Learning-aided 3D Occupancy Mapping with Bayesian Generalized Kernel Inference

Kevin Doherty, *Student Member, IEEE,* Tixiao Shan, *Student Member, IEEE,* Jinkun Wang, and Brendan Englot, *Member, IEEE*

***Abstract*—We consider the problem of building descriptive 3D maps from sparse and noisy range sensor data. We expand our previously proposed method leveraging Bayesian kernel inference for prediction of occupancy in locations not directly observed by a range sensor. In this work, we show that our kernel inference approach generalizes previous "counting sensor model" approaches from discrete occupancy grids to continuous maps. Our approach enables prediction about occupancy in regions unobserved by the range sensor based on local measurements, and smoothly transitions to a prior in regions lacking sufficient data for reliable inference. Furthermore, we demonstrate quantitatively using simulated data that the mapping performance of our method can be improved by considering rays as continuous observations, rather than sampling discrete free-space point observations along rays. Though the maps produced by our method are in principle continuous, discretizing space affords us several computational advantages, including the ability to apply recursive Bayesian updates, that allow us to perform inference very efficiently, even on large datasets. To demonstrate this advantage, we present experimental results applying this method to large-scale lidar data collected with a ground robot, showing real-time performance. Other field robotics applications, including underwater 3D mapping with sonar, are explored qualitatively.**

***Index Terms*—Mapping, Range Sensing, Learning and Adaptive Systems, Field Robots.**

I. Introduction

TRADITIONAL approaches to occupancy grid mapping [1], [2] discretize space into cells and estimate whether each individual cell is occupied, independent of all others. This poses a modeling challenge for sparse lidar measurements collected by ground or aerial vehicles, or noisy sonar measurements in the case of marine robots. In particular, sparsity and noise in range sensor data can leave gaps and inconsistencies in traditional occupancy grid maps, which can be misleading in robot planning and exploration scenarios. If a sensor ray has not directly passed through a grid cell, the occupancy state of that cell is assumed unknown. However, this assumption neglects the possibility that there may be ample local information to accurately infer the state of the cell in question. Thus, in order to build denser occupancy maps, we can make use of local context to predict occupancy in unobserved areas, rather than estimating occupancy for every location in space independently.

Our goal is to efficiently build dense, descriptive 3D occupancy maps given sparse, noisy range sensor data by inferring occupancy in unobserved regions of the map. Such maps will aid planning and navigation for a range of robotic systems, but in this work we focus on applications to ground vehicles equipped with lidar sensors and underwater vehicles equipped with sonar, in environments that are richly populated with structures of interest. Methods that currently address this goal are either inefficient computationally (as in the case of Gaussian process regression, with a $\mathcal{O}(N^3)$ time complexity in the number of range measurements [3]), or they infer point estimates of occupancy probabilities, as in the case of logistic regression [4], without estimating higher-order moments of the probability distribution over occupancy, which can be used to guide robot planning and exploration over incomplete maps.

To build more descriptive 3D occupancy maps, we propose a method leveraging Bayesian kernel inference. This concept first appeared in our recent prior work [5], where we showed that the proposed Bayesian kernel inference-based mapping approach accurately predicts occupancy maps in substantially less time than Gaussian process-based alternatives, while providing uncertainty estimates that can model situations without enough observed data to make a reliable occupancy prediction. Furthermore, it admits exact recursive updates to its predictions given new data, and reverts to a prior, which we may specify, whenever there is insufficient training data in the local vicinity of a query point to infer occupancy reliably. These properties make the proposed approach particularly well-suited for computationally-constrained mobile robot platforms, drones, and low-cost autonomous underwater vehicles, especially when the former are combined with high data-throughput sensors like lidars, and without network access to more powerful computational resources, since our method features a computational complexity that is *logarithmic* in the size of the training data.

In this paper, we expand our prior work [5] in several ways. First, we derive the Bayesian generalized kernel inference based mapping method as a continuous-space generalization of the classic "counting sensor model" [6]. This approach provides new insight into our method as a "kernel-smoothed"

Manuscript received August 1, 2018; accepted April 10, 2019. This paper was recommended for publication by Associate Editor N. Gans and Editor F. Chaumette upon evaluation of the reviewers' comments. This work was supported in part by the National Science Foundation, grant numbers IIS-1652064 and IIS-1723996.

K. Doherty is with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: kjd@csail.mit.edu).

T. Shan, J. Wang, and B. Englot are with the Department of Mechanical Engineering, Stevens Institute of Technology, Hoboken, NJ 07030 USA (e-mail: {tshan3, jwang92, benglot}@stevens.edu).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2019.2912487

sensor model, which we formalize in terms of a bound on Kullback-Liebler divergence. We also demonstrate quantitative advantages on simulated data by modeling sensor rays as continuous free-space observations, rather than sampling discrete free-space points along rays. To demonstrate the computational benefits obtained by our method while retaining accuracy, particularly when restricting the set of query locations to a grid, we substantially expand our experimental results from prior work, applying our method to real-time, large-scale mapping with an outdoor mobile ground robot, and to underwater mapping tasks performed by a sonar-equipped remotely-operated vehicle (ROV). Finally, we expand previous qualitative results in a long-term station-keeping example by providing a theoretical comparison of the limiting behavior of our approach in comparison to Gaussian process regression coupled with a Bayesian committee machine. In particular, we show formally that occupancy probabilities predicted by the latter method "saturate", i.e. tend toward 0 or 1 in the infinite limit, despite repeatedly observing the same scene, while our approach predicts a fixed, non-binary occupancy probability with lower variance in the limit.

II. Related Work

Historically, there have been many approaches seeking to relax the assumption of grid cell independence in traditional occupancy grid mapping. Thrun [7] proposed an expectation-maximization method where the predicted occupancy state in each cell is "flipped" until the map estimate most consistent with the observations is achieved. While avoiding the independence assumption, this method is intractable for large, finely-discretized grids.

The normal distributions transform (NDT) [8], originally proposed for scan-matching with range data, has been used for similar 3D occupancy mapping applications to ours [9]. NDT partitions the workspace into a grid of cells where each cell stores the parameters of a normal distribution representing the likelihood of a range measurement. Occupancy mapping with NDT makes use of recursive updates to the mean and covariance of these distributions.

More recently, learning-based methods making use of Gaussian process regression have been demonstrated in 2D [3] and scaled to 3D ([10], [11], [12]) by leveraging approximation techniques such as partitioning data across multiple, small-scale regressions, and fusing their products using the Bayesian committee machine [13]. These methods have shown great success in predicting occupancy within sparsely covered regions of a map, but in general they suffer from a time complexity that is $\mathcal{O}(N^3)$ in the number of range measurements, which may be prohibitive for real-time mapping and navigation tasks.

Several methods have been proposed that seek to achieve the map prediction accuracy of Gaussian processes with reduced computational complexity. To this end, Hilbert maps have been proposed [4]. Hilbert maps make use of a logistic regression classifier trained via stochastic gradient descent to enable faster training and inference. By making use of kernel feature approximations, Hilbert maps can perform inference in time that is linear in the number of measurements, multiplied by a constant coefficient that trades off the quality of the kernel approximation with the computational efficiency. Furthermore, through approximate Bayesian updates, real-time viable incremental 3D Hilbert mapping has been demonstrated [14].

Building on the Hilbert maps formulation, several methods have been proposed that seek to improve the underlying feature representation that is used in the logistic regression model. The use of automatic relevance determination to shape the kernel according to the local directions of maximum variance in range measurements has shown promising results for mapping tasks in 2D and 3D [15]. Recently these methods have been augmented through the use of features derived from the latent representation learned by a variational autoencoder trained offline on structured range data [16]. While these methods improved the computational efficiency of the Hilbert map method, the logistic regression classifier used by all of the Hilbert mapping approaches does not provide associated uncertainties in occupancy probability estimates, which can be used to guide robot planning and exploration [17].

Finally, "confidence-rich" grid maps [18] extend traditional occupancy grid mapping to consider dependencies between observations within the measurement cone. The technique also augments standard occupancy grid maps with confidence values, similar to our proposed work and to Gaussian process occupancy mapping, and its utility in support of planning has been demonstrated in 2D mapping scenarios [19]. While this method models correlations between observations within the measurement cone of a sensor, we focus on dependencies that exist between observations which may not be in the same measurement cone, and may have been measured at different points in time.

Of these methods, only Gaussian process regression and Bayesian generalized kernel inference (the proposed framework) have been shown to be capable of estimating a full posterior distribution over occupancy probability at all points in 3D space. Consequently, these methods will form our basis for comparison and evaluation in the work that follows.

III. Background - The Counting Sensor Model

Related to our approach is the *counting sensor model* for occupancy grid mapping, proposed in [6]. Assuming map cells are indexed by $j \in \mathbb{Z}^+$, the counting sensor model for the j-th map cell with occupancy probability θ_j has the Bernoulli likelihood:

$$p(y_i \mid \theta_j) = \theta_j^{y_i}(1-\theta_j)^{1-y_i}, \tag{1}$$

where we define the measurements $\mathcal{Y} = \{y_1, \ldots, y_N \mid y_i \in \{0, 1\}\}$ indicating whether a beam was reflected by or passed through the map cell containing the corresponding position in $\mathcal{X} = \{x_1, \ldots, x_N \mid x_i \in \mathbb{R}^3\}$, so that the i-th range measurement gives the pair $(x_i,\ y_i)$. A single beam from a range sensor may contain many measurement pairs, where 3D positions x_i corresponding to free space (with $y_i = 0$) are sampled along the beam and those corresponding to occupied space (i.e. those with $y_i = 1$) are located at the end-points. Here we take i as the index of each of these measurements, and there are N such measurements acquired by the vehicle.

Algorithm 1 Counting Sensor Model

Input: Training data: $\mathcal{X}, \mathcal{Y}$; Query cell: j
Initialize: $\alpha_j \leftarrow \alpha_0, \beta_j \leftarrow \beta_0$
for each $(x_i, y_i) \in (\mathcal{X}, \mathcal{Y})$ **do**
  if x_i in cell j **then**
    $\alpha_j \leftarrow \alpha_j + y_i$
    $\beta_j \leftarrow \beta_j + (1 - y_i)$
  end if
end for
return $\alpha_j,\ \beta_j$

In mapping, we are concerned with the posterior over possible θ_j; $p(\theta_j \mid \mathcal{X}, \mathcal{Y})$. Adopting a conjugate model, we have the prior over θ_j given by $\mathrm{Beta}(\alpha_0, \beta_0)$, where $\alpha_0,\ \beta_0 \in \mathbb{R}_{>0}$ are prior hyperparameters, usually set as $\alpha_0 = \beta_0 \approx 0$ to place a small, uninformative prior on occupancy probability. Applying Bayes' rule, we find that the posterior is given by $\mathrm{Beta}(\alpha_j, \beta_j)$, where α_j and β_j are defined as follows:

$$\alpha_j := \alpha_0 + \sum_{i,\ x_i \text{ in cell } j} y_i \tag{2}$$

$$\beta_j := \beta_0 + \sum_{i,\ x_i \text{ in cell } j} (1 - y_i). \tag{3}$$

That is, α_j maintains a count of instances where a beam is reflected in the j-th grid cell, while β_j is a count of instances where a beam passed through the cell, giving the model its name. The maximum *a posteriori* (MAP) estimate of θ_j then has the closed-form solution

$$\hat{\theta}_j = \frac{\alpha_j - 1}{\alpha_j + \beta_j - 2},\ \alpha_j, \beta_j > 1, \tag{4}$$

and we can also compute the expected value and variance of θ_j as follows:

$$\mathbb{E}[\theta_j] = \frac{\alpha_j}{\alpha_j + \beta_j} \tag{5}$$

$$\mathbb{V}[\theta_j] = \frac{\alpha_j \beta_j}{(\alpha_j + \beta_j)^2(\alpha_j + \beta_j + 1)}, \tag{6}$$

both of which will be useful during the mapping process. The inference process using the counting sensor model is summarized in Algorithm 1. As above, we assume grid cells are indexed by positive, nonzero integers j. Given observations $\mathcal{X},\ \mathcal{Y}$, the goal is to compute the parameters of the posterior Beta distribution for a query cell j, namely α_j and β_j. Given the inferred values of these parameters, any of the desired properties of the distribution over occupancy in the cell j can be computed, such as the mode (4), mean (5), or variance (6).

The counting sensor model has been used primarily in the context of discrete grid maps. In the following section, we will show that our method generalizes the counting model to continuous maps, and does so by considering the influence of local observations in a way that mitigates the issues otherwise encountered as a consequence of the independence assumption in a traditional realization of occupancy grid mapping.

IV. Occupancy Mapping with Bayesian Generalized Kernel Inference

To introduce correlation spatially between occupancy predictions at nearby points in the environment, we consider adapting the counting sensor model to continuous space. Such *continuous* occupancy maps have become desirable recently for their ability to reduce memory requirements while allowing queries at arbitrary resolution [3], [4]. Absent a grid, we lose many of the practical benefits of the counting model for mapping, as it is highly unlikely that any particular location in a continuous three-dimensional space will be observed more than once. To adapt this method to the continuous domain, we need to establish some neighborhood of a query point for which sensor observations will be considered relevant.

We first formulate this problem as one of inferring a smooth distribution over occupancy. The model considers all points that have been obtained, and in order to reason only about points in the local neighborhood of a query point, and therefore avoid the computational burden associated with the consideration of every point, we use a sparse kernel [20]. Finally, we consider a free space representation where sensor rays are treated as continuous observations as an alternative to our previous approach, which samples free space points along sensor rays. In our derivation, we will assume the map is static. We use test-data octrees [12] as our primary spatial representation, allowing us to merge neighboring cells with the same state, the results of which can be visualized most easily in Figure 9.

A. Bayesian Generalized Kernel Inference

We may extend the discrete counting sensor model to the case of continuous occupancy maps by constraining the inference problem to distributions over occupancy that are *smooth*. Here we define a smooth distribution as having bounded Kullback-Leibler divergence between the "extended likelihood" $p(y_i|\theta_*, x_i, x_*)$ and the likelihood $p(y_i|\theta_i)$. Here θ_* is the value of the latent variable, namely the occupancy probability, at the location of the query point x_*. In [21], it is shown that the maximum entropy distribution g, which, for some distribution f satisfies $D_{KL}(g\|f) \leq \rho(x_*, x)$ is of the form $g(y) \propto f(y)^{k(x_*,x)}$ where, $D_{KL}(\cdot\|\cdot)$ is the Kullback-Leibler divergence, $\rho : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^+$ bounds the information divergence from g to f, and $k : \mathbb{R}^3 \times \mathbb{R}^3 \to [0, 1]$ is a kernel function operating on our three-dimensional spatial inputs. The kernel k and smoothness bound ρ can be uniquely determined from one another, and a kernel function is easier to specify than a bound on information divergence. If g is taken to be the extended likelihood, and f is the likelihood, then the posterior distribution of a model with input-output pairs $\mathcal{D} = \{(x_i, y_i)\}$ is given by

$$p(\theta \mid x, \mathcal{D}) \propto \left[\prod_{i=1}^{N} p(y_i \mid \theta)^{k(x,x_i)}\right] p(\theta \mid x). \tag{7}$$

Supposing we adopt the same Bernoulli observation model described in Equation (1), then we can again place a prior

distribution $\mathrm{Beta}(\alpha_0, \beta_0)$ over θ and obtain the posterior

$$p(\theta \mid x, \mathcal{D}) \propto \theta^{\alpha_0 - 1}(1-\theta)^{\beta_0 - 1} \prod_{i=1}^{N} \theta^{k(x,x_i)y_i}(1-\theta)^{k(x,x_i)(1-y_i)}, \quad (8)$$

which is proportional to $\mathrm{Beta}(\alpha, \beta)$ with α and β defined as

$$\alpha := \alpha_0 + \sum_{i=1}^{N} k(x, x_i) y_i \quad (9)$$

$$\beta := \beta_0 + \sum_{i=1}^{N} k(x, x_i)(1 - y_i). \quad (10)$$

The similarities between this pair of equations and Equations (2) and (3) for the grid-based counting model are apparent: both models consider the sum of observations of each type. Distinctly, now observations are weighed by their distance to x, the point we are considering, according to the kernel function. Furthermore, here we are considering *all* of the observations, not just those in a particular cell. Using these definitions of α and β, we can compute the mode, mean, and variance for the continuous model exactly as given in Equations (4)-(6). In areas with little or no training data, this method will transition to the prior given by α_0 and β_0. The kernel in this inference model need not be positive-definite, nor symmetric. Consequently, we observe that this method generalizes the counting sensor model. By choosing a kernel such that if the query point x is in cell j, then $\forall_{i,\ x_i \text{in cell } j}\ k(x, x_i) = 1$ and $\forall_{i,\ x_i \neg \text{in cell } j}\ k(x, x_i) = 0$, we then recover the original counting sensor model.

One advantage to this approach comes when we choose to restrict our choice of query points during mapping to a grid. If we query the same locations in space repeatedly, we can integrate new data recursively by adding the kernel-weighted contribution of the novel observations to the parameters α and β as follows:

$$\alpha_t := \alpha_{t-1} + \sum_{i=1}^{N_t} k(x, x_i) y_i \quad (11)$$

$$\beta_t := \beta_{t-1} + \sum_{i=1}^{N_t} k(x, x_i)(1 - y_i), \quad (12)$$

for a set of N_t observations. The ability to perform recursive updates in this fashion also serves to decrease memory usage, as we can maintain a grid or octree, rather than storing large point clouds containing an entire history of observations.

B. Sparse Kernel

One issue with the updates as they were presented in Equations (9) and (10) is that they require the consideration of every observation we have made to perform inference at any given location. We would like to enforce some notion of locality on the inference model to satisfy the intuition that points which are far away in space are unlikely to be related at all. To do so, we choose a kernel function giving a value of 0 for points whose distance is greater than some fixed threshold

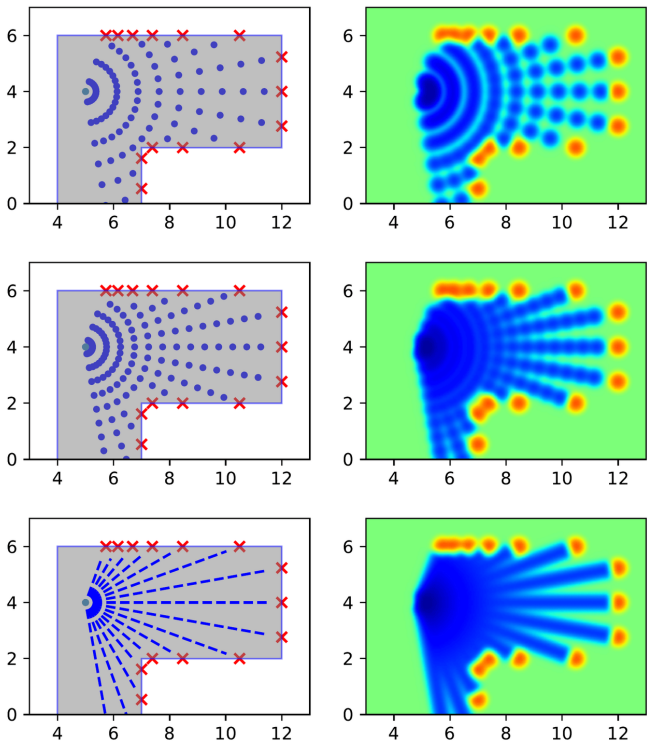


Fig. 1: An illustrative comparison of free-space representations. From top to bottom, we show the training data (left) and BGK inference result (right) due to: (1) coarse linear interpolation of free space, (2) finer linear interpolation of free space, and (3) point-to-line distance kernel evaluation.

$l > 0$. The kernel presented in [20] satisfies this property, and is defined as follows:

$$k(x, x') = \begin{cases} \sigma_0 \left[\frac{2 + \cos(2\pi \frac{d}{l})}{3} (1 - \frac{d}{l}) + \frac{1}{2\pi} \sin(2\pi \frac{d}{l}) \right] & \text{if } d < l \\ 0 & \text{if } d \geq l \end{cases} \quad (13)$$

where $d = \|x - x'\|$ and σ_0 is a constant hyperparameter. Choosing this sparse kernel allows us to exactly recover the occupancy probability at a given location by retrieving only the observations within a distance of l from the query point. We can efficiently and exactly evaluate the kernel over the relevant observations in $\mathcal{O}(\log N)$ time using a k-d tree. Simply by querying a k-d tree containing the training points for a scan with radius l about each query point x, we obtain all training points with nonzero contribution to the kernel computation in Equation (13). The overall computational complexity of the inference method is $\mathcal{O}(M \log N)$, where M is the number of test points and N is the number of training points.

C. Free Space Representation

For a range sensor, observations representing the "occupied" class are easily obtained, as they are the endpoints of a sensor ray. On the other hand, there have been several free-space representations used in this context: the nearest point on each ray to a query point [3], linearly or randomly interpolated

Algorithm 2 Bayesian Generalized Kernel (BGK) Inference

Input: Training data: $\mathcal{X}, \mathcal{Y}$; Query point: x_*
Initialize: $\alpha_*, \beta_* \leftarrow 0$
for each $(x_i, y_i) \in (\mathcal{X}, \mathcal{Y})$ **do**
  if `use_line` **then**
    Compute d from Eq. (14), (15)
  else
    $d \leftarrow \|x_i - x_*\|$
  end if
  $k_i \leftarrow k(x_i, x_*)$ Sparse kernel, Eq. (13)
  $\alpha_* \leftarrow \alpha_* + k_i y_i$
  $\beta_* \leftarrow \beta_* + k_i(1 - y_i)$
end for
return $\alpha_*,\ \beta_*$

points on each ray [4], [12], [14], and free space points weighed by the length of a range beam [22].

In our previous work [5], the algorithm we termed BGKOctoMap made use of free space points linearly interpolated along each range beam. This free space representation is potentially problematic for use with the Bayesian generalized kernel inference model. In the choice of sampling resolution, we face two potential issues: if the sampling resolution is too low (i.e. we take relatively few samples), our coverage of free space will be poor, and there may even be points which lie along the range beam that are not classified as free according to the model; if the sampling resolution is too high, we may bias the model toward predicting free space, since we are artificially increasing the amount of free space observations, and at its core, our method is counting observations.

Here we propose instead to consider each range beam as a single observation of free space, and use the point-to-line distance when evaluating the kernel. The differences resulting from this assumption, compared with using linearly interpolated free space points, are illustrated in Figure 1. The quality of the sample-based free-space representation is greatly impacted by the choice of sampling resolution. This approach is equivalent to using only the point on each range beam closest to the query point, as proposed in [3]:

$$x_{free} = \begin{cases} P, & \text{if } d < 0 \\ P + d\frac{PQ}{|PQ|}, & \text{if } 0 \leq d \leq 1 \\ Q, & \text{if } d > 1 \end{cases} \tag{14}$$

$$d = \frac{PQ \cdot Px_*}{|PQ|}, \tag{15}$$

in which PQ is the ray between the sensor origin and the terminal hit point of the range beam and Px_* is the ray between the sensor origin and the query point x_*.

In practice, evaluating this expression for every range beam during every query is undesirable, particularly since the resulting kernel computation for many of these points will be zero. Instead, we take free-space samples along the range beams as previously, but at query time, we retrieve these samples and map them back to the set of unique range beams that produced them. With this method, as long as our sampling resolution and radius for obtaining relevant observations (for example, in a query to a k-d tree) are sufficiently large relative to the kernel length scale, we can recover all of the lines with nonzero contribution from the kernel. The predictions are invariant to free space sampling resolution above the minimum necessary resolution to recover all of the range beams within a distance l of a query point. In addition, this line-based approach, which we term BGKOctoMap-L, retains the same $\mathcal{O}(M \log N)$ time complexity as our previous approach, with some additional memory expenditures to map all free-space points to their respective lines. In practice, we have found that BGKOctoMap-L allows us to sample rays at a lower resolution than when free-space samples are used directly for inference.

Parameter	Description	Value
l	Kernel Length	0.2 m
σ_0	Kernel Scale	0.1
α_0, β_0	Beta Prior Parameters	0.001

TABLE I: Kernel parameters and Beta prior parameters used for experimental evaluation.

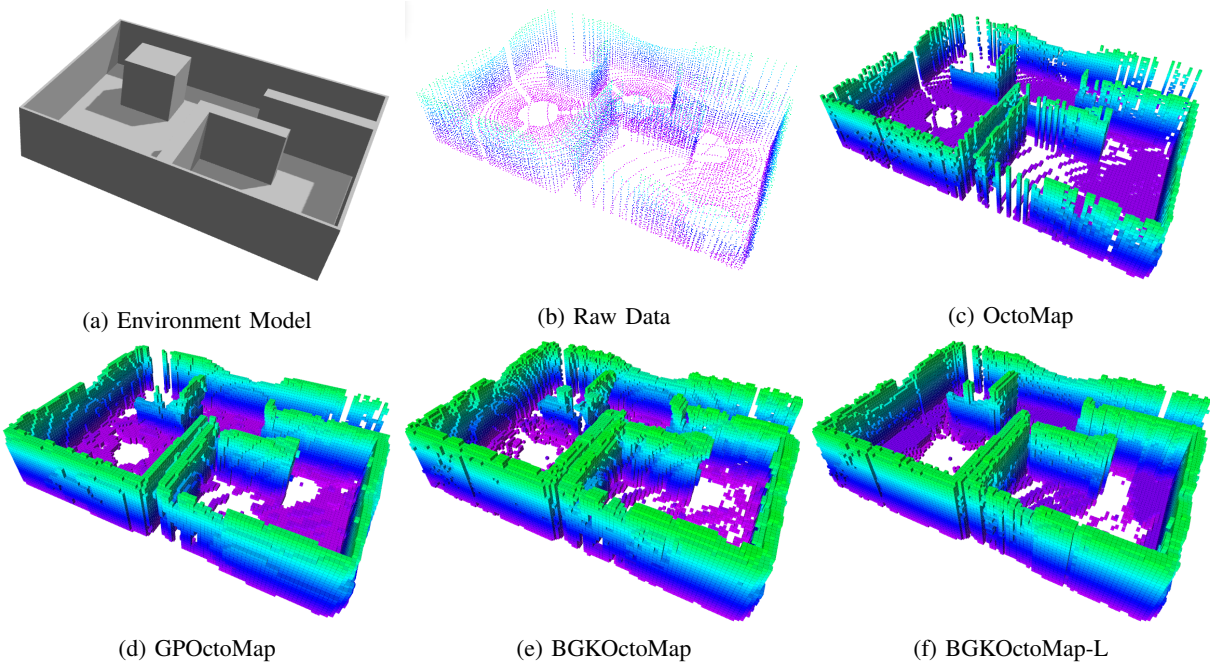
D. Algorithm Summary

Pseudocode for the Bayesian generalized kernel (BGK) inference procedure is provided in Algorithm 2. As in the inference procedure for the counting sensor model (Algorithm 1), the ultimate objective is to compute the values of the Beta posterior parameters given observations $\mathcal{X},\ \mathcal{Y}$. However, in the BGK inference procedure, these parameters correspond to a particular *location in 3D space*, x_*, rather than a map cell, and thus they are denoted α_* and β_*. We note the similarities between our method and the classic counting sensor model, and in particular we observe that for a query point x_*, we compute the parameters using *kernel-weighted* values y_i.

V. Experimental Results

We compared our proposed method (BGKOctoMap-L) to (1) our previous approach using free-space samples (BGKOctoMap) [5], (2) Gaussian process OctoMaps (GPOctoMap) [12], and (3) the standard OctoMap [23]. The comparison was performed over two simulated datasets with known ground truth, made in Gazebo [24]; a "structured" environment with mostly rectangular features and an "unstructured" environment with primarily rounded obstacles reminiscent of a forest, and real data both from the University of Freiburg [25], and collected with our own ground robot on the Stevens campus. The methods presented, with the exception of OctoMap, all use our own C++ implementation[1], which makes use of the Robot Operating System (ROS) [26] and the Point Cloud Library (PCL) [27]. For BGKOctoMap-L, we use the kernel parameters $l = 0.2$, $\sigma_0 = 0.1$, and sample free space linearly along rays at 1 meter resolution. We use the hyperparameters $\alpha_0 = \beta_0 = 0.001$ to provide an uninformative prior. All methods predict at a maximum resolution of 10 cm. For all other methods requiring free-space points, linearly-spaced

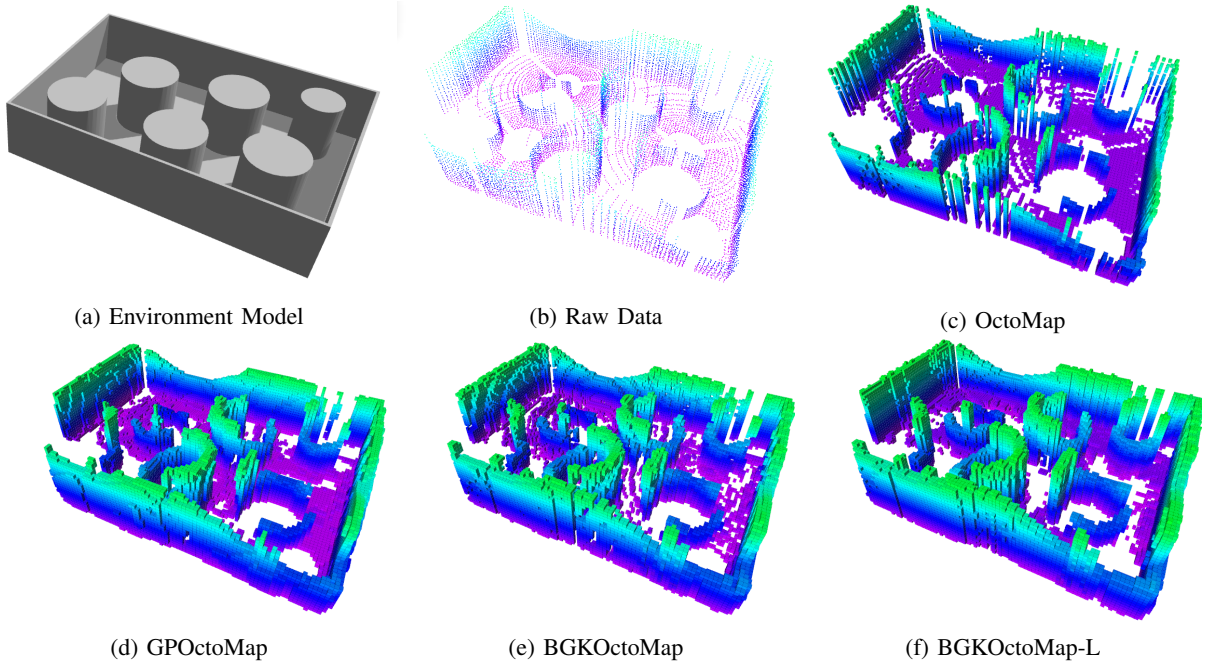
[1]The code for this paper is implemented in Learning-aided 3D Mapping Library, available at `https://github.com/RobustFieldAutonomyLab/la3dm`.



(a) Environment Model (b) Raw Data (c) OctoMap

(d) GPOctoMap (e) BGKOctoMap (f) BGKOctoMap-L

Fig. 2: Mapping results on a simulated structured-environment dataset.



(a) Environment Model (b) Raw Data (c) OctoMap

(d) GPOctoMap (e) BGKOctoMap (f) BGKOctoMap-L

Fig. 3: Mapping results on a simulated unstructured-environment dataset.

samples were taken along sensor rays at a resolution of 0.5 m. Parameter values are summarized for reference in Table I. All experiments were performed on a laptop with an 8-core 2.60 GHz Intel i7 CPU running Ubuntu Linux.

A. Simulated Data

Our structured and unstructured simulated datasets have dimensions $10.0 \times 7.0 \times 2.0$ meters. In Figure 2, we show the results of the four mapping algorithms we consider in the structured environment. In these figures, as well as in subsequent visualizations, only the cells determined to be occupied are shown. We find that with the exception of OctoMap, all of the methods are able to fill in gaps in the walls where sensor coverage is poor. While BGKOctoMap is able to complete the walls in many cases, there are also many artifacts on the walls due to the bias toward predicting free space caused by oversampling free space points along rays. This is amended in the result from BGKOctoMap-L. Visually, we find that GPOctoMap and BGKOctoMap-L offer the most intuitively satisfying results, given what we know about the environment. In Figure 3, we find that GPOctoMap provides the best results. This may be due to the fact that there are many occluded regions of the unstructured map, and GPOctoMap is

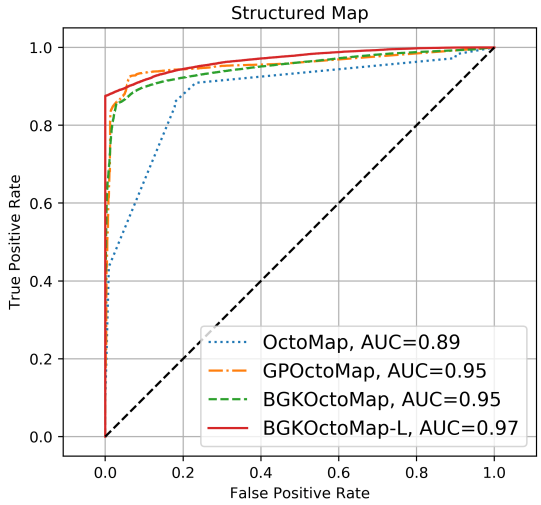


Fig. 4: Receiver operating characteristic of the four competing methods evaluated on the structured map dataset.

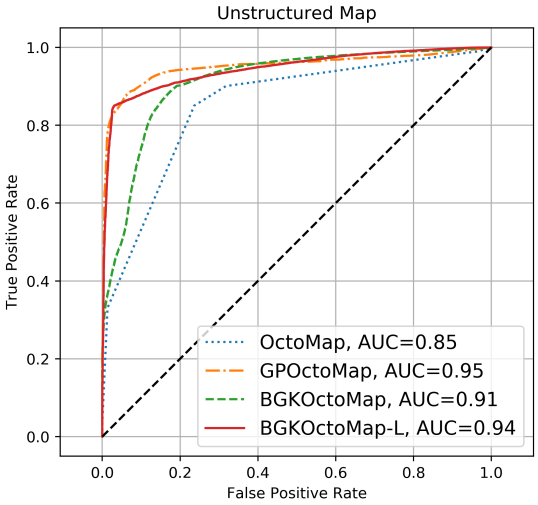


Fig. 6: Receiver operating characteristic of the four competing methods evaluated on the unstructured map dataset.

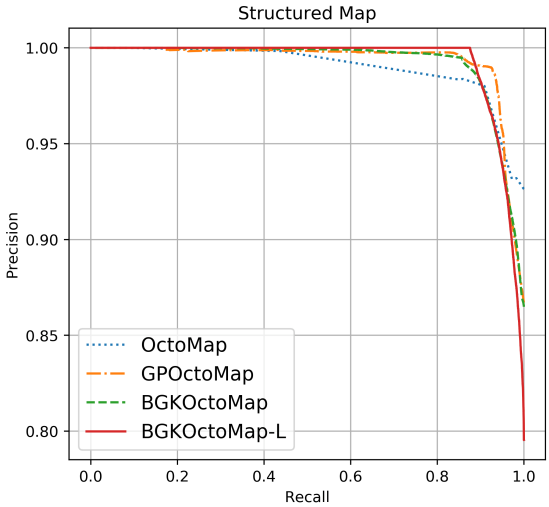


Fig. 5: Precision-recall curve for the four competing methods evaluated on the structured map dataset.

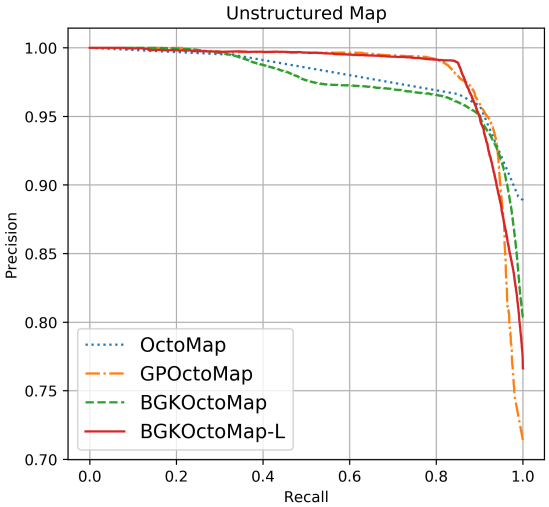
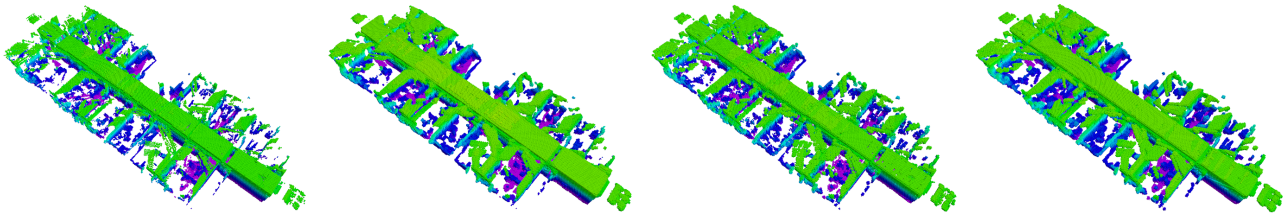


Fig. 7: Precision-recall curve for the four competing methods evaluated on the unstructured map dataset.

better-suited to the task of extrapolating larger-scale trends in spatial structure.

In Figures 4 and 6 we provide the receiver operating characteristic (ROC) curves for each method applied to the structured and unstructured simulation datasets, respectively. The ROC curve is produced by varying the threshold on occupancy probability at which we deem a location occupied. For the purposes of our evaluation, we consider the positive class to be occupied and the negative class to be free space. A range of these values is evaluated, and the true positive rate and false positive rate are computed. In general, as we lower the threshold on occupancy probability, we will increase the true positive rate as well as the false positive rate. The ROC curve shows how these rates vary with one another as the occupancy probability threshold is changed, and it is desirable for a method to achieve a high true positive rate when there is a very low false positive rate (i.e. a curve tending toward the upper-left of the plot is best). We also provide the area under the curve (AUC), which reflects the mapping performance of each method. We find that the BGKOctoMap-L method, with its continuous free-space representation, slightly outperforms BGKOctoMap and GPOctoMap on the structured environment. In contrast, the harder unstructured environment is most accurately mapped by GPOctoMap, though BGKOctoMap-L slightly outperforms standard BGKOctoMap.

In addition the ROC curves, we provide precision-recall curves for the structured and unstructured maps respectively in Figures 5 and 7. Precision is defined as the ratio of true positives to actual positives (that is, the total number of true positives plus false positives), and recall is defined as the ratio of true positives to the total number of predicted positives (i.e. true positives plus false negatives). Along the curve we vary the threshold on occupancy probability determining whether we predict the positive class (occupied) or the negative class (free) and plot the corresponding values of precision and recall. In general, better performance is indicated by higher recall and higher precision, a curve tending toward the upper-right of the plot. Here we observe that all of the methods perform



(a) OctoMap (b) GPOctoMap (c) BGKOctoMap (d) BGKOctoMap-L

Fig. 8: Maps built using the Freiburg corridor dataset. All methods produce qualitatively similar maps in the case of relatively dense data.

similarly on the structured dataset, while BGKOctoMap-L and GPOctoMap slightly outperform other methods on the unstructured dataset. Most notably, we find that here is an area where the *line-based* free-space representation presents quantitative advantages. Examining the ROC and precision-recall curves for the unstructured map (Figures 6,7) we find that augmenting the BGKOctoMap method with the line-based representation offers performance much more comparable to GPOctoMap. Qualitatively, we observe in Figure 3 some artifacts in the map produced by BGKOctoMap due to the effects of free-space sampling (in particular curved features on the flat surfaces where sensor measurements were made). These effects are substantially reduced by considering the sensor ray as a continuous free-space measurement with BGKOctoMap-L.

In terms of computation time, shown in Table II, we find that at the scale of the simulated data, there is little difference between the competing approaches. Most notably, OctoMap has the slowest run time by a small margin. This is due to the fact that the three other methods benefit from parallelization, while OctoMap does not. The remaining methods, however, make use of parallelization for the same set of operations, so comparison between these methods is more appropriate.

B. Lidar Mapping Experiments

We qualitatively evaluated each method on the Freiburg corridor dataset [25], as well as a large dataset we collected at the Stevens campus using a Clearpath Jackal unmanned ground vehicle (UGV) equipped with a Velodyne VLP-16 lidar. To support online planning and decision-making by UGVs over the maps constructed, in any desired location, an embeddable, real-time viable mapping capability is essential.

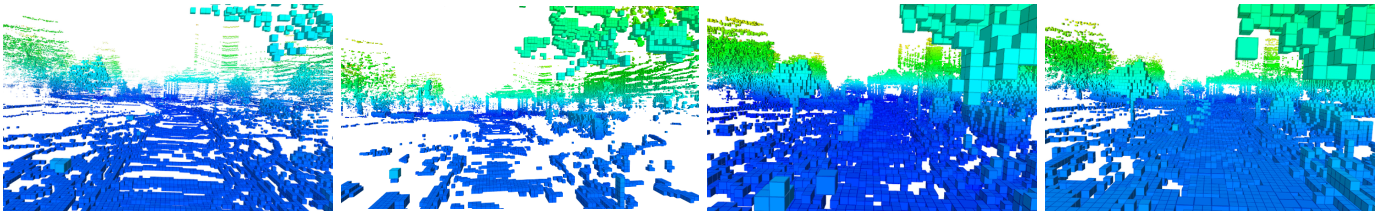
The Freiburg corridor dataset is substantially larger than the simulated datasets, both physically, spanning $43.8 \times 18.2 \times 3.3$ meters, and in the amount of data collected by the range sensor. Consequently, it also provides a more realistic scenario for comparison of these methods on real range data. In Figure 8, we show the maps produced using the corridor dataset. On real data of this scale, we observe that all of the methods outperform OctoMap on the task of producing a dense map from range data, though all of the alternatives we tested give qualitatively similar results. The processing time required for each method is presented in Table II; here, we find that time performance of GPOctoMap begins to diverge.

The Stevens campus dataset was collected over about 40 minutes, with a total distance traveled of 2.9 km. Lidar data was collected while teleoperating the Jackal UGV. Incoming point clouds were registered using lidar odometry and mapping (LOAM) [28]. Scans are registered with LOAM at a rate of 1 Hz, then the registered scans are integrated into the occupancy map using an effective maximum range of 50 m. Though methods like those in [29] exist, which allow consideration of pose uncertainty during mapping, we found that LOAM provided accurate pose estimates with very low drift and gave high-quality scan registrations that were useful for mapping even under the assumption of perfect knowledge of pose.

As with the other datasets, Table II shows the average time per scan and the total time spent processing point clouds for each method. The reported time per scan includes the entire processing time for a given lidar scan. This includes storing hit points and free-space sample points or lines (the "training data"), building a k-d tree from the data, and querying the tree to update each grid-cell in the neighborhood of the training data. Since some point clouds fall outside the effective maximum range, timing results are provided for scans which are within sensor range (since data falling outside this range is not processed). GPOctoMap and OctoMap were not able to process every scan in real-time without dropping messages. For these methods, average times were computed on the scans which were processed, and total times are extrapolated estimates based on these expected processing times. Both of the methods using Bayesian generalized kernel inference, on the other hand, processed all scans falling within sensor range in real-time. A qualitative comparison of each method from a ground-level view is provided in Figure 9, and from an aerial view in Figure 10. This dataset is also visualized in full in this paper's video attachment, for both a 50m and 8m maximum sensing range[2]. We note that while the primary computational experiments were performed on laptop with a 2.6 GHz i7 CPU, the visualization for the video was produced using a desktop computer equipped with a substantially more powerful 10-core 3.0 GHz i7. Empirical time differences between the methods persisted even with this change in computation capabilities.

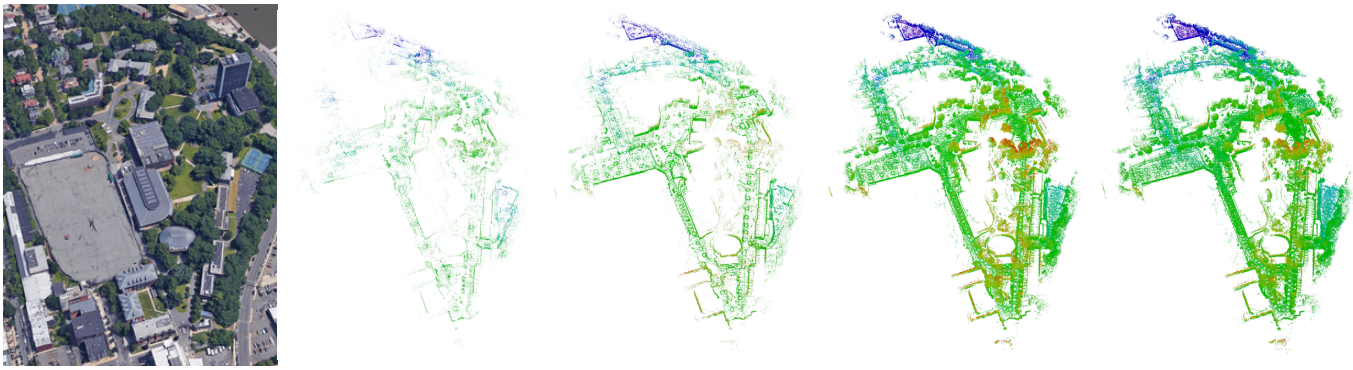
From Figure 9, we observe that BGKOctoMap and BGKOctoMap-L are able to successfully fill in gaps in lidar data in the floor, trees, and buildings in the distance. Since we make the static map assumption, we also obtain ghost artifacts of the operator walking along with the robot. On this data, GPOctoMap performs fairly poorly, relative to the alternatives. This is counterintuitive, but likely owes to a few factors: it

[2] https://youtu.be/SRXLMALpU20



(a) OctoMap (b) GPOctoMap (c) BGKOctoMap (d) BGKOctoMap-L

Fig. 9: Map comparison on data from the Stevens Institute of Technology campus, collected with a VLP-16 lidar-equipped Jackal UGV. Sparsity in the map built using GPOctoMap results from the constraint that the method runs in real-time and drops scans that are unable to be processed quickly enough. All maps have the same minimum cell size, and differences in cell sizes in the visualization are due to the cell merging procedure from our use of "test-data octrees" [12]. Maps are colored according to elevation.



(a) Satellite Image (b) OctoMap (c) GPOctoMap (d) BGKOctoMap (e) BGKOctoMap-L

Fig. 10: Aerial view of the Stevens campus maps, showing satellite imagery of the Stevens campus from Google Earth compared to the 3D maps produced by each method. Due to real-time constraints, the dropped scans of OctoMap and GPOctoMap result in much sparser maps than those produced by the proposed methods. Maps are colored according to elevation.

Dataset	Dimensions (m)	Scans	Pts./Scan	Sampled Pts./Scan	Method	Avg. Time/Scan (s)	Time (s)
Structured Simulation	$10.0 \times 7.0 \times 2.0$	12	3500	1506	**BGKOctoMap-L**	**0.013**	**0.15**
					BGKOctoMap	0.013	0.16
					GPOctoMap	0.018	0.21
					OctoMap	0.021	0.25
Unstructured Simulation	$10.0 \times 7.0 \times 2.0$	12	3500	1506	**BGKOctoMap-L**	**0.013**	**0.15**
					BGKOctoMap	**0.013**	**0.15**
					GPOctoMap	0.013	0.16
					OctoMap	0.014	0.17
Freiburg Corridor FR-079	$43.8 \times 18.2 \times 3.3$	66	89445	7601	**BGKOctoMap-L**	**0.32**	**3.8**
					BGKOctoMap	0.38	4.6
					GPOctoMap	0.46	5.6
					OctoMap	0.73	8.8
Stevens Campus	$622.4 \times 344.8 \times 43.4$	2968	26237	12237	**BGKOctoMap-L**	**0.53**	**1587.9**
					BGKOctoMap	0.67	1988.6
					GPOctoMap	3.3	9794.4
					OctoMap	14.6	43332.8

TABLE II: Computation times for the four environments used in mapping experiments with real and simulated lidar data.

must drop a large number of scans during processing, and in the absence of sufficient "hit" points, GPOctoMap may more readily infer free space.

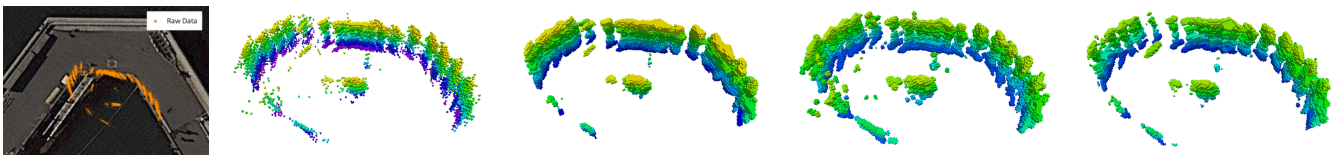
C. Underwater Sonar Mapping Experiments

We also evaluated the proposed methods on two underwater sonar datasets collected by a VideoRay ROV using a Tritech Micron single-beam scanning sonar. The sonar data was filtered to extract the point along each beam with the maximum intensity, representing the range to the nearest obstacle. The range data was clustered to produce landmark observations, and the landmark locations and robot poses were estimated jointly using the simultaneous localization and mapping approach of Wang et al. [30]. Given the estimated poses and range information, we applied each of the occupancy mapping methods under consideration to compare performance on sparse and noisy range data.

The first dataset, which features a series of submerged cylindrical pier pilings, was collected at Hudson River Park, Pier 84, in Manhattan, New York, USA. The second dataset, which features three angled sections of a corrugated seawall, was collected at the United States Merchant Marine Academy in Kings Point, New York, USA. Results from all four mapping approaches tested are shown in Figures 11 and 12 respectively. It is evident that both GPOctoMap and BGKOctoMap-L, in addition to providing descriptive 3D maps, aid in denoising the sonar data as well. Because low-cost underwater robots are

(a) Satellite Image (b) OctoMap (c) GPOctoMap (d) BGKOctoMap (e) BGKOctoMap-L

Fig. 11: Results of the four competing mapping methods applied to the Hudson River Park, Pier 84 dataset.



(a) Satellite Image (b) OctoMap (c) GPOctoMap (d) BGKOctoMap (e) BGKOctoMap-L

Fig. 12: Results of the four competing mapping methods applied to the US Merchant Marine Academy dataset.

often equipped with severely limited embedded computing resources, and online mapping is essential for obstacle avoidance if such a robot is operating autonomously, BGKOctoMap-L presents relevant advantages in its scalability.

D. Station-Keeping Experiment

Here we demonstrate the stable performance of the proposed mapping algorithm, one of its most desirable and useful features. We provide a station-keeping scenario in which a simulated robot repeatedly scans a single location. Using the structured environment simulation, we repeatedly input the same point cloud to both BGKOctoMap and GPOctoMap. The effects of this demonstration are provided in Figure 13, where we show the output of each method after 1, 15, 30, and 60 scans. We observe that while our method does experience some slight change due to the contribution from the new data (particularly in areas where $\alpha_0+\beta_0 \approx \sum_{i=1}^{N} k(x, x_i)$ after one scan), the change is mild in comparison to that of GPOctoMap. Repeated application of the BCM update approximations cause GPOctoMap to gradually predict that the walls and floor of the map are thicker, even though we update it repeatedly with the same point cloud in each iteration.

In the case demonstrated in Figure 13, we can analytically determine the probabilities predicted by a Gaussian process regression model updated using the BCM in the limit as the number of scans, K, goes to infinity:

$$\lim_{K\to\infty} p(y_* = 1 \mid \mathcal{D}_{1:K}) = \begin{cases} 1, & \text{if } \mu_*^{(1)} > 0 \\ 0.5, & \text{if } \mu_*^{(1)} = 0 \\ 0, & \text{if } \mu_*^{(1)} < 0, \end{cases} \tag{16}$$

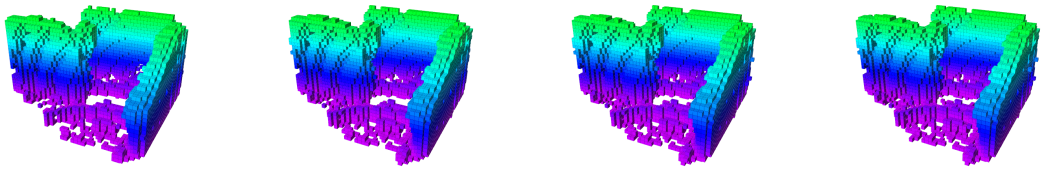
where $\mu_*^{(1)}$ is the mean value predicted by Gaussian process regression given the data from the first scan, $\mathcal{D}_i$, and training data is specified with $y = 1$ for "occupied" observations and $y = -1$ for "free" observations. That is, the predictions made by the Gaussian process in this formulation saturate, drifting toward 1 or 0 as more repeated observations are made. In contrast, taking the limit of the expected value of θ as the number of scans grow in the Bayesian kernel inference framework, we obtain that the predicted mean converges to a value that is approximately identical to the value predicted after the first scan, while the variance decreases. A detailed derivation of these results is provided in Appendix A.

We observe that by estimating the expected value of the parameter θ, we explicitly enable confident predictions that are neither 0 nor 1. If a substantial amount of conflicting data is collected, all in roughly the same proximity to a query point x_*, our estimate will be 0.5 with very low variance. That is, we know with relative certainty that the state of x_* is unknown.

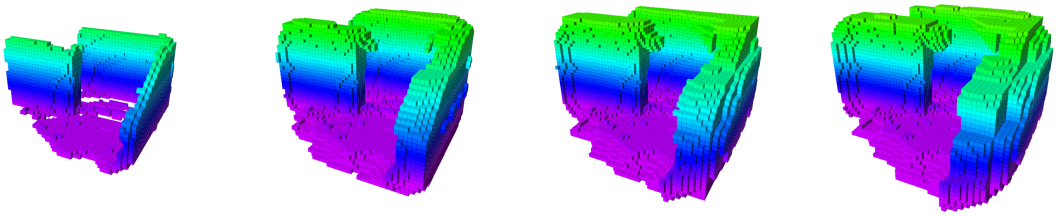
VI. Discussion

There are a number of areas where our method could be improved. Our principal assumption in our use of this method is that the likelihood of a given sensor observation satisfies a smoothness constraint in space. By choosing a kernel with a fixed length scale, we are applying the same smoothness constraint everywhere. Real environments, however, have structure in which the likelihood of a sensor observation need not vary smoothly in every direction, everywhere in the map. For example, we would expect the likelihood to be roughly constant for measurements along the tangent of a planar wall, but to change sharply along the normal to the wall. In [31], a mapping approach is proposed which varies the length scale of the kernel in each direction at different locations in space using the covariance of local observations, determined by clustering the data as a preprocessing step. Using local information not only to infer about occupancy, but to shape the kernel may improve these results further.

In addition, while it is computationally beneficial to rely only on local information, it is sometimes the case that structure in data exists at a larger scale than may effectively be captured by local kernel inference. Since we make use of only nearby training data at query time, the method is less adept at extrapolating large-scale trends in data. Nonetheless, we have observed that it is very capable of filling in many of the gaps encountered in lidar data. The only methods we found



(a) BGKOctoMap



(b) GPOctoMap

Fig. 13: In this simulated station-keeping demonstration, we show the results of updating both BGKOctoMap (Top) and GPOctoMap (Bottom) after the introduction of 1, 15, 30, and 60 scans containing the same data (Left to Right).

to operate in real-time in our campus mapping experiment were those that made use of Bayesian generalized kernel inference. For these reasons, we advocate for the Bayesian generalized kernel inference approach in scenarios where a vehicle may collect a large amount of sensor data which needs to be processed in real-time. In the small data regime, the $\mathcal{O}(N^3)$ complexity of Gaussian process-based methods may still allow real-time performance, and in these cases it is the preferred method to extrapolate larger-scale spatial trends. The latter case is demonstrated, for example, in the unstructured simulation experiments, where the limited amount of sensor data made extrapolation necessary to build an accurate map. On the other hand, the Bayesian generalized kernel inference method more conservatively reverts to an uncertain prior at distances far from the observed data.

VII. Conclusion and Future Work

In this work, we have shown that the Bayesian generalized kernel inference-based approach to mapping generalizes the counting model in [6] to continuous spaces. We have extended our mapping framework to use continuous range beams for free space representation, rather than samples along the beams, and show that this adaptation may improve mapping performance and computation time. Furthermore, we validated the method in a large-scale mapping experiment using a lidar-equipped Clearpath Jackal UGV, in which we demonstrated real-time performance, even for very large point clouds.

Several interesting avenues for future work in this area remain open. In particular, we do not address in this work the issue of *dynamic maps*. Relaxing the static map assumption to adapt these inference-aided mapping techniques to dynamic environments is critical for reliable performance in many practical scenarios. Additionally, this approach to mapping could be very useful for *exploration* of unknown environments. One key component of our method that can aid in both of these tasks is *the ability to represent uncertainty in occupancy probability.* This framework could flexibly accommodate temporal uncertainty in the case of a dynamic environment. In the context of exploration, the variance or entropy of the Beta posterior at different locations can guide a robot toward more uncertain regions of the map.

Appendix
Analysis of Station-Keeping Scenario

Gaussian process regression [32] is characterized by a predicted mean function and a predicted covariance function as follows:

$$\bar{f}_* = \kappa_*^T(\kappa + \sigma_n^2 I)^{-1}\boldsymbol{y} \tag{17}$$

$$\text{cov}(f_*) = \kappa_{**} - \kappa_*^T(\kappa + \sigma_n^2 I)^{-1}\kappa_*. \tag{18}$$

Here κ denotes the $N \times N$ matrix containing the results of the kernel function $k(\cdot)$ being evaluated for all pairs of input training data points X, κ_* is the $N \times M$ matrix formed by the kernel function outputs between each test data point in X_* and each training data point, and κ_{**} is the $M \times M$ matrix of kernel function outputs evaluated between all pairs of test data points. The vector $\boldsymbol{y}$ contains the labels $y_i \in \{-1, 1\}$ for the training data[3], and I is the $N \times N$ identity matrix. For this analysis we focus on a single scalar-valued output μ and its variance σ^2.

Since our goal is to predict occupancy probabilities, and the output of the Gaussian process regression is not constrained to the interval [0,1], the regression outputs are "squashed" using a logistic model

$$p(y_* = 1|X, \boldsymbol{y}) = \frac{1}{1 + \exp(-\gamma\omega_*)}, \tag{19}$$

where $\omega_i = \sigma_{min}^2\mu_i/\sigma_i^2$ is the weighted mean, σ_{min}^2 is the minimum variance, and γ is a positive constant parameter.

The Bayesian committee machine method for combining predictions of multiple Gaussian process regression models

[3]This is a convenient abuse of notation, but the training target values $y_i \in \{-1, 1\}$ for the Gaussian process regression model should not be confused with the target values in the Bayesian kernel inference model $y_i \in \{0, 1\}$).

partitions data into K subsets and performs the following updates to the mean and inverse variance for the case of one-dimensional target variables:

$$\mu_* = \sigma_*^2 \sum_{i=1}^{K} \sigma_*^{-2,(i)} \mu_*^{(i)} \tag{20}$$

$$\sigma_*^{-2} = -(K-1)\sigma_f^{-2} + \sum_{i=1}^{K} \sigma_*^{-2,(i)}, \tag{21}$$

where the superscript (i) denotes that the value was computed using the output of the Gaussian process model trained on dataset $\mathcal{D}_i$, and where σ_f^2 is a hyperparameter.

In this test scenario, the BCM performs poorly because datasets are assumed conditionally independent given the previously predicted target values, which causes the prediction variance to decrease, and then the result is "squashed" using the logistic function in Equation (19), causing the probabilities to saturate toward 0 or 1 with more data.

To see how this happens in practice, we can evaluate the BCM update to the mean in (20) and variance in (21) in this scenario. Formally, letting $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K \mid \forall_{i>1} \mathcal{D}_i = \mathcal{D}_1\}$, we can obtain the following expression for the inverse covariance:

$$\begin{aligned}
\sigma_*^{-2} &= -(K-1)\sigma_f^{-2} + \sum_{i=1}^{K} \sigma_*^{-2,(i)} \\
&= -(K-1)\sigma_f^{-2} + \sum_{i=1}^{K} \sigma_*^{-2,(1)} \\
&= -(K-1)\sigma_f^{-2} + K\sigma_*^{-2,(1)}
\end{aligned} \tag{22}$$

Here we observe that the inverse variance of K estimators trained on the same data is K times the inverse variance of one of the estimators. That is, each time we repeat this estimation procedure, the predicted variance decreases. We can observe a similar result for the Bayesian nonparametric kernel inference method we have presented. When we examine the predicted mean, we obtain the following:

$$\begin{aligned}
\mu_* &= \sigma_*^2 \sum_{i=1}^{K} \sigma_*^{-2,(i)} \mu_*^{(i)} \\
&= K\sigma_*^2 \sigma_*^{-2,(1)} \mu_*^{(1)}
\end{aligned} \tag{23}$$

Finally, we examine the result of "squashing" in (19) by dividing the expression for the mean in (23) by the variance term σ_*^2:

$$\frac{\mu_*}{\sigma_*^2} = \frac{K\sigma_*^2 \sigma_*^{-2,(1)} \mu_*^{(1)}}{\sigma_*^2} \tag{24}$$

$$= K\sigma_*^2 \sigma_*^{-2,(1)} \mu_*^{(1)}. \tag{25}$$

The output of the squashing logistic function then becomes:

$$\begin{aligned}
p(y_* = 1 \mid \mathcal{D}_{1:K}) &= \frac{1}{1 + \exp(-\gamma\omega_*)}, \ \gamma > 0 \\
&= \frac{1}{1 + \exp(-\gamma \frac{\sigma_{min}^2 \mu_*}{\sigma_*^2})} \\
&= \frac{1}{1 + \exp(-K\gamma\sigma_{min}^2 \sigma^{-2,(1)} \mu_*^{(1)})}.
\end{aligned}$$

In the limit of an infinite number of scans, this becomes:

$$\lim_{K\to\infty} p(y_* = 1 \mid \mathcal{D}_{1:K}) = \begin{cases} 1, & \text{if } \mu_*^{(1)} > 0 \\ 0.5, & \text{if } \mu_*^{(1)} = 0 \\ 0, & \text{if } \mu_*^{(1)} < 0, \end{cases} \tag{26}$$

so we observe that the predicted probability will be driven toward 1 or 0 if the predicted mean is nonzero, else it will remain at 0.5. It is this process that causes the artifacts observed in Figure 13. The nonparametric Bayesian inference method differs in that our estimation operates directly on probabilities. By performing this update to the variance in Equation (6), we can easily find that the variance decreases with each identical set of data. However, the equivalent update to the mean (5) for K datasets with our proposed formulation is as follows:

$$\begin{aligned}
\boldsymbol{E}[\theta] &= \frac{\alpha}{\alpha + \beta} \\
&= \frac{\alpha_0 + \sum_{j=1}^{K} \sum_{i=1}^{N} k(x_i, x_*) y_i}{\alpha_0 + \beta_0 + \sum_{j=1}^{K} \sum_{i=1}^{N} k(x_i, x_*)} \\
&= \frac{\alpha_0 + K \sum_{i=1}^{N} k(x_i, x_*) y_i}{\alpha_0 + \beta_0 + K \sum_{i=1}^{N} k(x_i, x_*)},
\end{aligned}$$

which, when there is sufficient training data, can be approximated by the Nadaraya-Watson estimator [33], [34], denoting the estimate of the mean after K updates as $\hat{m}_*$ and after one scan as $\hat{m}_*^{(1)}$, as:

$$\begin{aligned}
\hat{m}_* &= \frac{K \sum_{i=1}^{N} k(x_i, x_*) y_i}{K \sum_{i=1}^{N} k(x_i, x_*)} \\
&= \frac{\sum_{i=1}^{N} k(x_i, x_*) y_i}{\sum_{i=1}^{N} k(x_i, x_*)} \\
&= \hat{m}_*^{(1)}
\end{aligned}$$

and so the estimate of the mean remains unchanged whenever this approximation holds, while, we note, the predicted variance will decrease due to the additional observed data.

ACKNOWLEDGMENTS

We thank Prof. Philippos Mordohai for fruitful discussions and feedback that have improved this manuscript.

REFERENCES

[1] A. Elfes, "Using Occupancy Grids for Mobile Robot Perception and Navigation," *Computer*, vol. 22(6), pp. 46-57, 1989.

[2] H.P. Moravec and A. Elfes, "High Resolution Maps from Wide Angle Sonar," *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 116-121, 1985.

[3] S.T. O'Callaghan and F.T. Ramos, "Gaussian Process Occupancy Maps," *The International Journal of Robotics Research*, vol. 31(1), pp. 42-62, 2012.

[4] F. Ramos and L. Ott, "Hilbert Maps: Scalable Continuous Occupancy Mapping with Stochastic Gradient Descent," *Proceedings of Robotics: Science and Systems*, 2015.

[5] K. Doherty, J. Wang and B. Englot, "Bayesian Generalized Kernel Inference for Occupancy Map Prediction," *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3118-3124, 2017.

[6] D. Hähnel, R. Triebel, W. Burgard and S. Thrun, "Map Building with Mobile Robots in Dynamic Environments," *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1557-1563, 2003.

[7] S. Thrun, "Learning Occupancy Grid Maps with Forward Sensor Models," *Autonomous Robots*, vol. (15)2, pp. 111-127, 2003.

[8] P. Biber and W. Straßer, "The Normal Distributions Transform: A New Approach to Laser Scan Matching," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2743-2748, 2003.

[9] J.P. Saarinen, H. Andreasson, T. Stoyanov and A.J. Lilienthal, "3D Normal Distributions Transform Occupancy Maps: An Efficient Representation for Mapping in Dynamic Environments," *The International Journal of Robotics Research*, vol. 32(14), pp. 1627-1644, 2013.

[10] S. Kim and J. Kim, "GPmap: A Unified Framework for Robotic Mapping based on Sparse Gaussian Processes," *Proceedings of the 9th International Conference on Field and Service Robotics*, 2013.

[11] S. Kim and J. Kim,"Recursive Bayesian Updates for Occupancy Mapping and Surface Reconstruction," *Proceedings of the Australasian Conference on Robotics and Automation*, 2014.

[12] J. Wang and B. Englot, "Fast, Accurate Gaussian Process Occupancy Maps via Test-Data Octrees and Nested Bayesian Fusion," *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1003-1010, 2016.

[13] V. Tresp, "A Bayesian Committee Machine, *Neural Computation*, vol. 12(11), pp. 2719-2741, 2000.

[14] K. Doherty, J. Wang and B. Englot, "Probabilistic Map Fusion for Fast, Incremental Occupancy Mapping with 3D Hilbert Maps," *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1011-1018, 2016.

[15] V. Guizilini and F. Ramos, "Large-scale 3D Scene Reconstruction with Hilbert Maps," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3247-3254, 2016.

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

[17] M.G. Jadidi, J.V. Miro, R. Valencia and J. Andrade-Cetto, "Exploration on Continuous Gaussian Process Frontier Maps," *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 6077-6082, 2014.

[18] A.-A. Agha-Mohammadi, E. Heiden, K. Hausman and G. Sukhatme, "Confidence-rich Grid Mapping," *Proceedings of the 18th International Symposium on Robotics Research*, 2017.

[19] E. Heiden, K. Hausman, G.S. Sukhatme and A.-A. Agha-Mohammadi, "Planning High-speed Safe Trajectories in Confidence-rich Maps," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2880-2886, 2017.

[20] A. Melkumyan and F. Ramos, "A Sparse Covariance Function for Exact Gaussian Process Inference in Large Datasets," *Proceedings of the International Joint Conferences on Artificial Intelligence Organization*, vol. 9, pp. 1936-1942, 2009.

[21] W. Vega-Brown, M. Doniec and N. Roy, "Nonparametric Bayesian Inference on Multivariate Exponential Families," *Advances in Neural Information Processing Systems*, pp. 2546-2554, 2014.

[22] S.T. O'Callaghan and F.T. Ramos, "Continuous Occupancy Mapping with Integral Kernels," *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 1494-1500, 2011.

[23] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss and W. Burgard, "OctoMap: An Efficient Probabilistic 3D Mapping Framework based on Octrees," *Autonomous Robots*, vol. 34(3), pp. 189-206, 2013.

[24] N. Koenig and A. Howard, "Design and Use Paradigms for Gazebo, an Open-Source Multi-Robot Simulator," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, pp. 2149-2154, 2004.

[25] University of Freiburg OctoMap 3D Scan Dataset http://ais.informatik.uni-freiburg.de/projects/datasets/octomap/

[26] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler and A. Ng, "ROS: An Open-Source Robot Operating System," *IEEE ICRA Workshop on Open Source Software*, 2009.

[27] R. Rusu and S. Cousins, "3D is Here: Point Cloud Library (PCL)," *Proceedings of the IEEE International Conference on Robotics and Automation*, 2011.

[28] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real-time," *Proceedings of Robotics: Science and Systems*, 2014.

[29] M.G. Jadidi, J.V. Miro and G. Dissanayake, "Warped Gaussian Processes Occupancy Mapping With Uncertain Inputs," *IEEE Robotics and Automation Letters*, vol. 2(2), pp. 680-687, 2017.

[30] J. Wang, S. Bai and B. Englot, "Underwater Localization and 3D Mapping of Submerged Structures with a Single-Beam Scanning Sonar," *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 4898-4905, 2017.

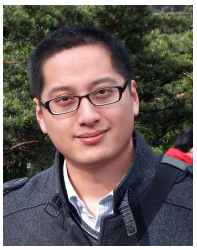
[31] V. Guizilini and F. Ramos, "Large-scale 3D Scene Reconstruction with Hilbert Maps," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3247-3254, 2016.

[32] C.E. Rasmussen and C.K.I. Williams, *Gaussian Processes for Machine Learning*, Cambridge, MA: The MIT Press, 2006.

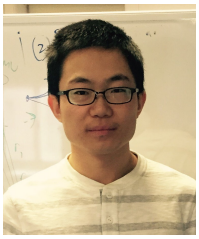
[33] E.A. Nadaraya, "On Estimating Regression," *Theory of Probability & Its Applications*, vol. 9(1), pp. 141-142, 1964.

[34] G.S. Watson, "Smooth Regression Analysis," *Sankhya: The Indian Journal of Statistics, Series A*, vol. 26(4), pp. 359-372, 1964.

Kevin Doherty (S'14) received a Bachelor of Engineering in Electrical Engineering from Stevens Institute of Technology, Hoboken, NJ, USA, in 2017. He is currently a Ph.D. student in the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, USA, and a member of the MIT-Woods Hole Oceanographic Institute Joint Program in Applied Ocean Science and Engineering.



Tixiao Shan (S'17) received a Bachelor of Science in Mechanical Engineering and Automation from Qingdao University, Qingdao, China, in 2011, and a Master's Degree in Mechatronic Engineering from Shanghai University, Shanghai, China, in 2014. He is currently a Ph.D. student in the Department of Mechanical Engineering, Stevens Institute of Technology, Hoboken, NJ, USA.



Jinkun Wang received a Bachelor of Science in Mechanical Engineering from the University of Science and Technology of China, Hefei, China, in 2014, and a Master of Engineering in Mechanical Engineering from Stevens Institute of Technology, Hoboken, NJ, USA, in 2016. He is currently a Ph.D. student in the Department of Mechanical Engineering, Stevens Institute of Technology.

Brendan Englot (S'11–M'13) received S.B., S.M., and Ph.D. degrees in Mechanical Engineering from Massachusetts Institute of Technology, Cambridge, MA, USA, in 2007, 2009, and 2012, respectively.

He was a research scientist with United Technologies Research Center, East Hartford, CT, USA, from 2012 to 2014. He is currently an Assistant Professor with the Department of Mechanical Engineering, Stevens Institute of Technology, Hoboken, NJ, USA. His research interests include motion planning, localization, and mapping for mobile robots, learning-aided autonomous navigation, and marine robotics. He is the recipient of a 2017 National Science Foundation CAREER award.