# Automorphing Kernels for Nonstationarity in Mapping Unstructured Environments

**Ransalu Senanayake**[*]
The University of Sydney

**Anthony Tompkins**[*]
The University of Sydney

**Fabio Ramos**
The University of Sydney

**Abstract:** In order to deploy robots in previously unseen and unstructured environments, the robots should have the capacity to learn on their own and adapt to the changes in the environments. For instance, in mobile robotics, a robot should be able to learn a map of the environment from data itself without the intervention of a human to tune the parameters of the model. To this end, leveraging the latest developments in automatic machine learning (AutoML), probabilistic programming, and statistical sampling, under the Hilbert mapping framework which can represent the occupancy of the environment as a continuous function of locations, we formulate a Bayesian framework to learn all parameters of the map. Crucially, this way, the robot is capable of learning the optimal shapes and placement of the kernels in Hilbert maps by merely embedding high-level human knowledge of the problem by means of prior probability distributions. Since the proposed framework employs stochastic variational inference, the model learns tens of thousands of parameters within minutes in both big data and data-scarce regimes. Experiments conducted on simulated and real-world datasets in static and dynamic environments indicate the proposed method significantly outperforms existing stationary occupancy mapping techniques, verifying the importance of learning the interdependent position-shape relationship of kernels alongside other model parameters.

**Keywords:** occupancy mapping, autoML, probabilistic programming, RKHS

## 1 Introduction

Modeling the environment a robot operates in is fundamental to safer decision-making such as path planning. To this end, discerning occupied areas from unoccupied areas of the environment using sensor measurements such as lidar or sonar is required. Typically, these occupancy states exhibit highly nonlinear and spatially correlated patterns that cannot be captured with a simple linear classification model. Therefore, from a machine learning perspective, deep learning models and kernel-based models can be regarded as the potential candidates for occupancy mapping as they are known to perform well in nonlinear classification settings. Nonetheless, because it is required to learn the occupancy level using very few sparse sensor measurements in a reasonable time, kernel methods have been the *de jure* choice in recent occupancy mapping [1, 2]. Further, experiments have corroborated their promising applications in 2D, 3D, and spatiotemporal mapping [1, 2, 3, 4], though with some heuristic parameter choices.

One of the major challenges in employing kernel methods in occupancy mapping is the requirement of choosing parameters and hyperparameters of the model [5]. In order for mobile robots to maneuver fully autonomously in unknown environments or to interact with humans and other agents, the robots should have the capability to automatically learn their model parameters from data. Only the most simple environments contain spatially homogenous features, however this is typically not the case in real-world mapping - e.g. walls and furniture may contribute to sharp features while open

[*] Equal contribution. E-mail: `{ransalu.senanayake,anthony.tompkins,fabio.ramos}@sydney.edu.au`

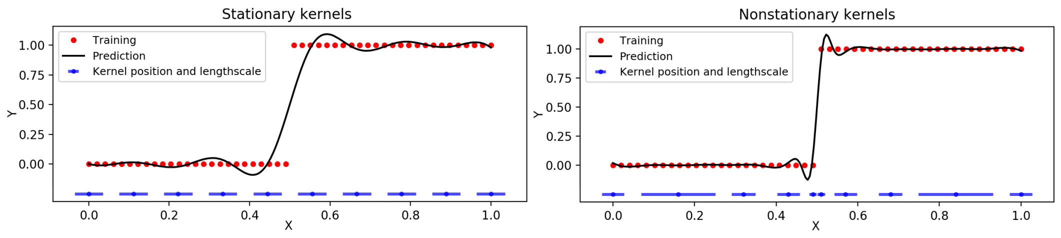

Figure 1: Comparison of stationary and nonstationary squared-exponential kernels, $\exp(-\|\mathbf{x}-\tilde{\mathbf{x}}\|_2^2/2l^2)$ with bivariate Gaussian distributions $\tilde{\mathbf{x}}$ hinged on the environment with lengthscales $l$, and their ability to represent sharp spatial changes. Note that both examples have the same number of kernels, however in the non-stationary case the kernels have different positions and lengthscales to account for abrupt changes in the training data.

spaces and large hills may contribute to spatially smooth features. To better understand the significance of representing nonstationarity in terms of kernels, first consider the squared-exponential kernel which is parametrized with lengthscale and position hyperparameters. As seen in Figure 1 with large lengthscales it is possible to capture smoother changes across the space, while small lengthscales allow one to capture sharp changes in the space. Hyperparameter optimization is critical for almost all machine learning methods and the best values are almost always dependent on the dataset. Often, a single best lengthscale is chosen that performs, on average, the best for the entire dataset. There is also a computational constraint of the modeling method that restricts one from placing a large number of kernels around the space being mapped - this introduces the problem of where one should place kernels. In our contribution, we address both problems - where to place kernels and what lengthscales they should have. These autodidactic and model adaptation paradigms are vital for a robot to achieve full autonomy.

Another important aspect that should be taken into account when designing robot models is the uncertainty inherent to all system levels—from sensor and actuator imperfections to model misspecifications. To represent this uncertainty, probabilistic formulations [6, 7] have been widely used in robotic applications such as simultaneous localization and mapping [8], occupancy grid mapping [9], and human-robot interaction [10, 11, 12]. Although these models make use of the Bayes theorem to compute inverse probabilities assuming the conjugacy of prior-posterior probability distribution pairs [13], they do not leverage the fully Bayesian treatment of introducing probability distributions over all parameters to account for diverse sources of uncertainty. In addition to capturing uncertainty, Bayesian models are known to be suitable for small data settings [14]. Another incentive to use Bayesian models in robotics is that they provide an interface to incorporate high-level human knowledge about the system into the model through prior probability distributions. Irrespective of the attractive proprieties of Bayesian formulations, considering complex Bayesian treatments to capture all levels of uncertainties has been hindered in many robotics applications because, i) designing tractable Bayesian models require tedious mathematical derivations which require subject expertise, and ii) many models cannot scale to run in real-world robotics applications. However, recent developments in AutoML techniques in machine learning can circumvent some of these issues, making them propitious for robotics applications such as kernelized continuous occupancy mapping.

In this paper, we use stochastic gradient descent with the reparameterization trick [15] to solve a challenging robot learning problem of determining parameters for Hilbert maps. Former Hilbert mapping techniques, including the Bayesian version, have used human designed kernel parameters. As illustrated in Figure 2, the most important parameters which include hinge positions and lengthscales can be learned. The contributions of the paper are:

1. Proposing a theoretical framework that works well in practice to learn all parameters in Hilbert maps in both static and dynamic environments,
2. Learning kernels to account for non-stationary and nonlinear patterns,
3. Proposing the use of low-discrepancy sampling in robotic mapping,
4. Demonstrating the importance of using complex Bayesian formulations for uncertainty representation in robotics and learning thousands of parameters in both small and bigdata settings without laborious mathematical derivations, and
5. A thorough analysis of factors that affects Hilbert mapping.

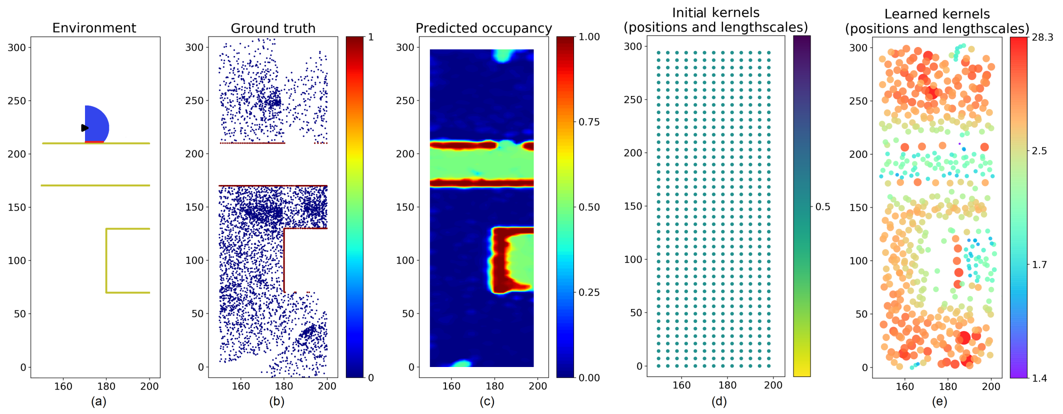

Figure 2: (a) A 50 $\times$ 300 m section of a simulated environment with obstacles in yellow. A robot shown as a black arrow has a lidar with beams shown in blue and the laser hit points in red. (b) The robot moves around and collects data. Red points are laser hit points and blue points are samples taken from lidar beams between the robot and laser hit points (c) The occupancy probability map. Red indicates occupied space, blue indicates free space, and colors in-between indicates the uncertainty of occupancy. (d) The map is built based on a set of squared-exponential kernels. The mean of the initial bivariate Gaussians is shown here—Gaussians are in a grid. (e) The proposed algorithm can learn both kernel parameters $l$ and positions $\tilde{\mathbf{x}}$ alongside other model parameters. Both the color and the size of the marker indicates the size of the learned lengthscales. For instance, larger lengthscales are shown in a bigger marker size and in red.

## 2 Background

### 2.1 Hilbert maps

With the advancement of depth sensors such as lidar and sonar, occupancy grid maps (OGM) developed in 1980s [16] became a popular choice for representing the environment. In OGMs, the world is divided into a fixed size grid with a prespecified resolution, after which, a Bayes filter is applied to sequentially update the occupancy of grid cells as new depth information is received from the sensors. In order to capture spatial correlations, that are otherwise disregarded in OGMs due to independence assumptions among grid cells, O'Callaghan et al. [17] proposed Gaussian process occupancy maps (GPOMs) where the spatial correlation is captured using a kernel function. However, vanilla Gaussian process-based methods have $\mathcal{O}(N^3)$ runtime complexity for $N$ data points in both learning and prediction, and hence they are not efficient enough to map real-world environments. The computational complexity grows over time even in faster approximations [18]. As an alternative, Hilbert maps (HMs), another kernelized technique, was proposed [1]. Unlike GPOMs, HMs are parametric techniques and therefore scalable to large datasets.

In Hilbert maps, the map is learned on a reproducing kernel Hilbert space (RKHS) where kernel functions are used to characterize spatial relationships. In the context of HMs, a kernel $k(\mathbf{x}, \tilde{\mathbf{x}}) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a function that measures the similarity between two multidimensional inputs $\mathbf{x}$, $\tilde{\mathbf{x}} \in \mathcal{X} \subset \mathbb{R}^2$. In 2D HMs, the pairwise similarities between the elements of the two sets of points $\{\mathbf{x}_n \in \mathbb{R}^2\}_{n=1}^N$ and $\{\tilde{\mathbf{x}}_m \in \mathbb{R}^2\}_{m=1}^M$ are computed. Here, $\mathbf{x}$ are longitude-latitude locations of either free or occupied $y \in \{0, 1\} = \{\text{free}, \text{occupied}\}$ data points sampled from lidar beams and $\tilde{\mathbf{x}}$ are points *hinged* on pre-defined locations of the space. A squared-exponential (SE) kernel $k(\mathbf{x}_n, \tilde{\mathbf{x}}_m; l) = \exp\big(-\|\mathbf{x}_n - \tilde{\mathbf{x}}_m\|_2^2 / 2l^2\big)$ with a heuristically determined lengthscale $l$ is used to compute the the feature vector $\phi(\mathbf{x}_n; l) = \big(k(\mathbf{x}_n, \tilde{\mathbf{x}}_1; l), k(\mathbf{x}_n, \tilde{\mathbf{x}}_2; l), ..., k(\mathbf{x}_n, \tilde{\mathbf{x}}_M; l)\big) \in \mathbb{R}^{M \times 1}$ for all data points $\{\mathbf{x}_n\}_{n=1}^N$. In this sense, $\big\{(\mathbf{x}_n, y_n)\big\}_{n=1}^N$ is the dataset and $\big\{l, \{\tilde{\mathbf{x}}_m\}_{m=1}^M\big\}$ is the pre-defined parameter set. Although random Fourier features or Nyström features can also be used in occupancy mapping, hinge features are not only intuitive but also have shown better performance in online occupancy mapping [1] and other similar problems [19, 20].

Once the feature vector is evaluated, it passes through a sigmoidal function to estimate the occupancy level $\hat{y} = p(y|\mathbf{x}_*, \mathbf{w}) = 1/(1+\exp(\mathbf{w}^\top \phi(\mathbf{x}_n; l)))$ of a query point in the space $\mathbf{x}_*$, given the weights $\mathbf{w} \in \mathbb{R}^{M \times 1}$. As this query point can be any longitude-latitude pair, as opposed to OGMs, HMs can produce maps with arbitrary resolution at prediction time. In order to learn weights, the loss function

$\sum_{i=1}^{N} \log(1+\exp(y_n \mathbf{w}^\top \Phi(\mathbf{x}_n; l, \tilde{\mathbf{x}}))) + \lambda_1 \|\mathbf{w}\|_2^2 + \lambda_2 \|\mathbf{w}\|_1$ with pre-determined penalty terms $\lambda_1$ and $\lambda_2$ is minimized. This becomes challenging as a human expert requires to choose the parameters $l$, $\tilde{\mathbf{x}}$, $\lambda_1$, and $\lambda_2$. Although Senanayake and Ramos [5] attempted to alleviate this issue to a certain extent by getting rid of $\lambda_1$ and $\lambda_2$ which mainly controls over-fitting, the lengthscales of the kernel $l$ and where to place them $\tilde{\mathbf{x}}$ were still prefixed values. On the other hand, although the model in [5] works well if $l$ and $\tilde{\mathbf{x}}$ are chosen appropriately, the model is based on a hand-derived approximate lower bound which is not amenable to further modifications such as automatically learning these parameters. As an attempt to reduce the number of unnecessary hinged points in 3D point clouds [21] placed kernels using a clustering technique in a preprocessing step.

### 2.2 Kernel learning

Kernels methods are used in robotics especially when the objective is to learn nonlinear patterns with a small amount of data [22, 23, 19, 24]. Although only SE kernels with fixed lengthscales are used in robotic mapping [1, 2, 5], different kernel learning techniques have been previously discussed in machine learning, especially in the Gaussian process literature. The selection of kernels is typically done through expert human knowledge [25], a model selection criteria such as Bayesian information criteria [26], or expensive optimization procedures [27]. Alternatively, it is possible to combine kernels as a sum or a product of kernels [25] or as representing them as a spectral mixture in the frequency domain [28]. However, unlike in Gaussian process where optimizing the hyperparameters is well-studied and readily available through the log marginal likelihood, directly learning parameters online in a classification setting is not straightforward in HMs.

## 3 Nonstationary kernels for Hilbert mapping

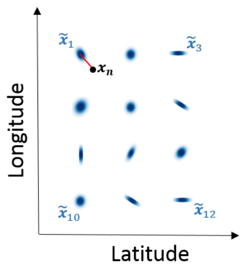

Figure 3: Feature vector computation. $\{\tilde{\mathbf{x}}\}_{m=1}^{M=12}$ are hinge distributions and $\mathbf{x}_n$ is the $n^{th}$ data point.

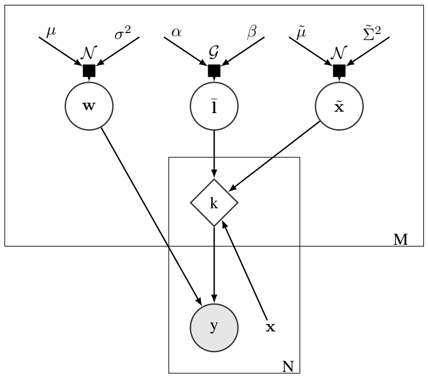

Figure 4: The graphical model. $k$ represents the kernel which is evaluated $N \times M$ times.

Table 1: Description of model parameters. Assuming independence, individual distributions are associated with all hinge points $m = 1...M$

| Var. | Distributions |
|---|---|
| Prior distributions: | |
| $w_m$ | $\mathcal{N}(\mu_m, \sigma_m^2)$ |
| $\bar{l}_m$ | $\mathcal{G}(\alpha_m, \beta_m)$ [1] |
| $\tilde{\mathbf{x}}_m$ | $\mathcal{N}(\tilde{\mu}, \tilde{\Sigma}^2)$ |
| Variational distributions: | |
| $qw_m$ | $\mathcal{N}(\mu_m^{(q)}, \sigma_m^{2(q)})$ |
| $q\bar{l}_m$ | $\mathcal{LN}(\alpha_m^{(q)}, \beta_m^{(q)})$ [2] |
| $q\tilde{\mathbf{x}}_m$ | $\mathcal{N}(\tilde{\mu}^{(q)}, \tilde{\sigma}^{2(q)})$ |

It is known that kernel methods are well suited for occupancy mapping, if parameters are appropriately set [1, 2]. In this section, we propose novel techniques for mapping unstructured environments without a human explicitly providing hyper-parameters. Firstly, we propose different positioning techniques for hinging kernels in Section 3.1. Then, we discuss the importance of nonstationary learning [29] in occupancy mapping. That is, rather than having a single lengthscale for all kernels as in [1, 5], kernels should have different parameters, depending on where they are placed in the space. For instance, with regards to SE kernels, kernels should have smaller lengthscales close to walls in order to capture sharp transitions from occupied to unoccupied. Similarly, bigger lengthscales are expected in largely unoccupied or unobserved areas. With the motivation from Section 3.1, we propose a technique to simultaneously learn the position and kernel parameters in Section 3.2.

[1]The gamma distribution is defined as $\mathcal{G}(x; \alpha, \beta) := \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$ where $\Gamma(\alpha) := \int_0^\infty z^{\alpha-1} e^{-z} \mathrm{d}z$ is the gamma function. $\alpha > 0$ is the shape parameter and $\beta > 0$ is the rate parameter. In literature, the scale parameter is sometimes defined as the inverse of the rate parameter instead of the rate parameter.

[2]The log-normal distribution $\mathcal{LN}$ is obtained by transforming a standard normal variable $z' = \exp(\mu + \sigma z)$.

### 3.1 Various initializations for hinged kernels

Rather than initializing hinge points randomly in the space, it is possible to place them based on quasi-random locations in order to guarantee a more uniform spread of kernels in the space. This is done by using a low-discrepancy sequence [30]. Some of the commonly used Quasi-Monte Carlo (QMC) sampling techniques include Halton, generalized Halton, and Sobol sequences [30, 31].

SE kernels with a fixed lengthscale have been the *de facto* choice in Hilbert mapping [1, 2, 32, 5]. We propose to hinge multiple kernels with varying properties on every hinge location $\tilde{\mathbf{x}}_m$ rather than hinging a single kernel. For instance, it is possible to hinge a set of SE kernels $\{k(\cdot, \tilde{\mathbf{x}}_m; l_r)\}_{r=1}^{R}$ with $R$ different lengthscales $l_r$. More broadly, these can also be a set of $R$ different kernel types $\{k_r(\cdot, \tilde{\mathbf{x}}_m; \theta_r)\}_{r=1}^{R}$ with $\theta_r$ parameters corresponding to the the $r^{\text{th}}$ kernel. In addition to the hackneyed SE kernels, this pool of kernels can consist of Matérn kernels, rational quadratic kernels, etc. A spectral mixture of kernels [33] is also another choice.

Adding hinged kernels increases the dimensionality of the feature vector by $R$ times. For $\Phi_r$ feature vectors individually computed on $M$ hinge points as in Section 2.1, define the aggregated feature vector $\Phi_\Sigma = \|_{r=1}^{R} \Phi_r \in \mathbb{R}^{1\times RM}$ with $\|$ indicating vector concatenation. The classification model is $\hat{y} = p(y|\mathbf{x}_*, \mathbf{w}_\Sigma) = 1/(1+\exp(-\mathbf{w}_\Sigma^\top \Phi_\Sigma))$ with $\mathbf{w}_\Sigma \in \mathbb{R}^{RM\times 1}$. This is equivalent to joining $R$ individual sets of model weights $\mathbf{w}_r \in \mathbb{R}^{M\times 1}$ as $\sum_{r=1}^{R} \mathbf{w}_r^\top \Phi_r$ before the sigmoidal transformation.

### 3.2 Automorphing kernels for Hilbert mapping

Although the aggregation method proposed in Section 3.1 partially accounts for nonstationarity, it requires a set of predefined parameters. As with other continuous mapping techniques, the method cannot learn where to place kernels. However, for a robot to adapt to changes in unstructured environments, it is crucial to take the human out of the loop of parameter tuning. In the following sections, without loss of generality to other kernels, we explain using SE kernels to solve these issues.

#### 3.2.1 Model specification

In this section, as the main contribution of this paper, we propose a principled approach to provide two traits of adaptation to kernels in Hilbert maps: plasticity and mobility. That is, both the shape $l$ of the kernel and its locations $\tilde{\mathbf{x}}$ can be learned alongside feature weights $\mathbf{w}$ under the proposed framework. Further, as shown in Figure 2e, rather than considering a single lengthscale as in previous work [5] or a small set of lengthscales as in Section 3.1, the new technique can not only learn any lengthscale in $\mathbb{R}$, but also the kernels associated with each hinge location has its own local lengthscale. These individual lengthscales $\{l_m\}_{m=1}^{M}$ essentially model the nonstationary behavior and can easily acclimatize to local changes in the environment.

Since observed occupancy values are always binary and they are independent of each other, we assume the likelihood follows a Bernoulli distribution $p(\mathbf{y}|\mathbf{x}, \mathbf{w}, \mathbf{l}, \tilde{\mathbf{x}})$ where $\log(\theta/(1-\theta))) = \mathbf{w}^\top \Phi(\mathbf{x}; \mathbf{l}, \tilde{\mathbf{x}})$. Note that this is equivalent to logistic regression [13]. The prior distributions over weights are defined as Gaussian distributions. Since the hinge locations can be anywhere in the space, they are also defined as Normal distributions. As shown in Figure 3, kernel functions are now implicitly evaluated between datapoints point and hinge distributions, naturally accounting for uncertainty. A meaningful lengthscale can only be positive and hence the prior distribution over inverse squared-lengthscales $\bar{l} = 1/2l^2$ is defined as a gamma distribution. The first three rows of Table 1 provides a summary of all variables. Here, for computational efficiency we assume all variables are independent. As these prior distributions were empirically sufficient for modeling occupancy, we do not complicate the model with hyper-prior distributions.

Our objective is to learn the posterior distribution: parameters conditioned on data. However, because of the Bernoulli likelihood, the posterior is intractable and hence is approximated using another distribution $q$. Indicating longitude and latitude with lon and lat, respectively, the basic formulation with mean-field variational approximation is given in Figure 3 and the following equation,

$$\underbrace{\prod_{m=1}^{M} q(w_m)q(l_m^{\text{lon}})q(l_m^{\text{lat}})q(\tilde{\mathbf{x}}_m)}_{\text{factorized variational distribution}} = \underbrace{q(\mathbf{w}, \mathbf{l}, \tilde{\mathbf{x}})}_{\substack{\text{variational}\\ \text{distribution}}} \approx \underbrace{p(\mathbf{w}, \mathbf{l}, \tilde{\mathbf{x}}|\mathbf{x}, \mathbf{y})}_{\text{posterior}} \propto \underbrace{p(\mathbf{w})p(\mathbf{l})p(\tilde{\mathbf{x}})}_{\text{priors}} \underbrace{p(\mathbf{y}|\mathbf{x}, \mathbf{w}, \mathbf{l}, \tilde{\mathbf{x}})}_{\text{likelihood}}.$$

### 3.2.2 Model learning

The posterior distribution defined in Section 3.2.1 is intractable due to the Bernoulli likelihood. Since distributions over lengthscales and positions are introduced in addition to distributions over weights, obtaining a maximum a posteriori (MAP) estimation is not feasible even with the lower bound derived in [34]. As occupancy mapping is a very high dimensional problem, obtaining the posterior using Markov chain Monte Carlo (McMC) techniques is costly [13]. As an alternative, we use variational inference (VI) with the reparameterization trick [15]. Although there are other alternatives such as VI with stochastic search [35] and Hamiltonian Monte-Carlo [36], we used the well-established method [15] as it can easily perform stochastic gradient descent (SGD) with minibatches. Rather than deriving the lower-bound for the specific case, as an AutoML technique, we made use of probabilistic programming [37] to minimize the Kullback-Leibler (KL) divergence between the variational distribution and posterior $\mathbb{KL}(q \| p)$. For this reason, the proposed model is easily amenable for extensions. The choice of distributions is detailed in Table 1. To keep variances of variational distributions non-negative a softplus transformation was applied.

## 4 Experiments and discussions

We conducted a series of experiments on four different datasets given in Table 2. These datasets contain both static and dynamic environments. As with [5, 18], our model will estimate the average long-term occupancy which is different to mapping short-term occupancy [3] or removing dynamics to build a static occupancy map [38, 39]. When demonstrating basic concepts and observations, a portion of dataset 1 is used as it is simple and easy to visualize. We used TensorFlow with the Edward library [40] to program. Demonstrations can be found at github.com/MushroomHunting/automorphing-kernels.

Table 2: Description of the datasets.

| Dataset | Real | Dynamic | Description |
|---|---|---|---|
| 1 | ✗ | ✗ | A $600 \times 300\text{m}^2$ area [5]. This is a simple but large environment. |
| 2 | ✓ | ✗ | Intel lab dataset: a complex indoor environment. |
| 3 | ✗ | ✓ | Vehicles move in two directions and the robot sits in the middle [5]. |
| 4 | ✓ | ✓ | Lidar dataset in a busy intersection [5]. |

### 4.1 Experiment 1: Effect of kernel aggregation

As the first experiment, to verify that capturing non-stationarity is important, we hinged three SE kernels with $l = 5, 1, 0.1$ on fixed locations. We observe that by changing the lengthscale of all kernels we can observe differently learned weights - this indicates the requirement of learning kernel parameters depending on where they are in the environment. (See supplementary for experimental results).

### 4.2 Experiment 2: Effect of hinging techniques

In order to demonstrate the effect of positioning kernels, we learn the map for a fixed lengthscale chosen from five-fold cross-validation that minimizes AUC. We consider positioning kernels on a regular grid, MC samples, and three QMC sampling techniques—Halton, generalized Halton, and Sobol. The number of hinge kernels was set to 222. As shown in the supplementary materials, unlike the QMC techniques, MC sampling tend to make clusters making the Hilbert maps less accurate. Therefore, if kernel positions are not learned, they should be placed either quasi-randomly or on a regular grid.

For the intel dataset, as shown in Figure 8, we do further analysis to see the effect of the number of samples trade off. Interestingly, the time increases almost linearly with the number of features while the accuracy does not significantly improve after a certain number of features.

### 4.3 Experiment 3: Effect of learning parameters

This experiment was designed to validate the main contribution of the method—learning lengthscales and hinge locations. The learned environments for different datasets are shown in Figures 5

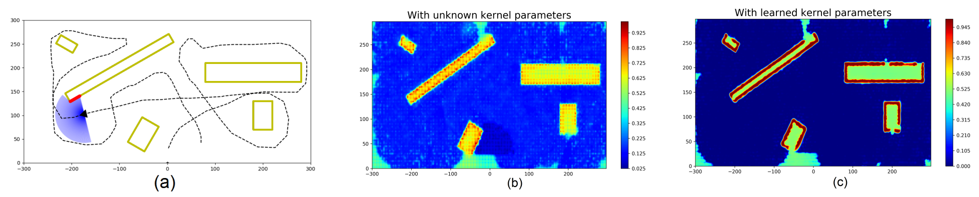

Figure 5: (a) Environment and entire dataset 1 (b) The averaged occupancy map of BHM with a random set of lengthscales (c) Predicted occupancy map using ABHM.

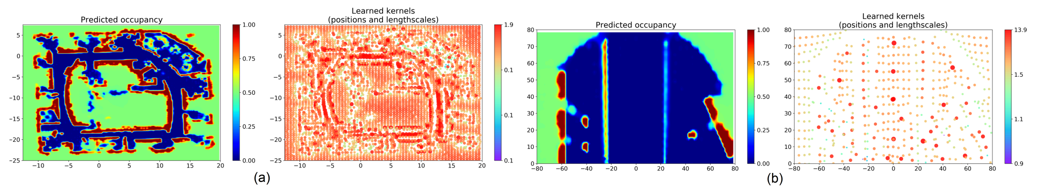

Figure 6: Predicted occupancy map and learned lengthscales. (a) Dataset 2 (b) Dataset 3

and 6 as well as in supplementary materials. To understand the full effect of the proposed model it is not enough to look at the predicted occupancy map—we must consider the underlying distributions. Figure 7 provides a visual map of the means and variances of a learned model's variational posteriors. Accounting for a large part of the upper and lower parts of the map, the position variance in Figure 7b shows that in areas of dense laser scans where no walls exist, a larger but uniform variance for each spatial dimension is learned. For the areas where the laser scanner has detected walls one observes a stark contrast exhibited by the smaller spatial variances. In the walled area spanning the middle of the map the learned variances in the latitudinal direction are stretched out further relative to the longitudinal direction reflecting the narrow corridor-like shape of the wall. Concerning now the lengthscale mean and variance in Figure 7c we can observe the most significant effect in terms of the learned posteriors. At the top and the bottom open areas the largest lengthscales are observed signifying a minimal complexity of occupancy. Paralleling the learned position variances, the learned lengthscale means are clustered around either areas of detail or areas of uncertain occupancy. This effect is repeated in the lengthscale variance.

The kernel weights means and variances are depicted in Figure 7d where one can see the highest weights appear around areas associated with the smallest position and lengthscale variances. Contrastingly, the most negative weights appear in regions of highly confident predicted empty occupancy. The weights closest to zero occur in areas of the map the robot has no visual perception and these constitute the insides of walls. The effect of the weight means is reflected in the weight variance where areas of high observability, which include open spaces and walls, have a low uncertainty in their estimates. Areas of low observability, i.e. inner parts of walls, have extremely high variances. This underlying analysis of the learned posterior distributions not only substantiates the motivation for spatially adaptive kernel learning, but also gives an explainable and intuitive un-

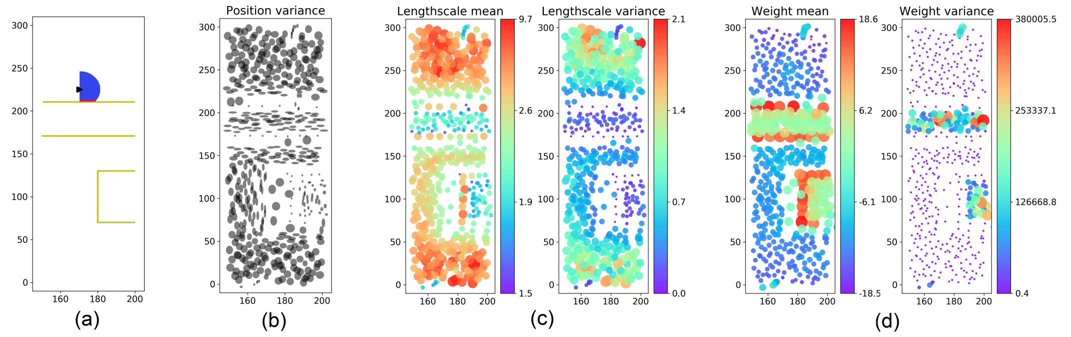

Figure 7: Uncertainty plots (a) A portion of the environment (b) Positions of hinge kernels $\tilde{\mathbf{x}}$ (c) Lengthscales (d) Weights

Table 3: Experiment 3: Losses on all real datasets. The higher the area under curve (AUC) or the lower the mean negative log loss (MNLL), the better the model is.

| Method | Dataset 1 | | Dataset 2 | | Dataset 3 | | Dataset 4 | |
|---|---|---|---|---|---|---|---|---|
| | AUC | MNLL | AUC | MNLL | AUC | MNLL | AUC | MNLL |
| ABHM | **0.999** | **0.015** | **0.994** | **0.093** | **0.993** | **0.175** | **0.889** | **0.477** |
| BHM | **1.000** | 0.176 | 0.921 | 0.362 | 0.990 | 0.280 | 0.825 | 0.570 |
| HM | 0.992 | 0.226 | 0.938 | 0.666 | 0.920 | 0.903 | 0.778 | 0.677 |
| VSDGPOM | 0.801 | 0.372 | 0.794 | 0.530 | 0.990 | 0.233 | 0.788 | 0.886 |
| DOGM | 0.792 | 0.593 | 0.901 | 0.744 | 0.980 | 0.495 | 0.779 | 3.449 |

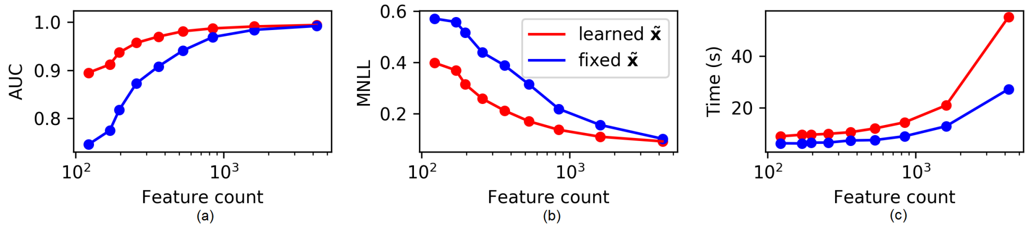

Figure 8: Performance vs. number of features for dataset 2. The blue lines show performance for fixed hinge positions while the red lines show the full ABHM model.

derstanding of what the model has learned which is often critically important for robotic tasks that interact with real-world environments.

Using all four datasets, the area under curve (AUC) and mean negative log loss (MNLL) were calculated. As reported in Table 3, these metrics were also calculated for occupancy grid maps with dynamic updates (DOGM), variational sparse dynamic Gaussian process occupancy maps (VSDGPOM) [18], HMs, and Bayesian Hilbert Maps with sequential updates (BHM). The best lengthscales for previous Hilbert mapping techniques were determined using five-fold cross validation. Even when compared with hand-crafted features, ABHM outperforms. This is because it models nonstationarity and can capture subtle changes. For dataset 1 which has straight boundaries, the AUC value of both BHM and ABHM are comparable. However, ABHM outperforms in complex datasets such as in dataset 2 and dynamic environments such as in datasets 3 and 4. This is because only ABHMs can adjust the position and shape of kernels to locally adapt to environments.

To further understand the relationship between the performance and number of hinge points, we analyzed the speed time and accuracy for dataset 2. We did this by, 1) learning both lengthscales and position, and 2) learning only the lengthscale keeping the kernels hinged on a grid. As shown in Figure 8, to achieve the same level of accuracy, only a smaller number of features is required when learning both the lengthscale and position.

**Runtime:** We conducted all experiments on a computer with a GTX1080 Ti 11 GB. For datasets 1 and 2, on average it takes around 10 minutes to learn all parameters. Note that this is to learn upwards of 57,600 parameters (8 parameters per hinge with more than 7200 hinges) and 300,000 data points. In contrast, [5] has an inevitable computational complexity $\mathcal{O}(M^3)$ while the proposed method uses stochastic gradient descent (SGD). Although analyzing the theoretical asymptotic complexity is not straightforward, it linearly increases with $M$ and $N$ empirically. In ABHM, we take the advantage of SGD to scalable for large datasets.

## 5 Conclusion

With the intention of building continuous occupancy maps without the human intervention, we devised methods to learn all parameters of the Hilbert maps. We also demonstrated the use of the latest AutoML techniques to learn complex models without relying on tedious mathematical derivations. Since kernel methods have also been successfully used in a variety of nonlinear path planning methods [22, 41, 32] we plan to extend these ideas to path planning so that mapping and path planning can be performed simultaneously in real-world in an end-to-end fashion under one framework.

## References

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

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

[3] R. Senanayake, L. Ott, S. O'Callaghan, and F. Ramos. Spatio-temporal hilbert maps for continuous occupancy representation in dynamic environments. In *Neural Information Processing Systems (NIPS)*, 2016.

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

[5] R. Senanayake and F. Ramos. Bayesian hilbert maps for dynamic continuous occupancy mapping. In *Conference on Robot Learning (CoRL)*, pages 458–471, 2017.

[6] S. Thrun. Probabilistic algorithms in robotics. *AI Magazine*, 21(4):93, 2000.

[7] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT press, 2005.

[8] M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on robotics and automation*, 17(3):229–241, 2001.

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

[10] J. Campbell and H. B. Amor. Bayesian interaction primitives: A slam approach to human-robot interaction. In *Conference on Robot Learning (CoRL)*, pages 379–387, 2017.

[11] V. Unhelkar, C. Guan, N. Roy, and J. Shah. Enabling robot teammates to learn latent states of human collaborators. 2018.

[12] M. A. Rana, M. Mukadam, S. R. Ahmadzadeh, S. Chernova, and B. Boots. Towards robust skill generalization: Unifying learning from demonstration and motion planning. In *Conference on Robot Learning (CoRL)*, pages 109–118, 2017.

[13] C. Bishop. Pattern recognition and machine learning (information science and statistics), 1st edn. 2006. corr. 2nd printing edn. *Springer, New York*, 2007.

[14] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian data analysis*. CRC press, 2013.

[15] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

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

[17] S. T. O'Callaghan, F. T. Ramos, and H. Durrant-Whyte. Contextual occupancy maps using Gaussian processes. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3630–3636, 2009.

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

[19] H. A. Kingravi, H. R. Maske, and G. Chowdhary. Kernel observers: systems-theoretic modeling and inference of spatiotemporally evolving processes. In *Advances in Neural Information Processing Systems*, pages 3990–3998, 2016.

[20] J. Whitman and G. Chowdhary. Learning dynamics across similar spatiotemporally-evolving physical systems. In *Proceedings of the 1st Annual Conference on Robot Learning (CoRL)*, volume 78 of *Proceedings of Machine Learning Research*, pages 472–481. PMLR, 13–15 Nov 2017.

[21] V. C. Guizilini and F. T. Ramos. Unsupervised feature learning for 3d scene reconstruction with occupancy maps. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 3827–3833, 2017.

[22] M. Mukadam, X. Yan, and B. Boots. Gaussian process motion planning. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 9–15. IEEE, 2016.

[23] M. Deisenroth and C. E. Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

[24] H. A. Kingravi, G. Chowdhary, P. A. Vela, and E. N. Johnson. Reproducing kernel hilbert space approach for the online update of radial bases in neuro-adaptive control. *IEEE Transactions on Neural Networks and Learning Systems*, 23(7):1130–1141, 2012.

[25] D. Duvenaud, J. R. Lloyd, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. *arXiv preprint arXiv:1302.4922*, 2013.

[26] D. Duvenaud. *Automatic Model Construction with Gaussian Processes*. PhD thesis, University of Cambridge, 2014.

[27] F. R. Bach, G. R. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *Proceedings of the twenty-first International Conference on Machine Learning (ICML)*, page 6. ACM, 2004.

[28] A. G. Wilson, E. Gilboa, A. Nehorai, and J. P. Cunningham. Gpatt: Fast multidimensional pattern extrapolation with gaussian processes. *arXiv preprint arXiv:1310.5288*, 2013.

[29] C. J. Paciorek and M. J. Schervish. Nonstationary covariance functions for gaussian process regression. In *Advances in Neural Information Processing Systems (NIPS)*, pages 273–280, 2004.

[30] J. Dick, F. Y. Kuo, and I. H. Sloan. High-dimensional integration: the quasi-monte carlo way. *Acta Numerica*, 22:133–288, 2013.

[31] J. Yang, V. Sindhwani, H. Avron, and M. Mahoney. Quasi-monte carlo feature maps for shift-invariant kernels. In *International Conference on Machine Learning (ICML)*, pages 485–493, 2014.

[32] G. Vallicrosa Massaguer and P. Ridao Rodríguez. H-slam: Rao-blackwellized particle filter slam using hilbert maps. *Sensors (Switzerland), 2018, vol. 18, núm. 5, p. 1386*, 2018.

[33] A. Wilson and R. Adams. Gaussian process kernels for pattern discovery and extrapolation. In *International Conference on Machine Learning (ICML)*, pages 1067–1075, 2013.

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

[35] J. Paisley, D. Blei, and M. Jordan. Variational bayesian inference with stochastic search. *arXiv preprint arXiv:1206.6430*, 2012.

[36] R. M. Neal. Probabilistic inference using markov chain monte carlo methods. 1993.

[37] D. Tran, M. D. Hoffman, R. A. Saurous, E. Brevdo, K. Murphy, and D. M. Blei. Deep probabilistic programming. In *International Conference on Learning Representations (ICLR)*, 2017.

[38] D. Meyer-Delius, M. Beinhofer, and W. Burgard. Occupancy Grid Models for Robot Mapping in Changing Environments. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

[39] C. Stachniss and W. Burgard. Mobile robot mapping and localization in non-static environments. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 1324–1329, 2005.

[40] D. Tran, A. Kucukelbir, A. B. Dieng, M. Rudolph, D. Liang, and D. M. Blei. Edward: A library for probabilistic modeling, inference, and criticism. *arXiv preprint arXiv:1610.09787*, 2016.

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

# Automorphing Kernels for Nonstationarity in Mapping Unstructured Environments -Supplementary Materials

**Ransalu Senanayake***
The University of Sydney

**Anthony Tompkins***
The University of Sydney

**Fabio Ramos**
The University of Sydney

## 1 Experiment 1: Effect of kernel aggregation

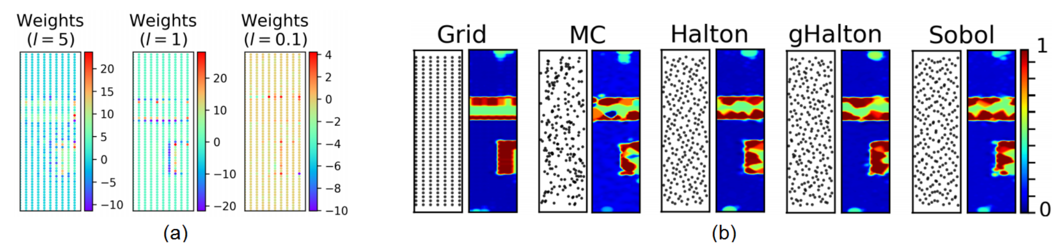

Figure 1: These images are based on the dataset in Figure 2a from main paper. (a) Kernel aggregation in experiment 1 (b) Positioning kernels in HMs with fixed position and lengthscales for different hinging schemes.

## 2 Experiment 3: Effect of learning parameters

Table 1: Example of model distribution parametrization for dataset1. $\tilde{\mathbf{x}}_{\text{init}}$ refers to some arbitrary hinge point initialization.

| Variable | Distributions |
|---|---|
| Prior distributions: | |
| $w_m$ | $\mathcal{N}(\mu_m = 0.0, \sigma_m^2 = 900)$ |
| $\bar{l}_m$ | $\mathcal{G}(\alpha_m = 1.05, \beta_m = 12.0)$ [1] |
| $\tilde{\mathbf{x}}_m$ | $\mathcal{N}(\tilde{\mu}\tilde{\mathbf{x}}_{\text{init}}, \tilde{\Sigma}^2 = 10.0)$ |
| Variational distributions: | |
| $q\mathbf{w}_m$ | $\mathcal{N}(\mu_m^{(q)} = 0.0, \sigma_m^{2(q)} = 120)$ |
| $q\bar{l}_m$ | $\mathcal{LN}(\alpha_m^{(q)} = 1.0, \beta_m^{(q)} = -3)$ [2] |
| $q\tilde{\mathbf{x}}_m$ | $\mathcal{N}(\tilde{\mu}^{(q)} = \tilde{\mathbf{x}}_{\text{init}}, \tilde{\sigma}^{2(q)} = 0.001)$ |

Table 1 Using a clipped part of video 1, how kernels are moved with the number of iterations in the optimization is shown in the attached video (after 55 seconds). The final lengthscales along with the predicted occupancy for datasets 1, 2, 3, and 4 are shown in Figure 2.

[1]The gamma distribution is defined as $\mathcal{G}(x;\alpha,\beta) := \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$ where $\Gamma(\alpha) := \int_0^\infty z^{\alpha-1} e^{-z} \mathrm{d}z$ is the gamma function. $\alpha > 0$ shape parameter and $\beta > 0$ rate parameter. In literature, the scale parameter is sometimes defined as the inverse of the rate parameter instead of the rate parameter.

[2]The log-normal distribution $\mathcal{LN}$ is obtained by transforming a standard normal variable $z' = \exp(\mu + \sigma z)$

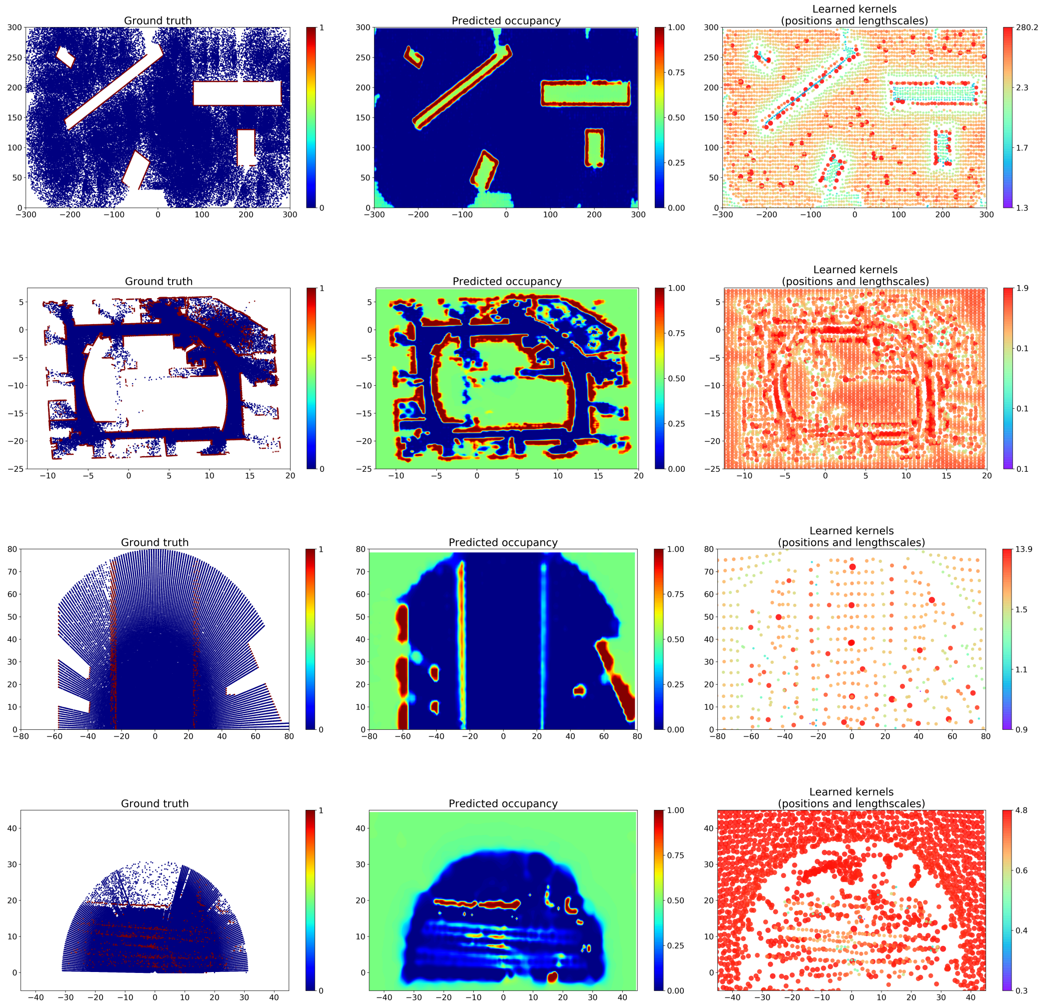

Figure 2: Left: laser scans, middle: predicted occupancy, right: learned kernel positions and lengthscales