Probabilistic Map Fusion for Fast, Incremental Occupancy Mapping with 3D Hilbert Maps

Kevin Doherty, Jinkun Wang, and Brendan Englot

***Abstract*— We present a novel formulation of Hilbert mapping in which we construct a global occupancy map by incrementally fusing local overlapping Hilbert maps. Rather than maintain a single supervised learning model for the entire map, a new model is trained with each of a robot's range scans, and queried at all points within the robot's perceptual field. We treat the probabilistic output of the classifier as a sensor, employing sensor fusion to merge local maps. This formulation allows Hilbert mapping to be used incrementally in real-world mapping scenarios with overlap between sensor observations. The methodology is applied to three-dimensional map-building, and evaluated using real and simulated 3D range data.**

I. INTRODUCTION

The ability to create rich, correlated occupancy maps in real-time during robot exploration would aid localization and enable path-planning and navigation in environments where sensor data can be noisy and sparse. Traditional approaches to generating occupancy maps are limited by the assumption that the occupancy probability of each cell in a map is an independent random variable until it is observed [1], [2]. Intuitively, this assumption does not hold true, since real environments have some inherent structure. Structural elements in a map can be inferred based on a sparse representation of that map using supervised learning. Gaussian process occupancy mapping [3] provides one method, using Gaussian process regression [4], to learn the structure of a map from a sparse training set of laser range-finder or sonar data, then infer the occupancy probability of cells which have not been directly intersected by the sensor.

Applications of Gaussian process regression are constrained by its computational complexity, which is $\mathcal{O}(n^3)$ in training and $\mathcal{O}(n^2 m)$ in test, where n is the number of training points and m is the number of test points. Nonetheless, it is desirable to have accurate inference capabilities, especially when the observations from sensor data are sparse. In an effort to provide a method which can perform comparable inference to Gaussian process occupancy mapping but is highly scalable to large datasets, Hilbert maps were recently developed by Ramos and Ott [5]. Hilbert maps have been shown to be as accurate as Gaussian process occupancy maps, but offer several advantages at scale. First, they implement a formulation of the logistic regression classifier which can be trained by stochastic gradient descent. The computational complexity of training a linear stochastic gradient descent model is $\mathcal{O}(kn\bar{p})$ where k is the number of non-zero attributes and $\bar{p}$ is the number of iterations or "epochs." This offers the benefits that training is linear in the size of the training data, as opposed to cubic in the case of Gaussian process regression, and it also suggests that we may speed up computation by introducing sparsity into the model weights. Since linear models are not capable on their own of representing nonlinear spatial relationships, a radial basis function (RBF) kernel is used in conjunction with a linear classifier to learn non-linear decision boundaries. The RBF kernel requires computation of pairwise distances among all training samples, but there exist several approximations of the RBF kernel which make kernel transformation much faster, with a limited tradeoff in accuracy.

While Hilbert maps have these advantages over Gaussian process occupancy mapping, the logistic regression model used in Hilbert maps does not provide the covariance information necessary to implement fusion using a Bayesian Committee Machine (BCM) [6], in contrast to Gaussian process regression, where the BCM has aided online incremental map fusion [7]. The original Hilbert mapping formulation requires that we maintain and update a single global classifier, however it is desirable to perform scalable, incremental updates to a map as new data is gathered in real-time. Our proposed formulation of Hilbert maps allows for the use of multiple estimators by enabling local map fusion between overlapping scans, and its results also comprise the first application of Hilbert maps in 3D.

In Section II we review in some detail the machine learning techniques relevant to our implementation of Hilbert maps, including logistic regression, nonlinear kernel approximation, and stochastic gradient descent training. In Section III we present a novel interpretation of estimator outputs, where each estimator is treated as a sensor that outputs occupancy probabilities. With this interpretation we apply sensor fusion to merge local maps. This technique, to our knowledge, is the first to allow the combination of predictions from multiple local estimators in the abscence of covariance information (i.e., where methods like the BCM cannot be applied). In Section IV we present computational results where we quantitatively compare our method to both OctoMap [8] and global Hilbert mapping on two simulated datasets. We then demonstrate the inference capability of the method over real data from the University of Freiburg [9]. With this method, we hope to bring supervised learning-aided occupancy mapping into the realm of real-time autonomous robotics, where it will hopefully benefit robots navigating

K. Doherty is with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, and J. Wang and B. Englot are with the Department of Mechanical Engineering, Stevens Institute of Technology, Castle Point on Hudson, Hoboken NJ 07030, USA {kdoherty, jwang92, benglot}@stevens.edu

under sparse and noisy data.

II. HILBERT MAPS

Hilbert maps combine several advances in machine learning to provide results comparable to Gaussian process occupancy mapping in a way that scales more suitably to the large amount of data from incoming sensors that must be processed during mapping. These advances include approximations to nonlinear kernel functions to speed up computation of kernel transformations and stochastic gradient descent for both training and updating the aforementioned classifiers in linear time.

A logistic regression [10] classifier is chosen, since its outputs most naturally resemble continuous probabilities, whereas alternative classifiers, like the Support Vector Machine (SVM) [11], provide only binary outputs. While there exist methods [12] to generate artificial probabilistic outputs from SVMs, it has been demonstrated by Ramos and Ott [5] that such methods often provide results claiming high degrees of confidence in regions with relatively few training points.

A. Logistic Regression

We adopt a nonlinear formulation of a logistic regression classifier in which the predicted probability of occupancy for a point $\mathbf{x}_*$ given a set of training points $\mathbf{x}$ and model weights $\mathbf{w}$ is as follows:

$$p(y_* = 1|\mathbf{x}_*, \mathbf{w}) = \frac{1}{1 + \exp(\mathbf{w}^T k(\mathbf{x}_*, \mathbf{x}))} \tag{1}$$

where $k(\cdot)$ denotes the kernel function mapping input 3-dimensional coordinates to a high-dimensional feature space. Probability of non-occupancy for a cell $p(y_* = -1|\mathbf{x}_*, \mathbf{w})$ is equivalent to $1 - p(y_* = -1|\mathbf{x}_*, \mathbf{w})$, since we maintain that the state of a particular cell is binary.

The major benefit of this logistic regression formulation over Gaussian process regression is that this method can be trained very quickly using Stochastic Gradient Descent (SGD) and evaluation is $\mathcal{O}(n)$ where n is the size of the test data. As a result, computation of the kernel transformation becomes the most costly part of classification.

B. Nonlinear Kernel Approximation

In order to represent the non-linear spatial dependency of occupancy in 3-dimensional space, we use the radial basis function (RBF) kernel. The kernel transformation is computed as:

$$k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2) \tag{2}$$

where the parameter γ is tuned in cross-validation. The RBF kernel maps each sample in the training data $\mathbf{x}$ to a feature vector containing the pairwise distances between that sample and every other training sample. During prediction, the distance between each query point and every training point must be computed to generate a corresponding test feature vector. This pairwise distance computation becomes computationally expensive as we scale the data. This presents difficulty in applying these techniques to large datasets, especially if we seek to perform real-time inference during map traversal.

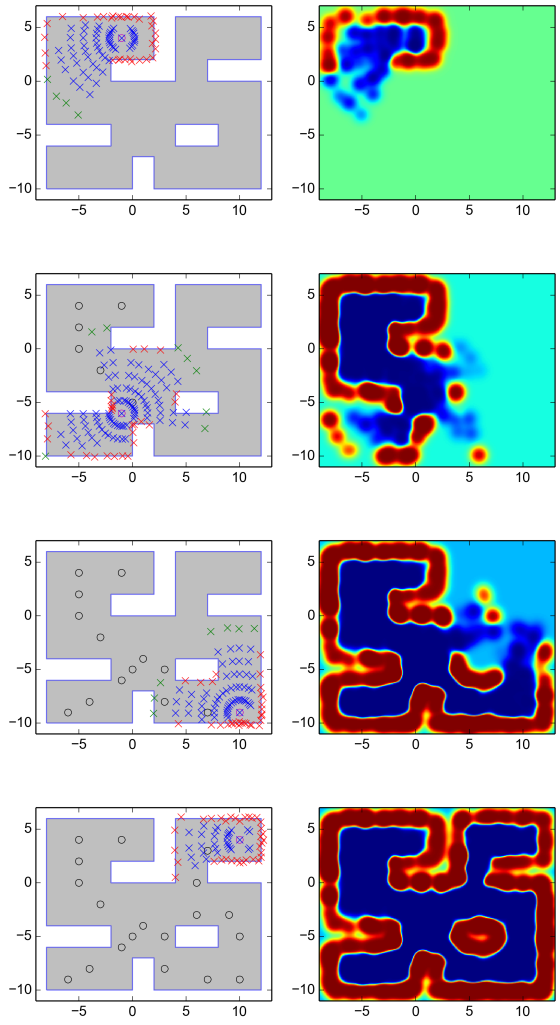


Fig. 1: A representative 2D case of incremental overlapping Hilbert maps applied to a simulated environment. Logistic regression classifiers were trained on range sensor data from each scan and evaluated at every cell (5 cm resolution) in the map at each step. Scans 1, 7, 14, and 20 of 20 are shown. A magenta circle marks the robot's current location and black circles represent the robot's previous locations. Red markers denote hit points, blue markers denote free points, and green markers denote points assumed free where the simulated laser reached its maximum range without encountering any obstacles. In this example, the occupancy probability of unobserved regions decreases over the course of traversal, which reflects the updated belief after observation that new cells are more likely to be unoccupied than occupied.

For this reason, we opt to approximate the full RBF kernel. Specifically, we use the Nyström method [13] to approximate the RBF kernel with the projection of the original kernel onto a set of m inducing points $\hat{\mathbf{x}}_{1\ldots m}$, where we may choose the value of m. In this method, we seek to find a function $\phi_{\text{Nyström}}$

that satisfies the following:

$$k(\mathbf{x}, \mathbf{x}') \approx \phi_{\text{Nyström}}(\mathbf{x})\phi_{\text{Nyström}}(\mathbf{x}'). \tag{3}$$

The resulting feature transformation of $\mathbf{x}$ is as follows:

$$\phi_{\text{Nyström}}(\mathbf{x}) = D^{-1/2}V^T(k(\mathbf{x}, \hat{\mathbf{x}}_1), \ldots, k(\mathbf{x}, \hat{\mathbf{x}}_m)) \tag{4}$$

where D is a diagonal matrix containing the decreasing non-negative eigenvalues of the kernel computed between all m inducing points and V is the set of its respective eigenvectors. The Nyström method was selected due to the promising results demonstrated in [5]. Kernel computation remains the most expensive step in the classification process, though this approximation offers a significant increase in speed.

C. Stochastic Gradient Descent

The general optimization problem we seek to solve in SGD is to find the weights $\mathbf{w}$ and bias b which minimize the regularized training error

$$E(\mathbf{w}, b) = \frac{1}{n}\sum_{i=1}^{n} L(\mathbf{y}_i, f(\mathbf{x}_i)) + \alpha\|\mathbf{w}\|_1 \tag{5}$$

where $L(\mathbf{y}_i, f(\mathbf{x}_i))$ is the loss function of our choosing, α is a regularization parameter, and $\|\mathbf{w}\|_1$ is the L1-norm of the weights. For linear logistic regression, the loss function, after substitution of the hypothesis function from Equation 1 for $f(\mathbf{x}_i)$ is as follows:

$$L(\mathbf{y}_i, f(\mathbf{x}_i)) = \log(\exp(-\mathbf{y}_i(\mathbf{x}_i^T\mathbf{w} + b)) + 1). \tag{6}$$

To use this loss function for non-linear decision boundaries we replace $\mathbf{x}$ with the feature vectors from the kernel computation $k(\mathbf{x}, \mathbf{x}')$. We choose to regularize the L1-norm, since we are also interested in allowing feature selection to take place during learning. Ideally, features which have no significant bearing on the occupancy of a region will be eliminated, and the sparsity of the resulting weights will speed up evaluation of the model during testing.

III. PROBABILISTIC LOCAL MAP FUSION

Previously there have been several general approaches to constructing a global map from local maps. One approach is to develop a single classifier trained on accumulated data from across the map and predict values in postprocessing of the entire map. This is characteristic of the original methods of Gaussian process occupancy mapping. While this allows the classifier to choose informative features from the entire map, the resulting correlated occupancy map cannot be used during local planning or exploration.

Kim and Kim [14] have demonstrated overlapping map fusion for local Gaussian processes with the Bayesian Committee Machine (BCM) [6]. Such a method is useful in the case of Gaussian processes, where the outputs of the estimator have both predicted mean and variance information. This variance information provides insight into the uncertainty of the estimator, which can be used to make informed decisions when updating a grid cell with more than one prediction. The logistic regression estimator we use, however, does not provide variance information for its predictions.

In [5] a method has been presented to incrementally update a single global estimator given new training data. As in Gaussian process occupancy mapping, this allows a map to be produced at arbitrary resolution which very effectively captures the spatial correlation among all cells. We seek, however, to apply the methods from Hilbert maps in a way that allows us to use the inference capability of estimators to incrementally update maps in real-time. This will allow the maps to support fast decision-making in the course of exploring unknown environments, as with Gaussian processes in [7].

Rather than maintaining a single classifier which we train online as the robot traverses the map, we opt instead to train a new logistic regression classifier at every new scan. This allows for real-time inference during exploration. In order for this method to be effective, we must handle overlapping estimator predictions. There are several consequences of this method, one of which is the loss of the multi-resolution property of Hilbert maps. In our method, a resolution must be chosen *a priori* in order to update the map cells. Using one incrementally updated classifier, a complete map of arbitrary resolution could be constructed *a posteriori*. However, without querying the classifiers during the exploration process, decisions regarding path-planning and navigation must be made using only the sparse sensor data. Despite this limitation, our proposed method is compatible with parameterizations that improve robustness. By applying the Reproducing Kernel Hilbert Spaces (RKHS) demonstrated in [5] to our incremental formulation, it is possible to maintain the robustness of Hilbert maps to noisy data, extending the applicability of the method.

A. Map Update Algorithm

To solve the problem of merging overlapping local Hilbert maps to produce a global map, we make the assumption that the map is static. Formally, the Markov assumption, or static world assumption, states

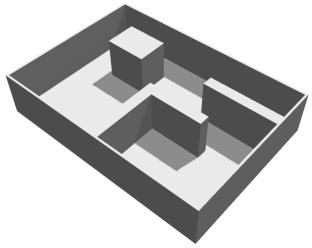
$$p(\mathbf{y}_t|\mathbf{m}, \mathbf{y}_{1:t-1}) = p(\mathbf{y}_t|\mathbf{m}) \tag{7}$$

given the map $\mathbf{m}$. That is, we assume the current observation $\mathbf{y}_t$ and previous observations $\mathbf{y}_{1:t-1}$ are independent, and the only state for which the ground truth may vary during exploration is the pose of the robot.

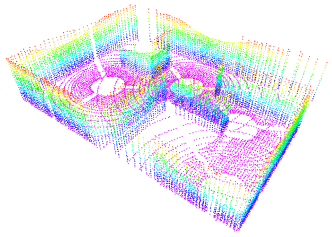
Under this assumption, we consider the problem of updating the occupancy probability to be analogous to integrating the outputs of several sensors which intersect the same cell in the case of occupancy grid mapping [15] or OctoMapping [8]. We generalize the sensor fusion update rule for all overlapping estimator predictions by treating each estimator as a sensor in itself, where the output of the sensor for each cell is the predicted occupancy probability for the cell. We formalize this notion with the relationship

$$p(\mathbf{m}_i|\hat{\mathbf{y}}_t) = \hat{\mathbf{y}}_t = p(y_* = 1|\mathbf{x}_*, \mathbf{w}_t) \tag{8}$$

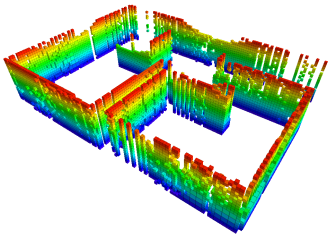
where $\mathbf{x}_*$ is the centroid of the query cell and $\mathbf{w}_t$ are the learned weights for the logistic regression model trained on data from the scan at time t.



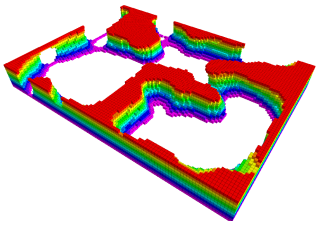
(a) The ground truth map



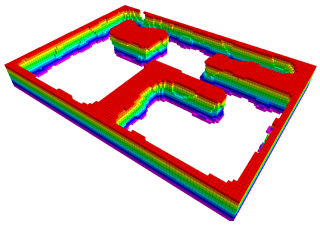
(b) The raw sensor data



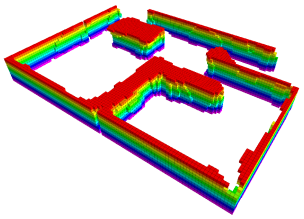
(c) The result of OctoMap



(d) The result of global Hilbert maps ($m = 100$)



(e) The result of global Hilbert maps ($m = 1000$)



(f) The result after incrementally applying overlapping Hilbert maps

Fig. 2: The results of 3D Hilbert mapping applied to a simulated environment. Training data is provided by the raw sensor data in (b) and the model is evaluated for all cells within the perceptual field of the robot at each scan. Note that the global Hilbert map with $m = 100$ in (d) fails to effectively capture the sharp changes in occupancy probability in this "structured" environment, while the equivalent classifier is much more effective when updated at each new scan. We apply the update rule from Algorithm 1 to fuse overlapping local Hilbert maps and form the global map seen in (f).

Algorithm 1 $updateMap(\mathcal{X}, \mathcal{Y})$

$K \leftarrow fitApproxRBF(\mathcal{X}, \mathcal{X})$;
$model \leftarrow train(K, \mathcal{Y})$;
$\mathcal{M} \leftarrow$ all cells $\in Map$ within bounding box of $\mathcal{X}$;
$\mathcal{X}_* \leftarrow m_i \in \mathcal{M}$ **if** m_i in robot's perceptual field;
$query \leftarrow K.tranformApproxRBF(\mathcal{X}_*)$;
$\hat{\mathcal{Y}} \leftarrow model.predict(query)$;
for $\hat{y}_i \in \hat{\mathcal{Y}}$ **do**
    $p_i \leftarrow [1 + (1 - p_i)/p_i * (1 - \hat{y}_i)/\hat{y}_i]^{-1}$;
end for

Method	AUC	Precision	Recall
OctoMap	0.92	0.320	0.097
Global Hilbert Map ($m = 100$)	0.91	0.792	0.108
Global Hilbert Map ($m = 1000$)	0.97	0.661	0.877
Overlapping Hilbert Maps	0.97	0.709	0.868

TABLE I: Numerical receiver operating characteristic (ROC) curve comparison of the methods tested on the "structured" map shown in Figure 2. Precision and recall scores were computed using a threshold occupancy probability of 0.7.

With this interpretation of the classifier output, applying the update rule gives the updated occupancy probability for grid cell $\mathbf{m}_i$ given the current prediction $\hat{\mathbf{y}}_t$ from the output of the logistic regression classifier as

$$p(\mathbf{m}_i|\hat{\mathbf{y}}_t) = \left[1 + \frac{1 - p(\mathbf{m}_i|\hat{\mathbf{y}}_{1:t-1})}{p(\mathbf{m}_i|\hat{\mathbf{y}}_{1:t-1})} \frac{1 - p(\mathbf{m}_i|\hat{\mathbf{y}}_t)}{p(\mathbf{m}_i|\hat{\mathbf{y}}_t)} \frac{p(\mathbf{m}_i)}{1 - p(\mathbf{m}_i)}\right]^{-1}. \quad (9)$$

The updated occupancy probability $p(\mathbf{m}_i|\hat{\mathbf{y}}_t)$ depends only on the current prediction, the previous prediction $p(\mathbf{m}_i|\hat{\mathbf{y}}_{1:t-1})$, and the prior occupancy probability $p(\mathbf{m}_i)$. In practice the prior occupancy probability is assumed 0.5.

The sensor fusion update rule as it applies to combining estimator predictions has the property that more informative predictions have more influence on the updated occupancy probability. That is, an estimated occupancy probability close to 0.5 offers little change to the final occupancy probability, since the estimate provides very little new information. Additionally, outputs that conflict (i.e. a low occupancy probability followed by a high occupancy probability) cause the update to tend toward 0.5, so that the lack of consensus among "sensors" is reflected in the updated probability. Since there is some degree of redundancy in the data from consecutive scans, the result of this update rule is that the various classifiers trained on data from different scans may serve to improve on each others' predictions.

The complete map update algorithm is presented in Algorithm 1, including the kernel computation step, the training and querying of a logistic regression classifier, and the application of the update rule to overlapping estimator predictions. The function $fitApproxRBF(\mathcal{X}, \mathcal{X})$ computes the Nyström approximation to the RBF kernel using the provided training data. The function $transformApproxRBF(\mathcal{X}_*)$ transforms the coordinates of each cell in the set of test points $\mathcal{X}_*$ to a feature vector using the previously fit RBF kernel approximation. All $p_i \in \mathcal{P}$ denote the occupancy probabilities for the corresponding cells $m_i \in \mathcal{M}$. Map is the set of all cells of a previously selected resolution

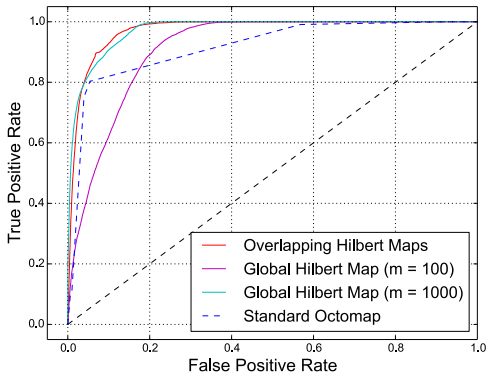


Fig. 3: Resulting receiver operating characteristic (ROC) curves for the "structured" environment depicted in Figure 2, comparing incremental overlapping Hilbert maps with the same classifier trained on data across the entire map and evaluated once for each cell in the global map. Standard OctoMaps are also compared.

in the global map, each initialized to the prior occupancy probability 0.5. An illustrative 2D example of the incremental construction of an occupancy map using Algorithm 1 is shown in Figure 1.

IV. COMPUTATIONAL RESULTS

The proposed algorithm was evaluated in two simulated environments and one real environment. The first of the two simulations provides a "structured" environment, representative of an indoor setting. The second simulation provides an "unstructured" environment, emulating navigation of a forest. We quantitatively compare the accuracy and computation time of our method, fusing several local overlapping Hilbert maps (each with 100 Nyström components), with a formulation of Hilbert mapping, which we call *global* Hilbert maps, that trains and queries a single predictor after map traversal (evaluated with 100 and 1000 Nyström components). We also compare our method with OctoMaps. In simulation, each method was directly compared against ground truth. The beams from the simulated laser range finder were sampled at regular intervals, providing training data for the unnocupied region, as in Figure 1. We then provide the visual output of inference for real laser data from the University of Freiburg. Finally, we demonstrate the applicability of this method to noisy sensor data using the RKHS method. We use $\gamma = 3.0$ for the RBF kernel except where otherwise noted. The tests for both the simulated data and real data were built using the Robot Operating System (ROS) [16]. The Gazebo Simulator [17] was used to develop the quantitative tests in the simulated environment. The logistic regression classifier and kernel approximation functions were provided by the scikit-learn [18] Python machine learning library. All computation was performed on an HP EliteBook 8570w with a 2.40 GHz Intel i7 CPU. Our goals in this section are three-fold: to show that fusing local overlapping Hilbert maps provides results comparable to the equivalent global Hilbert map, to demonstrate that Hilbert mapping can be applied in realistic 3D mapping scenarios and achieve near real-time performance, and to compare the performance of these methods with standard OctoMaps [8]. The results will demonstrate that compared to OctoMaps, both global and incremental overlapping Hilbert maps produce more accurate maps when compared to ground truth, and they provide useful predictions under sparse coverage of the environment.

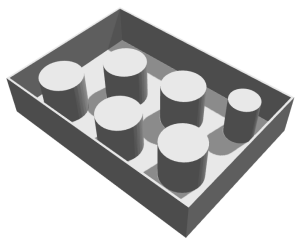
A. Structured Simulation

The "structured" environment represents a 10.0 m $\times$ 7.0 m $\times$ 2.0 m indoor setting, containing mostly rectangular obstacles. Some artifacts of the RBF kernel approximation can be found in the results of Hilbert mapping. For example, regions which in the ground truth map contained sharp corners are rounded in the corresponding Hilbert map. Both of the simulated environments were mapped using a laser range-finder with a 120° field of view. In each map, twelve scans were processed, comprised of four unique positions that were each scanned from three different heading angles. In Figure 2, we present the visual results after applying OctoMapping, global Hilbert mapping with 100 and 1000 Nyström components, and incremental overlapping Hilbert maps, as well as the ground truth map and raw sensor data. The sharp boundary edge is due to the use of bounding boxes around the training data for each scan to obtain the points that were used to query the classifier. Figure 3 shows the receiver operating characteristic (ROC) curves for the different methods on the structured map. The comparison of the area under the ROC curve (AUC) is provided in Table I, along with the precision and recall for each method with an occupancy probability threshold of 0.7, which shows that OctoMap tends to misclassify occupied voxels as unoccupied voxels, while the other methods more accurately predict the occupied voxels.

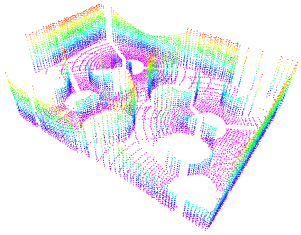
We found that using only 100 Nyström components for global Hilbert mapping was not enough to effectively capture the sharp changes in occupancy probability found on this map. However, the equivalent classifier (100 components) quite effectively captured the structure of the map by comparison when evaluated for each scan and updated using our incremental map fusion method. When we increased the number of Nyström components to 1000 we found that we could produce a slightly more accurate global map than incremental overlapping Hilbert maps, but with significant additional computation time required, shown in Table III.

B. Unstructured Simulation

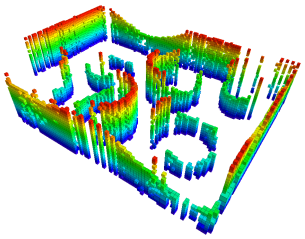
Our "unstructured" simulation replicates the case of navigating a forest region, dimensioned 10.0 m $\times$ 7.0 m $\times$ 2.0 m, containing several rounded obstacles with smaller perimeters. Additionally, many sections of the environment are obscured during traversal, leading to more sparsity in the training set. The surrounding walls, while adding some structure to support the mapping task, demonstrate the capability of these techniques to handle sensor data from a heterogeneous environment. The ROC curves from the methods evaluated



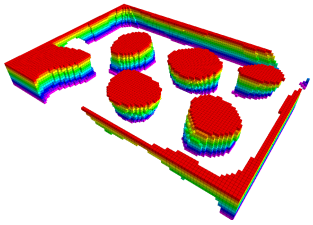
(a) The ground truth map



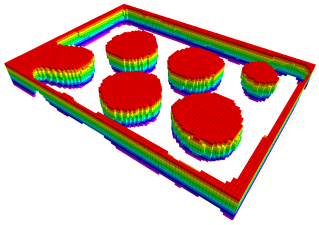
(b) The raw sensor data



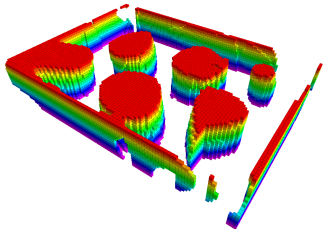
(c) The result of OctoMap



(d) The result of global Hilbert maps ($m = 100$)



(e) The result of global Hilbert maps ($m = 1000$)



(f) The result after incrementally applying overlapping Hilbert maps

Fig. 4: The results of 3D Hilbert mapping applied to a simulated "unstructured" environment. Training data is provided by the raw sensor data in (b). A standard OctoMap applied to the sensor data is shown in (c). Global Hilbert mapping with m = 1000 requires several minutes to compute, but generates a much more accurate, complete map. In (f) we show the results of incremental overlapping Hilbert maps, updated using the local map fusion algorithm presented in Algorithm 1.

on the map are shown in Figure 5. In the case of 100 Nyström components, the area under the ROC curve for fused overlapping Hilbert maps and a single global Hilbert map were comparable. If the number of components used was increased, the AUC for the global Hilbert maps method, shown in Table II, increased as well. Precision and recall scores are also provided in Table II, where again we observe the low recall score of OctoMap compared to the other methods. As in the case of the "structured" simulation, significant additional computation time was needed for the global Hilbert map to achieve a marginal benefit over the incremental mapping method. Increasing the number of Nyström components for incremental overlapping Hilbert maps did not significantly influence results, and if the predetermined number of Nyström components was greater than the number of training samples, standard evaluation of the full RBF kernel is preferred. It is intuitive that a larger number of components would better approximate a larger training set, as in the case of a global Hilbert map, but the number of training points per scan is relatively small compared to the data from the entire map, and thus less benefit comes from increasing the number of components for the incremental Hilbert mapping method. If the data from a scan is sparse, the number of incoming samples is often few enough that full evaluation of the RBF kernel is possible.

C. Real Data

We performed inference for qualitative evaluation on data from the University of Freiburg corridor OctoMap dataset[1].

[1] http://ais.informatik.uni-freiburg.de/projects/datasets/octomap/

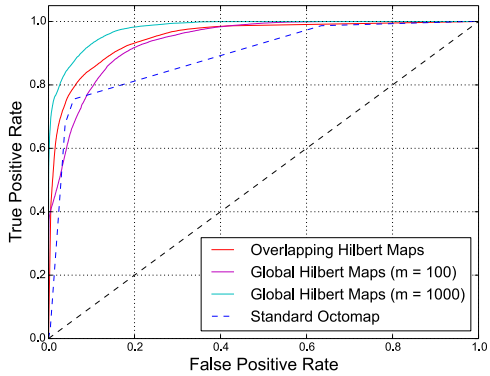


Fig. 5: ROC curves for the "unstructured" environment depicted in Figure 4, comparing incremental overlapping Hilbert maps with the same classifier trained on data across the entire map and evaluated once for each cell in the global map. Standard OctoMaps are also compared.

The entire map measures 43.8 m $\times$ 18.2 m $\times$ 3.3 m. 360° pan-tilt laser scans were sparsified to generate training sets comprised of approximately 6% of the original data. Figure 6 shows a view from outside the corridor produced by incremental overlapping Hilbert maps ($m = 100$, $\gamma = 10.0$ for the kernel approximation), compared to the sparsified raw data from the range sensor. Incremental overlapping Hilbert maps produce a denser, more complete map than the map produced by the sensor data alone. In Figure 7, we show the maps from a perspective within the corridor. In both Figures 6 and 7 we observe that while this method is capable of

Dataset	Scans	Points/Scan	Test Points/Scan	Method	Time/Scan (s)	Time (s)
Structured Simulation	12	3500	29892	Overlapping Hilbert Maps	1.89	22.68
				Global Hilbert Map ($m = 100$)	N/A	8.70
				Global Hilbert Map ($m = 1000$)	N/A	331.64
				OctoMap	0.02	0.2
Unstructured Simulation	12	3500	29892	Overlapping Hilbert Maps	1.83	21.96
				Global Hilbert Map ($m = 100$)	N/A	8.52
				Global Hilbert Map ($m = 1000$)	N/A	332.82
				OctoMap	0.01	0.14
Freiburg Corridor FR-079	66	4943	371170	Overlapping Hilbert Maps	14.6	963.6
				Global Hilbert Map ($m = 100$)	N/A	1656.95
				Global Hilbert Map ($m = 1000$)	N/A	11914.96
				OctoMap	0.1	6.7

TABLE III: Computation times for mapping the three environments examined.

Method	AUC	Precision	Recall
OctoMap	0.89	0.418	0.042
Global Hilbert Map ($m = 100$)	0.94	0.999	0.248
Global Hilbert Map ($m = 1000$)	0.98	0.990	0.629
Overlapping Hilbert Maps	0.95	0.797	0.832

TABLE II: A numerical comparison of the methods tested on the "unstructured" map depicted in Figure 4. Precision and recall metrics are computed for an occupancy threshold value of 0.7.

reasonable inference in a real environment, 100 components is not enough to capture the sharp changes in occupancy probability that occur throughout this environment. Similarly, when a single classifier is used, the number of components needed to accurately approximate a region also scales with the size of the region and its structural complexity, so there is a tradeoff between the quality of the approximation and computational practicality in these cases. To some degree, the incremental mapping method helps to balance this tradeoff, but the issue is nonetheless present when dealing with individual scans and more work here is needed.

Since these scans are much larger than the 120° sector scans used in simulation, updating the map takes much longer, as shown in Table III. It should be noted, however, that the time required to process each scan is dominated by the time taken to update the map after training. The mean time required only for training a new estimator using the laser range finder data was 0.5 seconds, with the rest of the time for each scan being used to query the predictor for large numbers of grid cells and updating all of the queried voxels.

D. Noise in Sensor Data

We now consider the case of sensor noise due to positional or sensor measurement uncertainty. In this demonstration, we perturb the data with Gaussian noise, $\mathcal{N}(0.0, 0.2)$ in the x- and y-direction and $\mathcal{N}(0.0, 0.02)$ in the z-direction. Figure 8 compares the output after traversing the "structured" map and fusing overlapping Hilbert maps with Nyström features to the output when the RKHS method is used. Using the RKHS method appears to provide a map closer to the maps produced without added noise, and shows that fusion of overlapping Hilbert maps is able to retain the robustness to noise provided by the original formulation of Hilbert maps.

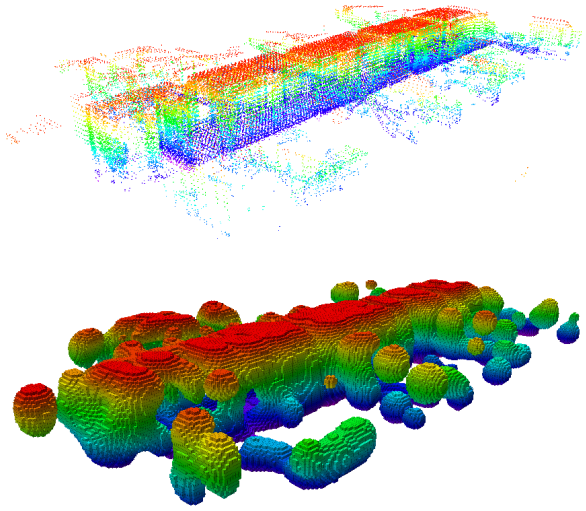


Fig. 6: The real 3D range sensor data from the Freiburg corridor map (top) compared to the result after performing incremental overlapping Hilbert maps (bottom) viewed from outside the corridor. A threshold occupancy probability of 0.7 was used. Each map is colored by height.

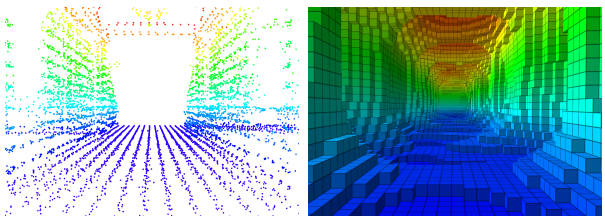


Fig. 7: View from inside the Freiburg corridor; raw data compared to the occupancy map produced using incremental overlapping Hilbert maps.

V. CONCLUSIONS AND FUTURE WORK

Hilbert maps are a recent innovation in robotic mapping, and there are many possible directions for future research with this technique. We hope this research moves toward real-time inference over occupancy maps. One concern with Hilbert maps as opposed to Gaussian process occupancy

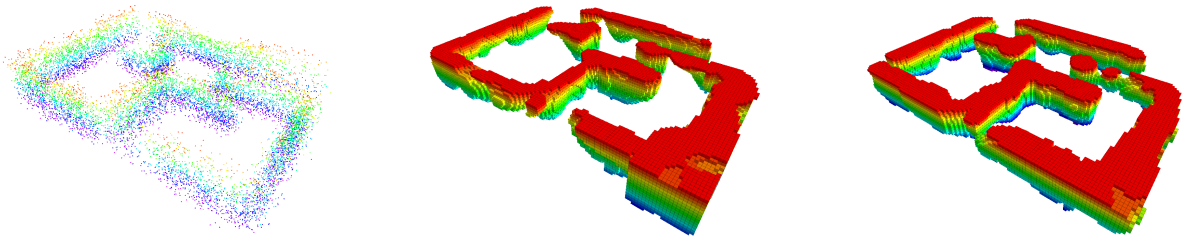


Fig. 8: Incrementally mapping the "structured" environment from Figure 2 when the range sensor is corrupted by noise. The first image (left) shows the raw sensor data. The second (center) shows the map produced using the Nyström method, as before. The rightmost image shows the same map produced using overlapping Hilbert maps with RKHS.

mapping is that Hilbert maps require parameter tuning, specifically of γ for the RBF kernel approximation and the regularization parameter α from the SGD formulation, to achieve robust performance. In principle, parameters can be tuned beforehand on similar maps. If a reserved set of sensor data from a scan is used as a cross-validation set, it is possible to perform grid search to estimate reasonable parameters automatically. Grid search is parallelizable, allowing many parameter configurations to be evaluated simultaneously. A major performance consideration for these methods is storing and querying large numbers of grid cells at fine resolution. It may be effective to apply multi-resolution techniques like OctoMap to significantly reduce the memory needed in the map update step. Updating map cells can also be done in parallel and with GPU programming, which can potentially offer a significant speed up over sequentially updating cells.

While we apply the proposed incremental map fusion method specifically to Hilbert maps, it can also be applied to the output of Gaussian process regression for fusion without the use of a BCM. Whether this technique would improve upon the results of Gaussian process occupancy mapping has yet to be determined. Another potentially promising area for future work in this local map fusion technique would be to limit the query area by filtering outliers from the training data. Further, a rigorous evaluation of kernel approximation techniques, particularly the sparse kernel approximation provided in [5], applied to 3D mapping using Hilbert maps would be interesting. Finally, it may be worthwhile to investigate the possibility of aggregating data prior to training and classification, i.e. submap fusion. For example, we train and test on 120° sector scans in our simulations, but it may be possible to increase the accuracy of estimators by aggregating data from several scans, then performing inference and fusion using the information from the larger submap. The resulting map could still be built quickly in steps, but the map construction would be less incremental, whereas one of our primary goals in this evaluation was to perform inference that assimilates each new individual scan.

Acknowledgments

This research has been supported in part by the National Science Foundation, grant number IIS-1551391.

References

[1] Elfes, Alberto. "Using occupancy grids for mobile robot perception and navigation," *Computer* vol. 22(6), pp. 46-57. 1989.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*, Cambridge, MA: MIT Press, 2005.

[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] C.K.I. Williams and C.E. Rasmussen, *Gaussian processes for machine learning*, Cambridge, MA: MIT Press, 2006.

[5] F. Ramos and L. Ott, "Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent" *Proceedings of Robotics: Science and Systems*, 9 pp., 2015.

[6] V. Tresp, "A Bayesian committee machine, *Neural Computation*, vol. 12(11), pp. 2719-2741, 2000.

[7] 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.

[8] 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.

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

[10] C.M. Bishop, *Pattern Recognition and Machine Learning*, Secaucus, NJ: Springer-Verlag, 2006.

[11] B. Scholkopf and A.J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*, Cambridge, MA: MIT Press, 2001.

[12] J. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods." *Advances in large margin classifiers*, vol. 10(3), pp. 61-74, 1999.

[13] C.K.I. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," *Advances in Neural Information Processing Systems 13*, T.K. Leen, T.G. Diettrich, and V. Tresp, Eds., Cambridge, MA: MIT Press, pp. 682-688, 2001.

[14] S. Kim and J. Kim, "Recursive Bayesian Updates for Occupancy Mapping and Surface Reconstruction," *Proceedings of the Australasian Conference on Robotics and Automation*, 8 pp., 2014.

[15] 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.

[16] M. Quigley, K. Conley, B. Gerkey, J. Faust, T.B. Foote, J. Leibs, R. Wheeler, and A.Y. Ng, "ROS: an open-source Robot Operating System." *ICRA Workshop on Open Source Software*, 6 pp., 2009.

[17] 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.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: machine learning in python." *The Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.