f}) = \mathcal{GP}(\mathbf{0}, K_{NN})$ where K_{NN} is the $N \times N$ covariance matrix whose elements are typically calculated as $k(\mathbf{x}, \mathbf{x}') := \alpha \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}'\|_2^2\right)$. Since the output is either 0 or 1 (unoccupied or occupied), the likelihood is a Bernoulli distribution $p(\mathbf{y}|\mathbf{f}) = \prod_{n=1}^{N} \phi_n^{y_n}(1 - \phi_n)^{1-y_n}$ where $\phi_n := \phi(f(\mathbf{x}_n))$ is the function evaluated at $\mathbf{x}_n$ and then "squashed" using a probit or sigmoid function denoted by $\phi(\cdot)$. The marginal likelihood can be calculated by integrating the joint prior-likelihood distribution over $\mathbf{f}$. However, because of the Bernoulli likelihood, exact computation of the posterior is not analytically tractable and hence the posterior is approximated by a Gaussian distribution $q(\mathbf{f}) \approx p(\mathbf{f}|\mathbf{y})$. GPOM uses a local probabilistic least square approximation [13] while we use a more robust approach in this paper which will be discussed in section 3.1.

Once the data are collected, the first step is to learn the model by optimizing hyperparameters α and γ with respect to the log marginal likelihood $\log p(\mathbf{y})$ [13] using an iterative optimization procedure. However, since $\log p(\mathbf{y}) = -\frac{1}{2}\mathbf{y}^\top K_{NN}^{-1}\mathbf{y} - \frac{1}{2}\log\big((2\pi)^N |K_{NN}|\big)$, each step of optimization has computational complexity $\mathcal{O}(N^3)$. Since the number of data points N grows over time, computations become extremely slow and hence GPOM is limited to a few hundred data points. Having trained the model, the predictive occupancy with mean and variance for a query location $\mathbf{x}_*$ can be obtained by integrating the approximated posterior as $p(f(\mathbf{x}_*)|\mathbf{y}, X, \mathbf{x}_*) = \int p(f(\mathbf{x}_*)|\mathbf{f}) q(\mathbf{f}) d\mathbf{f} = \mathcal{N}(\text{mean}, \text{var})$. This step also involves inverting K_{NN} which slows the GPOM further.

[10] extends this procedure to dynamic environments (DGPOM) by incorporating motion information of the environment into the static map. To this end, the velocities of the dynamic areas $\mathbf{v} := [v_{\text{longi}}, v_{\text{lati}}]$ are calculated by subtracting consecutive laser scans and then they are implicitly fed into the kernel as $\mathbf{x}_{\text{modified}} := [(x_{\text{longi}} + \int v_{\text{longi}} dt), (x_{\text{lati}} + \int v_{\text{lati}} dt), (t_{\text{new}} - t_{\text{old}})]$, assuming constant acceleration of dynamic objects. The authors illustrate the potential of this method for developing long-term maps [10]. However, the computational time dramatically increases as more data are collected. In section 3, we propose a scalable technique to build long-term maps which sequentially learns dynamic areas by itself without relying on underlying vehicle trackers or optical flow. Under our framework, we do not consider the dichotomy, static vs. dynamic maps, as static maps are essentially a sub-case of dynamic maps.

3 LONG-TERM MAPS WITH GAUSSIAN PROCESS (VSDGPOM)

3.1 Variational sparse Gaussian Process classification

In section 3 we described that the exact posterior is intractable and hence GPOM uses an alternative approximation which is often less robust. Other techniques, such as *variational inference* [2], approximate the intractable posterior $p(\mathbf{f}|\mathbf{y})$ with $q(\mathbf{f})$, a distribution of know form such as $\mathbf{f} \sim \mathcal{N}(\text{mean}, \text{variance})$. Alternatively, Markov chain Monte Carlo (MCMC) sampling and Laplace approximation can be can be used to approximate the posterior. In this paper, we use variational inference due to its appealing computational cost, being significantly faster than MCMC, while preserving similar level of accuracy. Variational inference is also known to provide more accurate approximations than first order methods such as Laplace approximation [2].

In variational inference, the parameters of $q(\mathbf{f}) = \mathcal{N}(\text{mean}, \text{variance})$ are iteratively estimated by minimizing the distance between the true posterior distribution and approximate posterior distribution measured by the KL-divergence $\mathbb{KL}[p(\mathbf{f}|\mathbf{y})\|q(\mathbf{f}))]$. Since the true posterior is difficult to compute, an alternative lower bound $\mathcal{L}$ is minimized. In 2015, Hensmen et al. [6] leverage properties of variational inference and incorporate them into the sparse GP classification framework [19]. Since *inducing inputs* are used in sparse GPs, an approximate posterior distribution $q(\breve{\mathbf{f}})$ can be defined

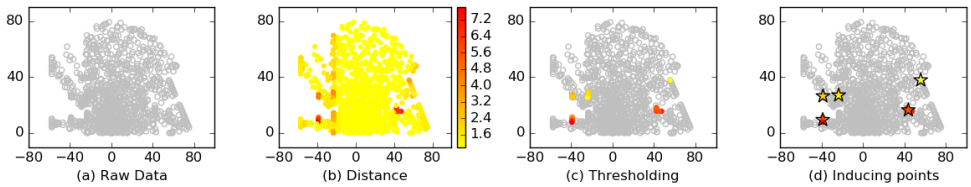


Figure 2: (a) Raw data of a new laser scan (both occupied and unoccupied plotted together at $t > 1$ for the dataset in Fig. 1a. The data is used to query the model and the distance metric is calculated as in (b). Thresholded data with $\zeta = 3.5$ shown in (c) are used to calculate cluster centroids (DBSCAN) as indicated in ★. $N_t = 1186$ data points in (a) is reduced to 21 after thresholding and, further reduced to $M_t = 5$ after obtaining centroids.

over inducing functions $\breve{\mathbf{f}} := \big(f(\breve{\mathbf{x}}_1), f(\breve{\mathbf{x}}_2), f(\breve{\mathbf{x}}_3), \ldots, f(\breve{\mathbf{x}}_M)\big)$ in a similar way $\mathbf{f}$ is defined as in Section 2, but with a smaller number of points $M \ll N$.

3.2 Choosing inducing points

Selecting the minimum number of inducing points and placing them appropriately are crucial to maintain the speed-accuracy trade off. The common practice is to naively choose a pre-determined number of inducing points (fixed M) randomly or using the k-means algorithm because at least obtaining a sub-optimal solution is computationally prohibitive [15]. In this section, we propose a technique capable of determining the number of inducing points M as well as where to place them. This involves two steps and they are summarized in Fig. 2, Algorithm 2 and, the following two subsections.

Algorithm 1 VSDGPOM learning algorithm

function LEARN(ζ)
  Initialize $(\mathbf{x}, y)$ as null matrices
  Initialize $t \leftarrow 0$
  while new laser scan **do**
    $t \leftarrow t + 1$
    $(\mathbf{x}_t, y_t) \leftarrow$ Extract new points
    if $t = 1$ **then**
      Randomly initialize $\breve{\mathbf{x}}, \gamma, \alpha, \mathbf{m}, \mathbf{S}$
    else
      $\breve{\mathbf{x}}_t \leftarrow$ Algorithm2()
    end if
    Augment $\breve{\mathbf{x}}$ with $\breve{\mathbf{x}}_t$
    Augment $\mathbf{m}, \mathbf{S}$ with size($\breve{\mathbf{x}}_t$) random
  end while
  Augment $(\mathbf{x}, y)$ with $(\mathbf{x}_t, y_t)$
  Optimize $\gamma, \sigma, \mathbf{m}, \mathbf{S}$ w.r.t. $\mathcal{L}$ - eq. (1)
end function

Algorithm 2 Choosing inducing points

function GETIND($\mathbf{x}_*, y_*, \mathbf{x}, \breve{\mathbf{x}}, \gamma, \alpha, \mathbf{m}, \mathbf{S}, \zeta$)
  Initialize $\mathbf{x}_{\text{informative}}$ as a null matrix
  Initialize $N_t \leftarrow$ length($\mathbf{x}$)
  for $i = 1$ to N_t **do**
    $\mu_{*_i} \leftarrow$ eq. (2)
    $\sigma_{*_i} \leftarrow$ eq. (3)
    $\text{dist}_i \leftarrow |y_{*_i} - \mu_{*_i}|/\sigma_{*_i}$
    if $\text{dist}_i \geq \zeta$ **then**
      Augment $\mathbf{x}_{\text{informative}}$ with $\mathbf{x}_{*_i}$
    end if
    Augment $\mathbf{x}_{\text{informative}}$ with $\mathbf{x}_{*_i}$
  end for
end function

3.2.1 Filtering uninformative data

Intelligent selection (Algorithm 2) is run for each new scan after the very first scan. Considering the t^{th} time step, N_t data points $\{\mathbf{x}_i, y_i\}_{i=0}^{N_t}$, both occupied and unoccupied are obtained from the sensor. Variational parameters and hyper-parameters have been optimized sequentially from time steps 0 to $t-1$. Based on these previously optimized parameters, we use the laser locations of the current scan $\mathbf{x}_i$ and query the predictive mean and variance. Then, the distance, $\text{dist}[y_i, p(y_*|x_*; \mu_{*_i}, \sigma_{*_i})] = |y_i - \mu_{*_i}|/\sigma_{*_i}$, is used as a metric to measure how influential the individual data point in the new laser scan to make a change in our model. This metric indicates dynamic areas which have been underrated in previous time steps. The distance values above a user-defined threshold ζ are considered as informative data points and such data points indicate candidate areas for inducing points.

3.2.2 Clustering filtered informative data

Our objective is to select highly representative inducing points. Since physical objects have several laser returns, it is ineffective to include all informative data (3.2.1) as inducing points. Therefore, at this stage, we *cluster* informative data and choose cluster centroids as inducing points.

The number of clusters depends on the number of dynamic objects in the environment and how well the model has been learned so far. Therefore, we cannot use the ubiquitous k-means algorithm. Here, we adopt the seminal Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm [4] that automatically decides the number of clusters by itself based on data density. As shown in Fig. 2d, centroids of these clusters are set as inducing point locations $\breve{\mathbf{x}}_t$. This essentially avoids the inclusion of extraneous inducing points making M significantly smaller.

3.3 Optimizing the lower bound

Now, the goal is to find the sparse and approximate posterior $q(\breve{\mathbf{f}}) = \mathcal{N}(\breve{\mathbf{f}}|\mathbf{m}, \mathbf{S})$ using the lower bound [6] given in (1),

$$\mathcal{L} = \sum_{n=1}^{N} \Big\{ \mathbb{E}_{q(f(\breve{\mathbf{x}}_n))}[\log p(y_n|f_n)] - \mathbb{KL}[q(\breve{\mathbf{f}}) \| p(\breve{\mathbf{f}})] \Big\}, \tag{1}$$

where $q(f(\breve{\mathbf{x}}_n))$ indicates marginals of $q(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mu, \mathbf{\Sigma})$ with,

$$\mu = K_{NM} K_{MM}^{-1} \mathbf{m}, \tag{2}$$

$$\mathbf{\Sigma} = K_{NN} + K_{NM} K_{MM}^{-1} (\mathbf{S} - K_{MM}) K_{MM}^{-\top} K_{NM}. \tag{3}$$

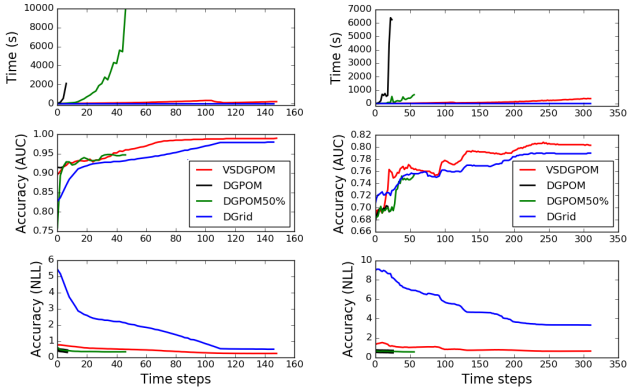
Learning the model requires the optimization of 1) variational parameters $\mathbf{m}$ and $\mathbf{S}$ and; 2) hyper-parameters of the kernel α and γ. Although (1) looks cumbersome, $\mathbb{KL}$ term can be analytically evaluated with a computational cost of $\mathcal{O}(M^3)$, while the $\mathbb{E}_{q(f_n)}$ term can be computed using 1D Gaussian quadratures. Note that its a summation over individual data points and hence *stochastic gradient descent (SGD)* can be effectively used for each new laser scan, making it suitable for big data sets. When compared with conventional GPs and GPOMs whose computational cost is $\mathcal{O}(N^3)$ with $M \ll N$, this a significant improvement to the speed of the framework, assuming inducing points are chosen using Algorithm 2. Once variational parameters and kernel hyperparameters are optimized, the predictive occupancy map can be generated by evaluating $\int p(\mathbf{f}_* \mid \breve{\mathbf{f}}) q(\breve{\mathbf{f}}) \mathrm{d}\breve{\mathbf{f}}$ any location in the continuous longitude-latitude space.

In summary, we describe how to utilize the sparse variatianal GP framework to build long term maps in dynamic environments. Two approximations: 1) variational approximation to obtain the posterior 2) nyström low rank approximation [20] to represent the covariance matrix using inducing point selection method were utilized to achieve a significant improvement in time compared to DGPOM.

4 EXPERIMENTS

In order to eliminate any confounding factors, and without loss of generality, the robot was kept stationary in all experiments. A moving robot has no detrimental effect on our algorithm. All algorithms were prototyped in python and executed in a 8 GB RAM laptop. Two datasets were used to demonstrate the speed and accuracy performance of the algorithm. Dataset 1 (Fig. 1) was obtained from a laser simulator. Dataset 2 [10] is a real four-way busy traffic intersection where vehicles move in various directions obeying traffic light signals. We used 1) area under ROC curve (AUC) and, 2) negative log loss (NLL), $-\log p(y|y_*)$ [2], to evaluate the accuracy of our method. NLL is a more representative measure of the robustness of these models [2] because it takes the probabilities of predictions into account. The smaller the NLL, the better the model is. ζ, the only free parameter of the proposed method, was kept constant at 5 through out all experiments. For DGrid, the optimal grid resolutions were chosen in advance by *grid search* that maximizes the AUC.

Data frames representing the past and future were randomly selected as the *test dataset* and it was never used for training. To maintain the class-balance, each frame of the test dataset had equal number of occupied and unoccupied points. At each time step, the entire test dataset was used to query the model and, AUC and NLL were calculated.



(a) Dataset 1 (b) Dataset 2

Table 1: Accuracy of occluded areas for dataset 1

Method	AUC ($\mu \pm 2\sigma$)
DGrid	0.50 ± 0.00
DGPOM	0.99 ± 0.02
VSDGPOM	$\mathbf{1.00 \pm 0.00}$
Method	**NLL ($\mu \pm 2\sigma$)**
DGrid	0.69 ± 0.00
DGPOM	0.31 ± 0.13
VSDGPOM	$\mathbf{0.02 \pm 0.16}$

Figure 3: Speed and accuracy performance of sequential learning. For each new laser scan, the models update their parameters and calculate accuracy against a test dataset which represents randomly chosen samples from the past and future. Compared to DGPOM, our model has an incredible improvement in speed (top row) for a similar accuracy (middle and center rows). For instance DGPOM50% (DGPOM with 50% data) takes 2 hours to update the map around $t = 40$ in dataset 1. Though DGrid is also fast, it has a low accuracy and a sluggish learning curve, especially when the probabilistic measure (NLL) is considered.

As shown in Fig. 3, VSDGPOM and DGPOM learn the correct occupancy probability within few iterations while DGrid has a poor learning curve (compare red and blue NLL curves). The observation that the accuracy measures of DGPOM is slightly better than VSDGPOM (note that NLL is a log-based metric) completely makes sense because VSDGPOM is an approximation to the Gaussian process whereas DGPOM is a full-rank matrix. This slight drop in spatio-temporal accuracy is negligible when compared to the time saved. The first row of Fig. 3 clearly shows that VSDGPOM does not have an exponentially increasing time as in DGPOM. This was possible because the speed of the core algorithm depends on the size (M) and quality of inducing points rather than the amount of data (N) collected. Note that DGPOM and DGPOM50% were automatically stopped after several time steps due to memory limitations as each dataset has at least 150,000 data points and hence it cannot capture long-term dynamics over a long period. In contrast, VSDGPOM can handle large datasets thanks to the information-driven intelligent inducing point selection algorithm.

Then, we manually labeled occluded areas and the accuracy was calculated to show the algorithm's robustness against occlusions (Table 1). This was possible only for dataset 1 as determining occluded areas exactly is only possible for a simulation dataset. Unsurprisingly, DGrid has an AUC of exactly 0.5 which is equivalent to a random guess. In contrast, DGPOM and VSDGPOM have an accuracy close to 1 because the kernels can interpolate based on neighbor points. In VSDGPOM, NLL is comparably smaller. These results are apparent when comparing occluded areas behind parked vehicles in Fig. 1. In contrast to many other techniques, our method did not rely on separate object tracking algorithms nor manual parameter tuning. These tasks were both embedded within the method.

5 FUTURE WORK

Probabilistic representations are quintessential to make path planning more reliable and safer [9]. Therefore, the long-term maps proposed in this paper will be incorporated with short-term maps [14] to make probabilistic path planing [7] more robust.

References

[1] Daniel Arbuckle, Andrew Howard, and Maja Mataric. Temporal occupancy grids: a method for classifying the spatio-temporal properties of the environment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 409–414, 2002.

[2] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] Alberto Elfes. *Occupancy grids: a probabilistic framework for robot perception and navigation*. PhD thesis, Carnegie Mellon University, 1989.

[4] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, volume 96, pages 226–231, 1996.

[5] Dirk Hahnel, Rudolph Triebel, Wolfram Burgard, and Sebastian Thrun. Map building with mobile robots in dynamic environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4270–4275, 2003.

[6] James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable variational gaussian process classification. In *the Eighteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

[7] Zita Marinho, Byron Boots, Anca Dragan, Arunkumar Byravan, Geoffrey J Gordon, and Siddhartha Srinivasa. Functional gradient motion planning in reproducing kernel Hilbert spaces. In *Proceedings of Robotics: Science and Systems (RSS)*, 2016.

[8] Daniel Meyer-Delius, Maximilian Beinhofer, and Wolfram Burgard. Occupancy Grid Models for Robot Mapping in Changing Environments. In *proc. AAAI Conference on Artificial Intelligence (AAAI)*, pages 2024–2030, 2012.

[9] Mohammad Norouzii, Jaime Valls Miro, and Gamini Dissanayake. Probabilistic stable motion planning with stability uncertainty for articulated vehicles on challenging terrains. *Autonomous Robots*, 40(2): 361–381, 2016.

[10] Simon O'Callaghan and Fabio Ramos. Gaussian process occupancy maps for dynamic environments. In *Experimental Robotics*, pages 791–805. Springer, 2016.

[11] Simon O'Callaghan, Fabio Ramos, and Hugh Durrant-Whyte. Contextual occupancy maps using Gaussian processes. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3630–3636, 2009.

[12] Fabio Ramos and Lionel Ott. Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent. In *Proceedings of Robotics: Science and Systems (RSS)*, Rome, Italy, 2015.

[13] Carl E Rasmussen and Christopher K I Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[14] Ransalu Senanayake, Lionel Ott, Simon O'Callaghan, and Fabio Ramos. Spatio-temporal Hilbert maps for continuous occupancy representation in dynamic environments. In *Neural Information Processing Systems (NIPS)*, 2016.

[15] Michalis K Titsias. Variational learning of inducing variables in sparse gaussian processes. In *the Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 12, pages 567–574.

[16] Aisha Walcott. *Long-term robot mapping in dynamic environments*. PhD thesis, Massachusetts Institute of Technology, 2011.

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

[18] Zhan Wang, Patric Jensfelt, and John Folkesson. Modeling spatial-temporal dynamics of human movements for predicting future trajectories. In *AAAI Conference on Artificial Intelligence*, pages 42–48, 2015.

[19] Christopher K I Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Neural Information Processing Systems (NIPS)*, 2000.

[20] Christopher K I Williams and Matthias Seeger. The effect of the input density distribution on kernel-based classifiers. In *International Conference on Machine Learning (ICML)*, pages 1159–1166, 2000.