Approximate Continuous Belief Distributions for Precise Autonomous Inspection

Shobhit Srivastava and Nathan Michael

***Abstract*— Precise inspection of cluttered environments by computationally-constrained systems requires an efficient and high-fidelity representation of the operating space. We propose a methodology to generate perceptual models of the environment that optimally handle the variation in clutter and provide a multi-resolution and multi-fidelity representation of the environment. Further, our approach is able to capture inherent structural dependencies thereby enabling efficient and precise inference. The approach employs a hierarchy of Gaussian Mixtures to approximate the underlying spatial distribution. We make use of techniques grounded in information theory to estimate the optimal size of the mixture model and to generate and update the hierarchy of mixtures. We show that the proposed approach is superior in terms of memory requirements and accuracy as compared to other state of the art 3D mapping techniques such as NDT occupancy maps and Gaussian Process occupancy maps.**

I. INTRODUCTION

Precise infrastructure inspection forms an integral part of a variety of applications ranging from assessment of structural deterioration for bridges and roof-tops to monitoring the system state in potentially unsafe environments such as power plants. Such inspection tasks may prove to be hazardous for the people involved which underlines the need for autonomous inspection. To be able to match the precision and surpass the efficiency of a manual inspector, a high fidelity perceptual model of the operating environment is essential. Further, it should be possible to generate and update this model with new information in real-time to be effective. This model can then be used to generate maps for navigation or to obtain high-fidelity reconstructions for the purpose of inspection. Such a model serves as a representation of the system's belief and directly impacts the accuracy of inspection. A key characteristic of target environments that can be leveraged for generating compact models is the presence of structural dependencies. A model that is able to capture and exploit these inherent correlations can scale to large environments. Further, a representation that models the information in the environment can optimally handle varying degree of clutter.

The proposed strategy seeks to achieve high fidelity through a sound treatment of the information content in the environment. A hierarchy of Gaussian Mixture Models (GMM) is constructed with increasing fidelity as we move down the hierarchy. The innate ability of the model to capture structural dependencies makes it compact and efficient to infer upon.

We gratefully acknowledge support from Westinghouse and ARL grant W911NF-08-2-0004.

The authors are with the Robotics Institute, Carnegie Mellon university, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA {shobhits,nmichael}@cmu.edu.

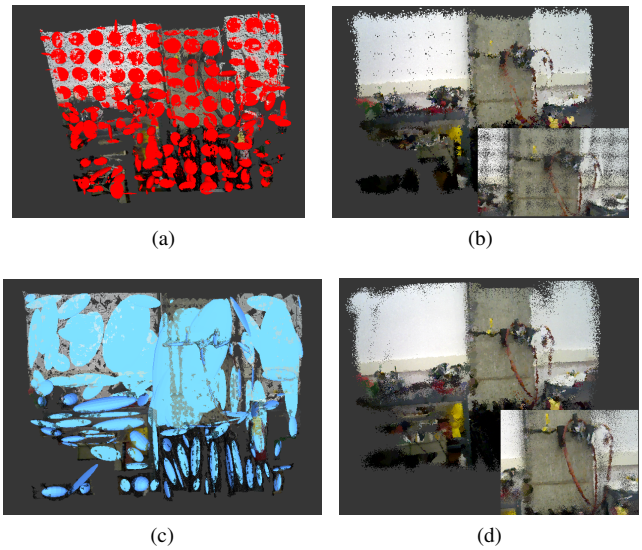


Fig. 1: The proposed methodology permits reasoning with respect to the environment without assuming a fixed resolution enabling high-fidelity representation. The NDT-OM representation with 372 decoupled Gaussians (in red) (a) results in the reconstructed point-cloud with gaps as can be seen in the zoomed-in view in (b) whereas the HGMM (proposed approach) representation with 137 coupled Gaussians (in blue) (c) provides a higher-fidelity reconstruction (d).

Traditionally, occupancy grids [1] have been used which discretize the world into fixed size cells called voxels. The state of each voxel, occupied or unoccupied, is updated from the sensor observations that fall in or pass through it. Each voxel, in this approach, is updated independently and thus fails to capture spatial dependencies and correlations inherent in the environment. This makes it vulnerable to sensor noise and leads to holes in the distribution. The size of a voxel and thus the model's resolution has to be pre-defined which makes it computationally inefficient in environments with varying amounts of detail. Octomaps [2], through on-demand sub-division of voxels, serve to make voxel grids efficient and provide a multi-resolution representation but are still vulnerable to sensor noise. An NDT occupancy map [3] provides a probabilistic representation by learning a decoupled Gaussian distribution per voxel. However, the cell independence assumption which induces decoupling of

Gaussians leads to low representation fidelity at cell boundaries (Fig. 1b). In contrast, the proposed approach learns a continuous distribution over space and is thus able to support arbitrary resolution representations.

Other techniques to generate continuous representations of the environment include Gaussian Process occupancy maps [4] and Hilbert maps [5]. Gaussian Process regression is a powerful framework for modelling surfaces as functions drawn from a Gaussian distribution. However, the complexity of querying a Gaussian process grows cubically with the size of the input which makes it intractable for direct application to large dense point clouds. The work of [6] proposes a strategy to discretize the environment into small regions based on the characteristic length-scale of the covariance function and train local Gaussian Processes in these regions. However, online training of hyper-parameters for large point clouds still poses a challenge and a single set of parameters trained offline may not be applicable for different regions in an environment with varying degrees of clutter. Bayesian Committee Machines [7] have been proposed [8] for updating the representation which restricts the model to a single fixed resolution and requires caching of mean and variance for each cell in the grid thus making it memory intensive. Hilbert maps learn a Logistic regression classifier on input data projected into Hilbert space. However, incrementally merging locally learnt logistic regression classifiers requires discretization of space [9] thus sacrificing arbitrary resolution capability of the technique. In contrast, the proposed approach supports arbitrary resolution representations and can be incrementally updated as presented in this paper. Also, the parameters required are less dependent on the environment and easy to tune. The approach is capable of estimating the optimal size of the model for a high fidelity representation via information theoretic principles. Further, our model can be described by a relatively small number of parameters making it memory efficient. A similar model to represent data as Hierarchical Gaussian Mixtures has been presented in [10]. However, a top-down hierarchy is formulated in this work aimed mainly at scan registration which does not require a strategy to incrementally update the model. Also, the question of the optimal size of the mixture is not addressed.

The paper is organized as follows. Section II lays the theoretical foundation required to develop the proposed approach. Section III develops the proposed formulation and presents the algorithms for generating and updating the model. An evaluation of the capabilities of the proposed technique and a comparison with Gaussian Process Mapping and NDT Occupancy maps is presented in Sect. IV. Future work is discussed in Sect. V.

II. GAUSSIAN MIXTURE MODEL

A Gaussian Mixture model (GMM) has been found to be a very powerful tool to model multi-modal probability distributions. It is a suitable representation for autonomous inspection due to the relatively smaller number of parameters involved and due to its ability to model multi-modal distributions with unknown parametric forms.

Given a point cloud Z of size N with points $z_i \in Z$ with i varying from 1 to N, a Gaussian Mixture model with J components is specified by its component parameters $\theta_j = (\pi_j,\ \mu_j,\ \Sigma_j)$ where π_j, μ_j and Σ_j are the prior, mean and covariance matrix for the j-th component. The likelihood of the point-cloud to be generated by this Gaussian Mixture Model is given as

$$p(Z \mid \theta) = \prod_{i=1}^{N} p(z_i \mid \theta) \tag{1}$$

$$= \prod_{i=1}^{N} \sum_{j=1}^{J} \pi_j\ p(z_i \mid \theta_j) \tag{2}$$

where

$$p(z_i \mid \theta_j) = \mathcal{N}(z_i \mid \theta_j) \tag{3}$$

and the corresponding log-likelihood of the data is

$$ln\ p(Z \mid \theta) = \sum_{i=1}^{N} ln \sum_{j=1}^{J} \pi_j\ p(z_i \mid \theta_j) \tag{4}$$

An Expectation-Maximization approach [11] is used to learn the parameters of the model. This approach introduces a set of correspondence variables C such that c_{ij} represents the binary association between points $z_i \in Z$ and components θ_j. The log-likelihood with the correspondence variables incorporated is

$$ln\ p(Z, C \mid \theta) = \sum_{i=1}^{N} \sum_{j=1}^{J} c_{ij}\ \{ln\ \pi_j\ +\ ln\ p(z_i \mid \theta_j)\}$$

The algorithm then iteratively calculates the expected value of the correspondence variables based on the current parameters of the system in the E-step and updates the parameters by maximizing the log-likelihood in the M-step. The complexity of training a GMM is $O(KNJ)$ where K is the number of iterations the algorithm takes to converge.

A. Divergence between Mixtures of Gaussians

Divergence measures seek to provide a measure of distance or dissimilarity between two pdfs. Here, we are interested in the divergence measure between two Gaussian distributions and the divergence measure between two GMMs. The closed form solution for Kullback-Leibler divergence between two Gaussian distributions $f = \mathcal{N}(\mu_f, \Sigma_f)$ and $g = \mathcal{N}(\mu_g, \Sigma_g)$ for D-dimensional data is given as

$$\begin{aligned} KL(f||g) = \frac{1}{2}(log\frac{|\Sigma_g|}{|\Sigma_f|} + trace(\Sigma_g^{-1}\Sigma_f) \\ + (\mu_f - \mu_g)^T\Sigma_g^{-1}(\mu_f - \mu_g) - D) \end{aligned} \tag{5}$$

A closed form solution for KL-divergence between two Gaussian mixtures does not exist [12]. However, an approximation has been derived in [13] and is reproduced here. For two GMMs p and q with M and K components respectively

and parameters $(\pi_m, \mu_m, \Lambda_m)$ and $(\tau_k, \nu_k, \Omega_k)$, it is given as

$$D_{KL}(q,p) = \sum_{i=1}^{M} \pi_i \min_j (KL(p_i \mid\mid q_j) + log\frac{\pi_i}{\tau_j}) \quad (6)$$

B. Fidelity Threshold and Divergence

We propose to use the divergence measure to derive an estimate of the relative expressive capability of GMMs trained on given data. This quantification of expressive capability enables us to estimate the optimal size of the mixture for the target environment. The key idea here is that even though real world data is inherently non-gaussian, there is a threshold on the size of the mixture model beyond which the fidelity of the model does not vary significantly even if more components are added. We refer to this optimal size of the mixture as the fidelity-threshold (λ_f). Figure 2 shows the result of an experiment conducted to substantiate this claim.

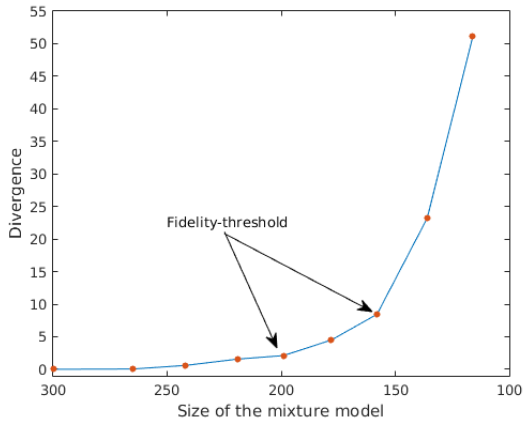


Fig. 2: Variation of KL-Divergence for GMMs of size varying from 300 to 116 with respect to the largest GMM of size 300. The possible fidelity thresholds are highlighted. Increasing the size of the GMM beyond these thresholds does not significantly affect the fidelity of representation as indicated by the small decrease in divergence.

III. METHODOLOGY

The proposed approach learns a hierarchy of Gaussian Mixture models that represent the environment at increasing levels of fidelity. An overview of the methodology to generate and update the HGMM is presented in Fig. 3. The model is divided into a local component that represents the region the system is currently observing and a global component which is the model for places the system has already been to. The input point cloud Z is tested for the novelty of information that it provides. If significantly novel, a local HGMM, $\mathcal{L}$, is instantiated after merging the current local HGMM with the global HGMM, $\mathcal{G}$. Otherwise, the current local HGMM is incrementally updated. The details of the methdology are provided in the following subsections.

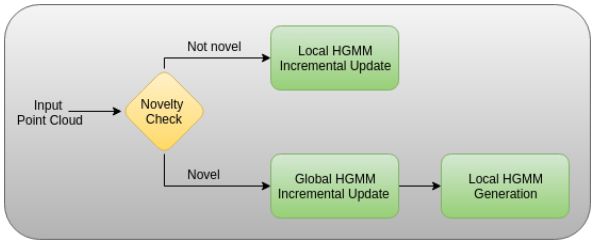


Fig. 3: Algorithmic flowchart for the proposed mapping formulation

A. Local HGMM Generation

Algorithm 1 outlines the proposed bottom-up algorithm to learn a hierarchy of GMMs over the input point-cloud. The algorithm requires a similarity threshold λ_d and the point-cloud Z as input. The lowest level is trained using EM (Line 5). The higher levels of the hierarchy are generated by merging [14] similar components where two components are said to be similar if their KL-Divergence (5) is within λ_d (Lines 12 to 15). The KL Divergence (6) of the current level with the lowest level is used to estimate the knee point and thus the fidelity-threshold (Lines 20 to 23). Once estimated, all levels of the hierarchy with size more than λ_f are pruned (Line 22). The process is then continued to build higher levels of the hierarchy (with lesser fidelity). The algorithm is terminated when the lowest desired fidelity GMM of size λ_t (user tunable parameter based on variation of divergence) has been generated (Line 25).

The input parameter λ_d regulates the rate of merging of Gaussian components to form higher layers of the hierarchy. A higher value of λ_d causes more components to get merged at each level thereby increasing the difference in fidelity between subsequent levels. The over-estimate of λ_f affects the accuracy of the model if it is not a strict over-estimate. Conversely, a very large value affects the computational complexity of the algorithm. The strategy used here involves applying a voxel-grid filter to the incoming point-cloud. The number of voxels occupied after the filtering is an over-estimate for λ_f. This technique is suitable for applications such as precise close-ranged inspection due to limited spatial extent of the input data. A higher value of λ_t leads to an earlier algorithm termination and fewer hierarchy levels. A lower value increases the depth of the hierarchy with the highest level corresponding to a reduced representational fidelity.

B. Novelty Check

Once a local HGMM has been generated, the incoming point cloud is tested for the novelty of its information content. To do this, the portion of the incoming point cloud that cannot be represented by the existing local HGMM is estimated. Inspired by the work of [15], a minimum likelihood or novelty threshold (λ_n) is defined as an empirically determined parameter. The likelihood that a point can be modelled by the existing HGMM is estimated by calculating the log likelihood using the GMM at the highest layer of the

Algorithm 1: Local HGMM Generation

Result: Local HGMM $\mathcal{L}$

λ_d, λ_t, $Z \leftarrow Input$;
$\mathcal{L} \leftarrow \{\emptyset\}$;
$\lambda_f \leftarrow$ *OverEstimate()*;
$div \leftarrow 0$, $l \leftarrow 0$;
$\mathcal{L} \leftarrow \mathcal{L} \cup$ *TrainGMM(*Z, λ_f*)*;
while *true* **do**
  $\mathcal{F} \leftarrow \{\emptyset\}$;
  for $i \leftarrow 1...|\mathcal{L}_l|$ **do**
    $\theta_i, w_i \leftarrow \mathcal{L}_{l,i}$;
    for $j \leftarrow$ *i+1...*$|\mathcal{L}_l|$ **do**
      $\theta_j, w_j \leftarrow \mathcal{L}_{l,j}$;
      if *KLDivergence(*θ_i, θ_j*)* $< \lambda_d$ **then**
        $\theta_i \leftarrow$ *Merge(*θ_i, θ_j*)*;
        $w_i \leftarrow w_i + w_j$;
      end
    end
    $\mathcal{F} \leftarrow \mathcal{F} \cup \{\theta_i, w_i\}$;
  end
  $\mathcal{L} \leftarrow \mathcal{F} \cup \mathcal{L}$;
  div $\leftarrow$ *KLDivergence(*$\mathcal{L}_l, \mathcal{L}_{|\mathcal{L}|-1}$*)*;
  if $\neg$*Pruned AND IsKneePoint(div)* **then**
    Prune($\mathcal{L}, l+2$*)*;
    $\lambda_f \leftarrow |\mathcal{L}_{l+1}|$;
  end
  if $|\mathcal{L}_l| < \lambda_t$ **then**
    break;
  end
end

Algorithm 2: Global HGMM Incremental Update

Result: Global HGMM $\mathcal{G}$

$\mathcal{G}, \mathcal{L}, N_\mathcal{G}, N_\mathcal{L} \leftarrow Input$;
$l_\mathcal{G} \leftarrow |\mathcal{G}|$;
$l_\mathcal{L} \leftarrow |\mathcal{L}|$;
while *true* **do**
  $\mathcal{G}_{l_\mathcal{G}} \leftarrow \mathcal{G}_{l_\mathcal{G}} \cup \mathcal{L}_{l_\mathcal{L}}$;
  $w_{l_\mathcal{G}} \leftarrow$ *UpdateWeight(*$N_\mathcal{G}, N_\mathcal{L}, w_{l_\mathcal{G}}$*)*;
  $w_{l_\mathcal{L}} \leftarrow$ *UpdateWeight(*$N_\mathcal{G}, N_\mathcal{L}, w_{l_\mathcal{L}}$*)*;
  $l_\mathcal{G} \leftarrow l_\mathcal{G} - 1$;
  $l_\mathcal{L} \leftarrow l_\mathcal{L} - 1$;
  if $l_\mathcal{G}$ *= 0 OR* $l_\mathcal{L}$ *= 0* **then**
    break;
  end
end

hierarchy. A point is considered to be novel if the likelihood is less than λ_n. If a significant portion of the incoming data is novel, a global HGMM update is triggered.

C. Local HGMM Incremental Update

A local HGMM is incrementally updated with the non-novel portion of the incoming data. The key insight here is that for a static environment, the value of λ_f for a particular region is not expected to vary with time. For updating the model, predictions for the non-novel portion of the data (Sect. III-B) are obtained from the lowest level of the HGMM. For each point z_i, posterior probability of membership per component θ_j is obtained as follows

$$p_{ij} = \frac{\pi_j p(z_i \mid \theta_j)}{\sum_{k=1}^{K} \pi_k p(z_i \mid \theta_k)} \tag{7}$$

A modified form of the maximization step of EM algorithm is used to update the parameters of the GMM. The standard maximization equations incrementally update the parameters of the mixture model for a point cloud of size N in the $(k+1)$-th iteration as follows

$$\mu_j^{k+1} = \frac{\sum_i^N \gamma_{ij}\, z_i}{\sum_i^N \gamma_{ij}} \tag{8}$$

$$\Sigma_j^{k+1} = \frac{\sum_i^N \gamma_{ij}\, z_i z_i^T}{\sum_i^N \gamma_{ij}} - \mu_j^{k+1}\, \mu_j^{k+1^T} \tag{9}$$

$$\pi_j^{k+1} = \sum_i^N \frac{\gamma_{ij}}{N} \tag{10}$$

where γ_{ij} is the expected value of the correspondence variable calculated in E-step of the algorithm. Let the support set of the existing local GMM be N. Then for the component $\theta_j = (\pi_j, \mu_j, \Sigma_j)$, we define

$$S_{\pi_j} = \sum_i^N \gamma_{ij} = N\pi_j$$

$$S_{\mu_j} = \sum_i^N \gamma_{ij}\, z_i = S_{\pi_j}\mu_j$$

$$S_{\Sigma_j} = \sum_i^N \gamma_{ij}\, z_i z_i^T = S_{\pi_j}(\Sigma_j + \mu_j\, \mu_j^T)$$

The updated mean, covariance and weights for the input point-cloud of size N' are then calculated as

$$S'_{\pi_j} = S_{\pi_j} + \sum_i^{N'} p_{ij} \tag{11}$$

$$\pi'_j = \frac{S'_{\pi_j}}{N + N'} \tag{12}$$

$$\mu'_j = \frac{S_{\mu_j} + \sum_i^{N'} p_{ij}\, z_i}{S'_{\pi_j}} \tag{13}$$

$$\Sigma'_j = \frac{(S_{\Sigma_j} + \sum_i^{N'} p_{ij}\, z_i z_i^T)}{S'_{\pi_j}} - \mu'_j\, \mu_j'^T \tag{14}$$

The update is then propagated to the higher levels of the hierarchy by merging similar components as presented in Algorithm 1

D. Global HGMM Incremental Update

Algorithm 2 provides an outline of the global HGMM incremental update required when the incoming data is significantly novel. This update involves merging the current

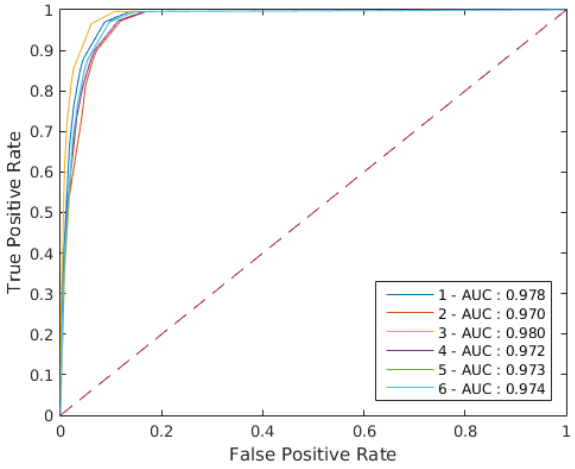


Fig. 4: ROC curves for the proposed approach with different levels of sub-sampling. The sub-sampling factor is varied from 1 to 6 for a point-cloud of size 307,200. The AUC is observed to be consistent despite increasing sub-sampling levels.

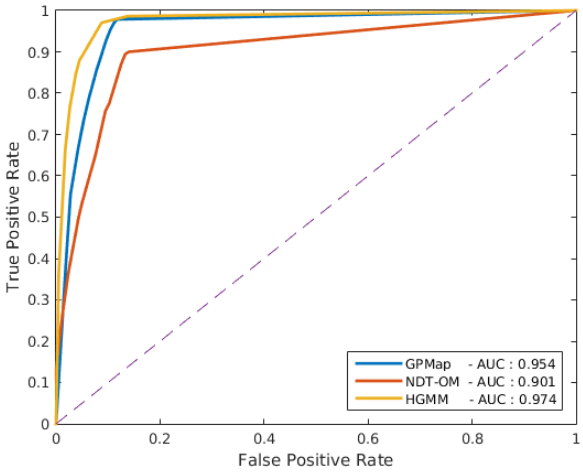


Fig. 5: ROC curve for the GPmap, NDT-OM and HGMM (proposed technique) based on real-world point-cloud data-set D1. The characteristics of the proposed approach appear favourable as compared to the noted.

local HGMM with the global HGMM. The key insight here is that the portion of the environment represented by the local HGMM cannot be modelled by the global HGMM. Thus, the update involves concatenation of corresponding levels of the two models with an adjustment of the weights. A weighted averaging scheme is adopted to scale the weights of the merged model. The updated weight $\pi_{\mathcal{G}}$ for the global GMM with a support set of size $N_{\mathcal{G}}$ is given as follows

$$\pi_{\mathcal{G}} = \frac{\pi_{\mathcal{G}} N_{\mathcal{G}}}{N_{\mathcal{G}} + N_{\mathcal{L}}} \tag{15}$$

where $N_{\mathcal{L}}$ is the support size of the local GMM $\mathcal{L}$ being merged.

IV. RESULTS

This section evaluates the capabilities of the proposed approach and provides a quantitative comparison with other state-of-the-art techniques including GP Mapping and NDT Occupancy maps. Two datasets have been used for this purpose. The first is a point-cloud dataset (D1) collected using an Asus Xtion Pro RGB-D sensor. The environment captured in the dataset exhibits varying levels of clutter with bare walls representing areas of low detail and numerous objects in the foreground representing regions of high detail. The data-set contains 186 point-clouds with odometry obtained from a motion-capture system. Figure 7a provides a cumulative view of the dataset. The second dataset (D2) is a publicly available point-cloud dataset from University of Freiburg [16]. The dataset used has 43 point-clouds and odometry at 1 Hz and captures the insides of a cluttered room (Fig. 6a).

A. Robustness to sparsity of sensor data

The robustness of the proposed approach to sparsity of input data is demonstrated in this section. For this, six instances of the proposed model are trained on a subset of the point clouds in D1 with each differing from the other in the degree of sub-sampling applied to the input data. The instances are then queried with points uniformly sampled from the input space at a fixed resolution of 1 cm. A high fidelity mesh, with an accuracy of up to 5 mm, is fitted to the input data and used as ground-truth. Figure 4 shows the Receiver Operating Characteristic (ROC) curves for each instance of the model.

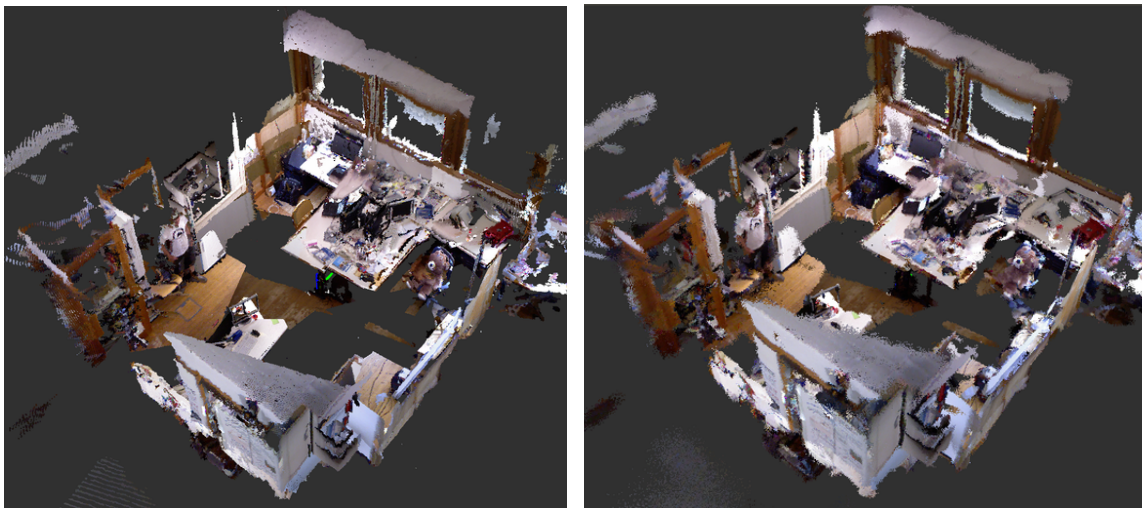
It is clear from Fig. 4 that there is no significant effect of data sparsity on the accuracy of the model. This robustness can be attributed to the fact that the model learns the distribution of points in space and thus is able to handle sparsity of input data as long as the distribution remains unchanged.

B. Comparison with GPmap and NDT-OM - Accuracy

1) Experimental Setup: An implementation of the Gaussian Process Mapping presented in [8] and the NDT-OM software[1] based on [14] are used for this comparison. The three techniques are trained on dataset D1 and queried with a set of uniformly sampled points at a resolution of 1 cm. A high fidelity polygon mesh, with an accuracy of 5 mm, is fitted to the input data and used as ground-truth. The training of hyper-parameters for GP mapping is done offline and the same set of parameters is used across all point-clouds. The size of the cell for BCM updates is 5 cm. A lazy grid size of 0.2 m was selected based on experimentation for NDT-OM for best performance. Figure 5 shows the ROC curves plotted for the three approaches.

2) Results: From Fig. 5, it can be seen that the characteristics of the proposed approach are favourable as compared

[1]NDT-OM software. `https://github.com/OrebroUniversity/perception_oru-release` [Accessed on 10th September, 2016]



(a) (b)

Fig. 6: The cumulative point cloud of the cluttered room environment from University of Freiburg dataset (a) and the reconstructed point-cloud (b) obtained by sampling from the GMM with an average size of 116 components forming the lowest level of the HGMM.

to NDT-OM and GPmap. This can be attributed to the cell-independence assumption made by NDT-OM which makes it vulnerable to sensor noise and to the high dependence of GP-mapping on its hyper-parameters for the covariance kernel and the squashing function. The hyper-parameters are trained offline on a sub-set of the points which may have led to reduced accuracy.

C. Comparison with GPmap and NDT-OM - Memory

Table I shows a comparison of the storage requirements for the three representations. For dataset D1, NDT-OM is found to learn an average of 364 Gaussians for a grid cell size of 0.2 m. GPmap needs to store the mean and covariance of all test points in memory for BCM update. For a cell size of 0.1 m, an average of 14612 points are stored along with their means and covariances. The proposed approach fits an average of 133 Gaussians per point cloud. This is less than NDT because no geometric discretization is involved which enables the proposed approach to learn wider Gaussians over planar areas (Fig. 1c). For the calculation of memory required, it is assumed that each Gaussian component is represented by 9 floats, 3 for mean and 6 for covariance.

TABLE I: Comparision of memory usage between GPmap, NDT-OM and HGMM

	NDT-OM	GPmap	HGMM
Number of components	364	14612	133
Size of a component (bytes)	36	36	36
Memory used (kB)	12.79	513.7	4.67

D. Reconstruction of point-clouds

To reconstruct a point cloud using the proposed model, a fixed number of samples are drawn from the the probability distribution represented by the model. The fraction of samples to be drawn from each component is defined by its weight in the mixture. Incremental reconstruction of point clouds is done by sampling incrementally from the current local HGMM.

Figure 6 shows the actual and reconstructed point-cloud obtained by sampling from the lowest level of the HGMM learnt incrementally over the point-clouds for dataset D2. The average size of the lowest level per point cloud is 116.

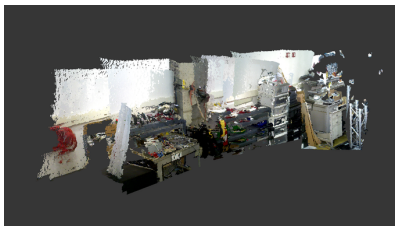
E. Metric map from continuous distribution

To generate an occupancy grid from a continuous belief distribution, samples are drawn from the distribution as described in Sect. IV-D. Sampling from each component ensures that the occupied space is covered and as no points are drawn from free space, a relatively small set of points needs to be sampled to generate the occupancy grid. Once the points have been sampled, they are binned into voxels of the desired size.

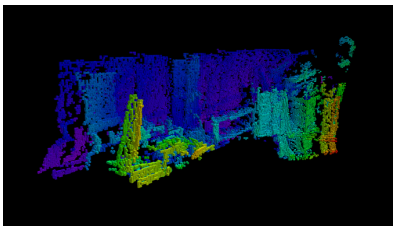
Figure 7 shows the occupancy map at 5 cm resolution generated by a grid, NDT-OM, GPmap and the proposed approach. The map for NDT-OM is generated by querying the model for log-likelihood and categorizing based on a threshold of $1e^{-5}$. The occupancy grid for GPmap is obtained by regressing from test-points defined at a resolution of 5 cm.

V. Conclusion

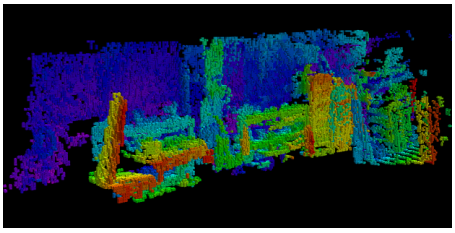
A novel approach to learn a continuous probability distribution over the environment along with a strategy to estimate the required model complexity is presented in this paper. A hierarchy of mixture models based on the representation fidelity of the model is developed. Algorithms to generate and incrementally update the hierarchy are presented. The



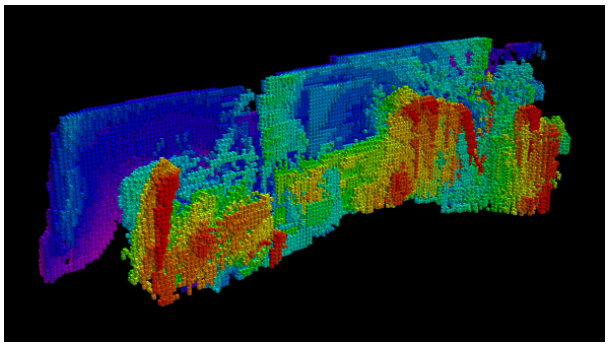
(a) Input point cloud



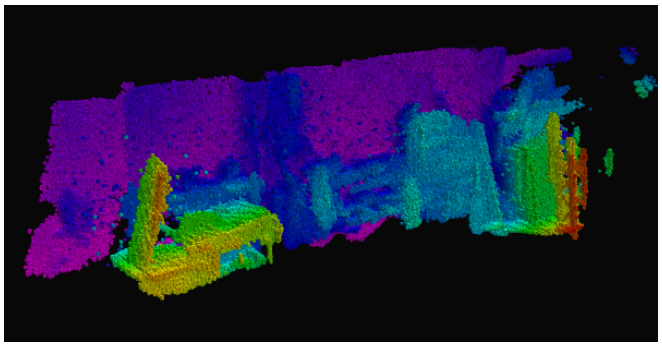
(b) Grid



(c) NDT



(d) GPmap



(e) HGMM

Fig. 7: The occupancy map at 5 cm resolution for dataset D1 generated using a naive grid (b), NDT-OM (c), GPmap (d) and HGMM (e). The gaps in the naive grid and NDT-OM approach are clearly visible. GPmap and HGMM produce dense occupancy grids due to the continuous distribution learned.

proposed approach is shown to outperform state-of-the-art techniques in memory and precision.

While we do not evaluate the real-time viability of the proposed approach, a preliminary GPU-based implementation is found to enable real-time operation in confined environments, suggesting the viability of a real-time online GPU-enabled implementation in general, cluttered environments. In the future, we will pursue parallel implementations [17] and adaptive compression-based techniques to enable real-time performance on computationally constrained platforms via low-energy and size GPU modules[2]. We also plan to incorporate an uncertainty measure to enable reasoning about unexplored and free space.

REFERENCES

[1] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[2] 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, no. 3, pp. 189–206, 2013.

[3] J. Saarinen, H. Andreasson, T. Stoyanov, J. Ala-Luhtala, and A. J. Lilienthal, "Normal distributions transform occupancy maps: Application to large-scale online 3D mapping," in *International Conference on Robotics and Automation*, pp. 2233–2238, IEEE, 2013.

[4] S. T O'Callaghan and F. T. Ramos, "Gaussian process occupancy maps," *The Intl. Journal of Robotics Research*, vol. 31–1, pp. 42–62, 2012.

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

[2]NVIDIA Jetson TX1 module. `http://www.nvidia.com/object/jetson-tx1-module.html` [Accessed on 8th July, 2016]

[6] S. Kim and J. Kim, "GPmap: A unified framework for robotic mapping based on sparse Gaussian processes," in *Field and Service Robotics*, pp. 319–332, 2015.

[7] V. Tresp, "A Bayesian committee machine," *Neural Computation*, vol. 12, no. 11, pp. 2719–2741, 2000.

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

[9] K. Doherty, J. Wang, and B. Englot, "Probabilistic Map Fusion for Fast, Incremental Occupancy Mapping with 3D Hilbert Maps," in *Int. Conf. on Robotics and Automation*, IEEE, May 2016.

[10] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz, "Accelerated Generative Models for 3D Point Cloud Data," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2016.

[11] J. A. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," *Int. Computer Science Institute*, vol. 4-510, p. 126, 1998.

[12] K. Kampa, E. Hasanbelliu, and J. C. Principe, "Closed-form Cauchy-Schwarz pdf divergence for mixture of Gaussians," in *The International Joint Conference on Neural Networks*, pp. 2578–2585, IEEE, 2011.

[13] J. Goldberger, S. Gordon, and H. Greenspan, "An efficient image similarity measure based on approximations of KL-divergence between two Gaussian mixtures," in *Proceedings of Ninth IEEE International Conference on Computer Vision*, pp. 487–493, 2003.

[14] 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 Intl. Journal of Robotics Research*, vol. 32, no. 14, pp. 1627–1644, 2013.

[15] P. M. Engel and M. R. Heinen, "Incremental learning of multivariate Gaussian mixture models," in *Brazilian Symposium on Artificial Intelligence*, pp. 82–91, Springer, 2010.

[16] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 573–580, IEEE, 2012.

[17] M. C. Altinigneli, C. Plant, and C. Böhm, "Massively parallel expectation maximization using graphics processing units," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 838–846, 2013.