Dense Incremental Metric-Semantic Mapping for Multi-Agent Systems via Sparse Gaussian Process Regression

Ehsan Zobeidi[1], Alec Koppel[2], *Member, IEEE,* and Nikolay Atanasov[1], *Member, IEEE*

***Abstract*—We develop an online probabilistic metric-semantic mapping approach for mobile robot teams relying on streaming RGB-D observations. The generated maps contain full continuous distributional information about the geometric surfaces and semantic labels (e.g., chair, table, wall). Our approach is based on online Gaussian Process (GP) training and inference, and avoids the complexity of GP classification by regressing a truncated signed distance function (TSDF) of the regions occupied by different semantic classes. Online regression is enabled through a sparse pseudo-point approximation of the GP posterior. To scale to large environments, we further consider spatial domain partitioning via a hierarchical tree structure with overlapping leaves. An extension to a multi-robot setting is developed by having each robot execute its own online measurement update and then combine its posterior parameters via local weighted geometric averaging with those of its neighbors. This yields a distributed information processing architecture in which the GP map estimates of all robots converge to a common map of the environment while relying only on local one-hop communication. Our experiments demonstrate the effectiveness of the probabilistic metric-semantic mapping technique in 2-D and 3-D environments in both single and multi-robot settings and in comparison to a deep TSDF neural network approach.**

I. Introduction

Autonomous robot systems navigating and executing complex tasks in real-world environments require an understanding of 3-D geometry and semantic context. This paper develops a probabilistic metric-semantic mapping algorithm, using streaming distance and semantic category observations (see Fig. 1), to reconstruct geometric surfaces and their semantic identity (e.g., chairs, tables, doors). To support collaboration among multiple robots, we also consider a distributed setting in which each robot observes the environment locally, with its onboard sensors, and communicates with the other robots to arrive at a common map.

We develop on a map representation which models geometric surfaces implicitly as the zero level-set of a TSDF function [1]–[3]. TSDF surface representations have gained popularity due to their high accuracy (compared to regular, adaptive, or sparse grid representations [4], [5]) and ability to directly provide distance and gradient information (compared to explicit mesh representations [6]) useful for specification of safety and visibility constraints. Classification of the geometric surfaces into semantic categories is also necessary to support context understanding and robot task specification [7]–[10].

We gratefully acknowledge support from ARL DCIST CRA W911NF-17-2-0181, NSF NRI CNS-1830399, and ONR SAI N00014-18-1-2828.

[1]Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093, USA {ezobeidi,natanasov}@ucsd.edu.

[2] Supply Chain Optimization Technologies, Amazon, Bellevue, WA 98033. Work completed while at U.S. Army Research Laboratory, Adelphi, MD 20783, USA aekoppel@amazon.com.

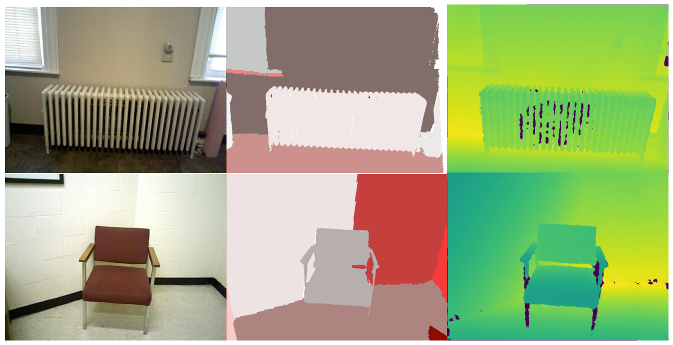


Fig. 1: RGB images (first column), segmented images (second column), and depth images (third column) used by the proposed approach for online construction of dense metric-semantic maps.

We propose a multi-class TSDF inference approach based on GP regression [11], [12]. GP inference techniques for mapping [13]–[16] have key advantages, including uncertainty quantification and resolution-free representation of the environment. Map uncertainty quantification is important for motion planning, as it captures sensing errors during collision checking, and for autonomous exploration, as the sensor motion can be planned to reduce uncertainty. In this paper specifically, uncertainty information enables us to formulate 3-D semantic segmentation as a regression problem by comparing the signed distance estimates of different classes. As discussed in more detail below, a regression formulation is conducive to more computationally efficient algorithms than classification for 3-D metric-semantic mapping. The uncertainty information is also needed to weight the map estimates of different robots appropriately when merging them across multiple robots to obtain a consistent joint map of the environment.

Our work contributes to the family of GP mapping algorithms by considering TSDF regression and multi-category classification, instead of binary occupancy mapping. Range sensors, such as LiDARs and depth cameras, do not provide direct TSDF observations because they measure distance in a specific viewing direction rather than to the nearest obstacle surface. To obtain TSDF training examples, we triangulate each depth image into a local mesh surface and measure the distance to it from a set of 3-D locations. We use semantic segmentation to divide the TSDF samples and train separate GPs for each object category. While the GPs are trained separately, all of their posteriors are taken into account when deciding the nearest surface to and the semantic categories of query points. This loosely coupled formulation is possible because segmentation errors in the sensor observations are related to physical proximity of objects of different classes in 3-D space. Hence, (discrete) semantic segmentation errors can be transferred into (continuous) distance measurement errors, allowing us to use GP regression with efficient closed-form incremental updates. In online mapping applications training needs to done incrementally and efficiently. Relying on GP regression with closed-form updates instead of GP classification with iterative approximation has a significant impact on the computational efficiency of our approach.

Efficient incremental training for GP classification [17]–[20] is more challenging because the non-Gaussian likelihood of the segmentation data is not conjugate with the GP prior and makes the integral needed for posterior normalization analytically intractable. Exciting recent developments enable scalable variational inference [18] and conjugate GP classification using latent variable augmentation [19], and offer a promising alternative to our approach.

Onboard sensors provide repeated observations of the same scene. While this redundancy is important for mitigating measurement noise, the amount of training data keeps growing over time. Hence, an important consideration is to build maps whose memory and computation requirements are determined by the underlying structure of the environment, rather than the number of observations. While GP training scales cubically with the number training examples, there are various ways to address this bottleneck [21]–[24]. We observe that, in our setting, the data can be compressed significantly through averaging before GP training and, notably, this does not affect the posterior TSDF distribution. The remaining training pairs are used as *pseudo-points* [21] to support the continuous GP representation with a finite set of parameters. To reduce the complexity in large maps further, one might consider local kriging, decomposing the spatial domain into subdomains and making predictions at a test location using only the pseudo-points contained within the subdomain. Choosing independent subdomains, however, leads to discontinuities of the predicted TSDF function at the subdomain boundaries. Ensemble methods that construct multiple local estimators and use a weighted combination of their predictions include Bayesian committee machines [25], [26], sparse probabilistic regression [27], or infinite mixtures Gaussian process experts [28]. These techniques avoid the discontinuities of local kriging but their computation cost is still significant for online training. Inspired by the adaptive occupancy representation of Octomap [4], we propose a hierarchical tree structure that decomposes the environment into overlapping subdomains, which prevents discontinuities in the GP posterior. Combining the data compression and hierarchical tree decomposition ideas allows our method to generate dense metric-semantic surfaces and, yet, remain efficient even in large environments.

Finally, we provide a distributed formulation of our TSDF GP regression, enabling multiple robots to collaboratively build a common metric-semantic map. Our distributed inference approach is inspired by probabilistic consensus techniques [29], [30]. We generalize those techniques to enable distributed function approximation instead of fixed-dimension parameter estimation. Each robot updates a local GP pseudo-point approximation and synchronizes its pseudo-point statistics with its one-hop communication neighbors. The number of pseudo-points maintained by a robot is increasing as the robot explores new regions of the environment online. We prove that the local GP estimates of each individual robot converge in *finite time* to the same GP posterior that would have been obtained by a central server using all observations obtained from all robots.

This paper improves the theoretical development for the single-robot setting in our prior work [31] and extends the approach to a decentralized multi-robot setting by introducing a novel approach for distributed incremental sparse GP regression with theoretical guarantees for consistent estimation. The main **contributions** of this work are to:

- develop an scalable incremental GP training and inference algorithm utilizing lossless data compression into a sparse set of pseudo-points (Sec. IV-B) and pseudo-point decomposition into a hierarchical tree structure (Sec. IV-D),
- achieve 3-D probabilistic metric-semantic mapping from streaming sensor data using the GP algorithm (Sec. V),
- propose a new distributed algorithm for GP regression for directed communication graphs and prove its convergence to the same posterior distribution as centralized GP regression (Sec. VI),
- enable a robot team to collaboratively build a common metric-semantic map using local observations and one-hop communication (Sec. VII, Sec. VIII, Sec. IX).

Our approach is demonstrated in simulated and real-world datasets and may be used either offline, with all sensory data provided in advance, or online, processing distance and semantic category observations incrementally as they arrive. We compare our incremental GP approach with a state-of-the-art neural network approach for TSDF reconstruction, called Implicit Geometric Regularization (IGR) [32].

II. Related Work

Various representations have been proposed for occupancy or geometric surface estimation from range or depth measurements. Occupancy grid mapping [33] discretizes the environment into a regular voxel grid and estimates the occupancy probability of each voxel independently. A dense voxel representation quickly becomes infeasible for large domains and adaptive resolution data structures, such as an octree, are necessary [4], [34]. While accurate maps may also be constructed using point cloud [35], [36] or surfel [37], [38] representations, such sparse maps do not easily support collision and visibility checking for motion and manipulation planning. Recent work is considering explicit polygonal mesh [6], [39], [40] and implicit signed distance function [41]–[44] models. We focus our review on TSDF techniques as they are most closely related to our work.

The seminal work of Curless et al. [1] emphasized the representation power of TSDF and showed that dense surface modeling can be done incrementally using range images. KinectFusion [41] achieved online TSDF mapping and RGB-D camera pose estimation by storing weighted TSDF values in a voxel grid and performing multi-scale iterative closest point (ICP) alignment between the predicted surface and the depth images. Niessner et al. [5] demonstrated that TSDF mapping can be achieved without regular or hierarchical grid data structures by hashing TSDF values only at voxels near the surfaces. These three works inspired a lot of subsequent research, allowing mapping of large environments [45], real-time operation without GPU acceleration [46], [47], map correction upon loop closure [48], [49], and semantic category inference [50]. Bylow et al. [51] propose a direct minimization of TSDF projective depth error instead of relaying on explicit data association or downsampling as in ICP. TSDF maps are accurate and collision checking in them is essentially a look-up operation, prompting their use as an alternative to occupancy grids for robot motion planning and collision checking [44], [52]. Voxblox [43] incrementally builds a (non-truncated) Euclidean signed distance field (ESDF), applying a wavefront algorithm to the hashed TSDF values. Fiesta [44] improves the ESDF construction by introducing two independent queues for inserting and deleting obstacles. Saulnier et al. [53] show that weights of the TSDF values arise as the variance of a Kalman filter and may be used as an uncertainty measure for autonomous exploration and active TSDF mapping.

Most TSDF mapping techniques, however, forgo probabilistic representations in the interest of scalability. Gaussian process (GP) inference has been used to capture correlation in binary occupancy mapping. O'Callaghan et al. [13] is among the first works to apply GP regression to infer a latent occupancy function using data from a range sensor. The GP posterior is squashed to a binary observation model a posteriori to recover occupancy likelihood. The resulting probabilistic least-squares method is more efficient than GP classification but still scales cubically with the amount of training data. To address this, several works [14], [26], [54], [55] rely on sparse kernels to perform separate GP regressions with small subsets of the training data and Bayesian Committee Machines (BCM) to fuse the separate estimates into a full probabilistic occupancy map. Ramos et al. [56], [57] proposed fast kernel approximations to project the occupancy data into a Hilbert space where a logistic regression classifier can distinguish occupied and free space. This idea has been extended to dynamic maps [58], [59] as well as into a variational autoencoder formulation [60] that compresses the local spatial information into a latent low-dimensional feature representation and then decodes it to infer the occupancy of a scene. Guo and Atanasov [61] showed that using a regular grid discretization of the latent function and a decomposable radial kernel leads to special structure of the kernel matrix (kronecker product of Toeplitz matrices) that allows linear time and memory representation of the occupancy distribution.

Augmenting occupancy representations with object and surface category information is an important extension, allowing improved situational awareness and complex mission specification for robots. Several works [8], [62]–[65] employ conditional random fields (CRFs) to capture semantic information. Vineet et al. [62] provide incremental reconstruction and semantic segmentation of outdoor environments using a hash-based voxel map and a mean-field inference algorithm for densely-connected CRFs. Grinvald et al. [50] reconstruct individual object shapes from multi-view segmented images and assemble the estimates in a voxelized TSDF map. Gan et al. [66] propose a continuous-space multi-class mapping approach, which relies on a Dirichlet class prior, a Categorical observation likelihood, and Bayesian kernel inference to extrapolate the class likelihoods to continuous space. Rosinol et al. [6] provide a modern perception library combining the state of the art in geometric and semantic understanding. Zheng et al. [67] incorporate spatial information across multiple levels of abstraction and form a probability distribution over semantic attributes and geometric representations of places using TopoNet, a deep sum-product neural network. Wang et al. [3] propose a fully convolutional neural network for semantic 3-D reconstruction that takes an octree of TSDFs fused from different camera views as input and generates a semantically labeled octree as output. IGR [32] uses a deep fully connected network with skip connections and softplus nonlinearity to map a 3D point and a latent shape vector of an object category to the signed distance from that point to the object surface. The authors observe that the norm of the gradient of an SDF function should be 1 everywhere and incorporate this in the training loss. Instead of single-object reconstruction, recent deep learning methods have focused on complete scene reconstruction using distance fields, radiance fields, and multi-view stereo [68]–[71]. These techniques have shown impressive performance but currently need all training data at once and provide only most likely estimates instead of complete posterior probability distributions. In contrast, online mapping applications require incremental update and expansion of the reconstructed scene as new data arrives. Also, as discussed in Sec. I, uncertainty quantification is needed to support safe navigation, autonomous exploration, and collaborative multi-robot mapping.

In many applications, mapping may be performed by a team of collaborating robots. Relying on centralized estimation has limitations related to the communication, computation, and storage requirements of collecting all robot measurements and map estimates at a central server. Developing distributed techniques that allow local inference and storage at each robot, communication over few-hop neighborhoods, and consensus among the robot estimates is important. Techniques extending consensus [72] to distributed probabilistic estimation [29], [73]–[75] are closely related to our work. These works show that distributed estimation of a finite-dimensional parameter is consistent when the probability density functions maintained by different nodes are averaged over one-hop neighborhoods in strongly connected di-graphs. We extend these results to distributed probabilistic estimation functions relying on local averaging of sparse (pseudo-point) GP distributions. Specific to cooperative semantic mapping, Choudhary et al. [76] develop distributed pose-graph optimization algorithms based on successive and Jacobi over-relaxation to split the computation among the robots. Koch et al. [77] develop a parallel multi-threaded implementation for cooperative 2-D SDF mapping. Lajoie et al. [78] propose a distributed SLAM approach with peer-to-peer communication that rejects spurious inter-robot loop closures using pairwise consistent measurement sets.

III. Problem Formulation

Consider a team of n robots operating in an unknown environment, represented by two disjoint sets $\mathcal{O} \subset \mathbb{R}^3$ and

$\mathcal{F} \subset \mathbb{R}^3$, comprising obstacles and free space, respectively. The obstacle region is a pairwise disjoint union, $\mathcal{O} = \cup_{l=1}^{\mathcal{C}} \mathcal{O}_l$, of $\mathcal{C}$ closed sets, each denoting the region occupied by object instances from the same semantic class. For example, $\mathcal{O}_1$ may be the space occupied by all chairs, while $\mathcal{O}_2$ may be the space occupied by all tables.

Each robot is equipped with a sensor, such as a lidar scanner or an RGB-D camera, that provides distance and class observations of the objects in its vicinity. We assume that the position $\mathbf{p}_t^i \in \mathbb{R}^3$ and orientation $\mathbf{R}_t^i \in SO(3)$ of each sensor $i \in \mathcal{V}$ at time step t are known, e.g., from a localization algorithm running onboard the robots. We model a sensor observation as a set of unit-vector rays $\{\boldsymbol{\eta}_k^i \in \mathbf{R}^3 | \|\boldsymbol{\eta}_k^i\| = 1\}$, e.g., corresponding to lidar scan rays or RGB-D image pixels. At time t, the k-th sensor ray of robot i, starts at position $\mathbf{p}_t^i$ and has direction $\mathbf{R}_t^i \boldsymbol{\eta}_k^i$. Each ray measures the distance to and semantic class of the object that it intersects with first. In practice, the class measurements are obtained from a semantic segmentation algorithm (e.g., [79]), applied to the RGB image or lidar scan (see Fig. 1), while the distance measurements are provided either as a transformation of the depth image or directly from the lidar scan.

Definition 1. A *sensor observation* of robot i at time t is a collection of distance $\lambda_{t,k}^i \in \mathbb{R}_{\geq 0}$ and object class $c_{t,k}^i \in \{1, ..., \mathcal{C}\}$ measurements acquired from position $\mathbf{p}_t^i$ along the sensor rays $\mathbf{R}_t^i \boldsymbol{\eta}_k^i$.

Given sensor poses $\mathbf{p}_t^i$, $\mathbf{R}_t^i$ and streaming observations $\lambda_{t,k}^i$, $c_{t,k}^i$ for $t = 1, 2, \ldots$, the main objective of this work is to construct a 3-D metric-semantic map of the observed environment incrementally by estimating the object class sets $\mathcal{O}_l$. We use an implicit TSDF representation of the sets $\mathcal{O}_l$.

Definition 2. The *truncated signed distance function* (TSDF) $f_l(\mathbf{x})$ of object class $\mathcal{O}_l$ is the signed distance from $\mathbf{x}$ to the boundary $\partial\mathcal{O}_l$, truncated to a maximum of $\bar{d} \geq 0$:

$$f_l(\mathbf{x}) := \begin{cases} -\min\left(d(\mathbf{x}, \partial\mathcal{O}_l), \bar{d}\right) & \text{if } \mathbf{x} \in \mathcal{O}_l \\ \min\left(d(\mathbf{x}, \partial\mathcal{O}_l), \bar{d}\right) & \text{if } \mathbf{x} \notin \mathcal{O}_l, \end{cases} \qquad (1)$$
$$d(\mathbf{x}, \partial\mathcal{O}_l) := \inf_{y \in \partial\mathcal{O}_l} \|x - y\|.$$

As we explained in Sec. V, we model the effect of noise from various sources as Gaussian noise on the TSDF values. We develop incremental sparse Gaussian Process regression to maintain distributions $\mathcal{GP}(\mu_{t,l}^i(\mathbf{x}), k_{t,l}^i(\mathbf{x}, \mathbf{x}'))$ over the TSDF functions $f_l(\mathbf{x})$ in (1) at each robot i, conditioned on the sensor observations $\left\{\lambda_{\tau,k}^i, c_{\tau,k}^i\right\}$ up to time t.

In Sec. IV, we review a sparse pseudo-point formulation of GP regression and introduce lossless compression and hierarchical decomposition of the training data to acheive incremental and scalable training. In Sec. V, we apply our general GP regression algorithm to the metric-semantic mapping problem, discussing construction of GP training data from the sensor measurements and semantic class prediction based on the TSDF function distributions of the different object classes. Next, we extend our approach from a centralized single-robot to a distributed multi-robot formulation. We develop new techniques for distributed incremental sparse GP regression

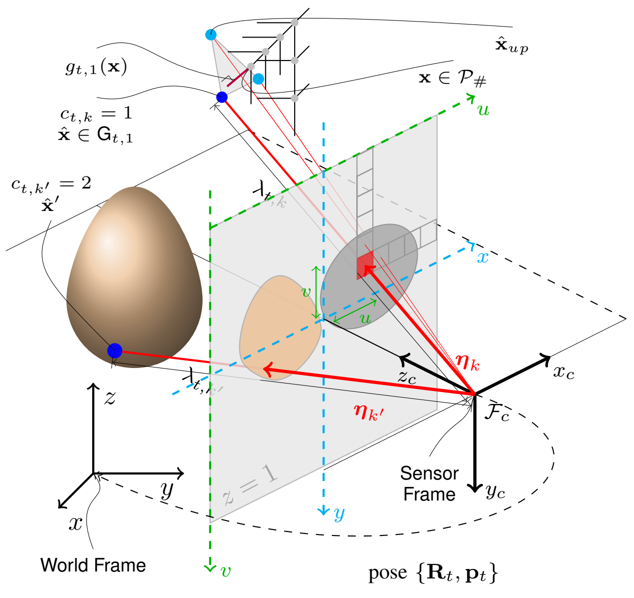


Fig. 2: Sensor observation at time t showing the distance $\lambda_{t,k}$, $\lambda_{t,k'}$ and class $c_{t,k}$, $c_{t,k'}$ measurements obtained along sensors rays $\boldsymbol{\eta}_k$, $\boldsymbol{\eta}'_k$ when a camera sensor is at position $\mathbf{p}_t$ with orientation $\mathbf{R}_t$. The pseudo-points $\mathcal{P}_\#$ (see Sec. V-A) close to the observed surface are shown in gray.

in Sec. VI and apply them to the collaborative semantic TSDF mapping problem in Sec. VII. Our method allows each robot to update its own TSDF GP model using local sensor observations and one-hop communication with its neighbors, yet guarantees theoretically that the individual GP models converge to the same posterior distribution as centralized GP regression.

IV. Data Compression and Decomposition for Incremental Sparse Gaussian Process Regression

This section reviews sparse Gaussian Process regression and introduces a new approach for compressing and decomposing training data acquired by repeated observation of the same locations. The latter is typical when an onboard robot sensor observes the same environment multiple times as discussed in Sec. V-A. When repeated observations are present, our data compression allows training a pseudo-point GP model with much fewer samples, yet provably generates the same GP posterior that would have been computed using the full uncompressed training set. The pseudo-point GP model and data compression allow us to design an efficient incremental GP algorithm that updates the posterior with sequential data instead of recomputing it from scratch. To handle large datasets, we introduce a hierarchical tree structure of pseudo-points such that separate GPs may be trained efficiently within each tree leaf.

A. Background on Sparse GP Regression

A Gaussian Process is a set of random variables such that the joint distribution of any finite subset of them is Gaussian. A GP-distributed function $f(\mathbf{x}) \sim \mathcal{GP}(\mu_0(\mathbf{x}), k_0(\mathbf{x}, \mathbf{x}'))$ is defined by a mean function $\mu_0(\mathbf{x})$ and a covariance (kernel) function $k_0(\mathbf{x}, \mathbf{x}')$. The mean and covariance are such that for any finite set $\mathcal{X} = \{\mathbf{x}_j\}_j$, the random vector $f(\mathcal{X}) := \left[f(\mathbf{x}_1), \ldots, f(\mathbf{x}_{|\mathcal{X}|})\right]^\top \in \mathbb{R}^{|\mathcal{X}|}$ has mean with j-th element $\mu_0(\mathbf{x}_j)$ and covariance matrix with (j, l)-th element $k_0(\mathbf{x}_j, \mathbf{x}_l)$ for $j, l = 1, \ldots, |\mathcal{X}|$. To avoid introducing additional symbols, we use the notation $f(\mathcal{X})$ to mean the application of the vector function $f(\mathbf{x})$ to each element of the set $\mathcal{X}$. Given a training set $\mathcal{D} = \{(\mathbf{x}_j, y_j)\}_{j=1}^{|\mathcal{X}|}$, generated according to $y_j = f(\mathbf{x}_j) + \eta_j$ with independent Gaussian noise $\eta_j \sim \mathcal{N}(0, \sigma^2)$, the posterior distribution of the random function $f(\mathbf{x})$ can be obtained from the joint distribution of the value $f(\mathbf{x})$ at an arbitrary location $\mathbf{x}$ and the random vector $\mathbf{y} := \left[y_1, \ldots, y_{|\mathcal{X}|}\right]^\top$ of measurements. In detail, the joint distribution is:

$$\begin{bmatrix} f(\mathbf{x}) \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_0(\mathbf{x}) \\ \mu_0(\mathcal{X}) \end{bmatrix}, \begin{bmatrix} k_0(\mathbf{x}, \mathbf{x}) & k_0(\mathbf{x}, \mathcal{X}) \\ k_0(\mathcal{X}, \mathbf{x}) & k_0(\mathcal{X}, \mathcal{X}) + \sigma^2 I \end{bmatrix}\right),$$

while the corresponding conditional distribution $f(\mathbf{x})|\mathcal{X}, \mathbf{y} \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ has mean and covariance functions [11]:

$$\begin{aligned} \mu(\mathbf{x}) &:= \mu_0(\mathbf{x}) + k_0(\mathbf{x}, \mathcal{X})(k_0(\mathcal{X}, \mathcal{X}) + \sigma^2 I)^{-1}(\mathbf{y} - \mu_0(\mathcal{X})), \\ k(\mathbf{x}, \mathbf{x}') &:= k_0(\mathbf{x}, \mathbf{x}') - k_0(\mathbf{x}, \mathcal{X})(k_0(\mathcal{X}, \mathcal{X}) + \sigma^2 I)^{-1} k_0(\mathcal{X}, \mathbf{x}'). \end{aligned} \tag{2}$$

Computing the GP posterior has cubic complexity in the number of observations $|\mathcal{X}|$ due to the matrix inversion in (2).

Inspired by Snelson and Ghahramani [21], we introduce a sparse approximation to the GP posterior in (2) using a set of *pseudo-points* $\mathcal{P} \subset \mathcal{D}$ whose number is $|\mathcal{P}| \ll |\mathcal{X}|$. Maintaining a GP distribution only over the pseudo-points is sufficient to obtain a high-fidelity approximation of the true GP posterior, assuming that the training data are independent samples drawn from the pseudo-point GP. The key idea is to first determine the distribution $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ of $\mathbf{f} := f(\mathcal{P})$ conditioned on $\mathcal{X}$, $\mathbf{y}$ according to (2):

$$\begin{aligned} \boldsymbol{\mu} &:= \mu_0(\mathcal{P}) + k_0(\mathcal{P}, \mathcal{X})(k_0(\mathcal{X}, \mathcal{X}) + \sigma^2 I)^{-1}(\mathbf{y} - \mu_0(\mathcal{X})) \\ &= \mu_0(\mathcal{P}) + k_0(\mathcal{P}, \mathcal{P})\left(k_0(\mathcal{P}, \mathcal{P}) + \Gamma\right)^{-1} \boldsymbol{\gamma} \qquad (3) \\ \Sigma &:= k_0(\mathcal{P}, \mathcal{P}) - k_0(\mathcal{P}, \mathcal{X})\left(k_0(\mathcal{X}, \mathcal{X}) + \sigma^2 I\right)^{-1} k_0(\mathcal{X}, \mathcal{P}), \\ &= k_0(\mathcal{P}, \mathcal{P})\left(k_0(\mathcal{P}, \mathcal{P}) + \Gamma\right)^{-1} k_0(\mathcal{P}, \mathcal{P}) \end{aligned}$$

where $\Gamma := k_0(\mathcal{P}, \mathcal{X})\left(\Lambda + \sigma^2 I\right)^{-1} k_0(\mathcal{X}, \mathcal{P})$, $\Lambda := k_0(\mathcal{X}, \mathcal{X}) - k_0(\mathcal{X}, \mathcal{P}) k_0(\mathcal{P}, \mathcal{P})^{-1} k_0(\mathcal{P}, \mathcal{X})$, and $\boldsymbol{\gamma} := k_0(\mathcal{P}, \mathcal{X})\left(\Lambda + \sigma^2 I\right)^{-1}(\mathbf{y} - \mu_0(\mathcal{X}))$. Using the definitions of information matrix $\Omega := \Sigma^{-1}$ and information mean $\boldsymbol{\omega} := \Omega \boldsymbol{\mu}$, we can equivalently write:

$$\begin{aligned} \boldsymbol{\omega} &= \Omega \mu_0(\mathcal{P}) + k_0(\mathcal{P}, \mathcal{P})^{-1} \boldsymbol{\gamma}, \\ \Omega &= k_0(\mathcal{P}, \mathcal{P})^{-1}\left(k_0(\mathcal{P}, \mathcal{P}) + \Gamma\right) k_0(\mathcal{P}, \mathcal{P})^{-1}. \end{aligned} \tag{4}$$

Then, the posterior density of $f(\mathbf{x})$ conditioned on $\mathcal{X}, \mathbf{y}$ is:

$$p(f(\mathbf{x})|\mathcal{X}, \mathbf{y}) = \int p(f(\mathbf{x})|\mathbf{f}) p(\mathbf{f}|\mathcal{X}, \mathbf{y}) d\mathbf{f} \tag{5}$$

which is a GP with mean and covariance functions:

$$\begin{aligned} \mu(\mathbf{x}) &= \mu_0(\mathbf{x}) + k_0(\mathbf{x}, \mathcal{P}) k_0(\mathcal{P}, \mathcal{P})^{-1}\left(\Omega^{-1}\boldsymbol{\omega} - \mu_0(\mathcal{P})\right) \\ k(\mathbf{x}, \mathbf{x}') &= k_0(\mathbf{x}, \mathcal{P}) k_0(\mathcal{P}, \mathcal{P})^{-1} \Omega^{-1} k_0(\mathcal{P}, \mathcal{P})^{-1} k_0(\mathcal{P}, \mathbf{x}') \\ &\quad + k_0(\mathbf{x}, \mathbf{x}') - k_0(\mathbf{x}, \mathcal{P}) k_0(\mathcal{P}, \mathcal{P})^{-1} k_0(\mathcal{P}, \mathbf{x}'). \end{aligned} \tag{6}$$

If we assume that conditioned on $\mathcal{P}$, the measurements y_j are generated independently, i.e., Λ is approximated by a diagonal matrix with elements $\lambda(\mathbf{x}_j) := k_0(\mathbf{x}_j, \mathbf{x}_j) - k_0(\mathbf{x}_j, \mathcal{P}) k_0(\mathcal{P}, \mathcal{P})^{-1} k_0(\mathcal{P}, \mathbf{x}_j)$, then the complexity of computing $\boldsymbol{\mu}$, Σ in (3) (training) and $\mu(\mathbf{x})$, $k(\mathbf{x}, \mathbf{x}')$ in (6) (testing) are $O(|\mathcal{P}|^2|\mathcal{X}| + |\mathcal{P}|^3)$ and $O(|\mathcal{P}|^2)$, respectively, instead of $O(|\mathcal{X}|^3)$ and $O(|\mathcal{X}|^2)$ without pseudo-points in (2). The use of pseudo-points leads to significant computational savings when $|\mathcal{P}| \ll |\mathcal{X}|$. To select the locations and values of the pseudo-points $\mathcal{P}$, the main approach in the literature is to perform iterative optimization to maximally approximate the training data [16], [27], [80]. However, iterative optimization is expensive in an incremental or distributed setting, which is the focus of this paper. Instead, we exploit the structure of the TSDF reconstruction problem to select pseudo-points close to the end points of the senor rays (observed surface) on a latent grid over the environment. This allows selecting pseudo-points and computing the GP posterior very efficiently. We assume that the kernel parameters are optimized offline and focus on online computation of the terms in (6), needed for prediction.

B. Repeated Input Data Compression

Next, we detail a way to obtain additional savings in terms of data storage requirements. Specifically, if the training data $\mathcal{D} = (\mathcal{X}, \mathbf{y})$ contains repeated observations from the same locations, i.e., the points in $\mathcal{X}$ are not unique, then the GP training complexity can be reduced from cubic in $|\mathcal{X}|$ to cubic in the number of distinct points in $\mathcal{X}$. We formalize this in the following proposition, which establishes that the GP posterior is unchanged if we compress the observations in $\mathbf{y}$ obtained from the same locations in $\mathcal{X}$. Repeated observations are meaningful when there is measurement noise, i.e., $\sigma > 0$. In this case the matrix $k_0(\mathcal{X}, \mathcal{X}) + \sigma^2 I$ in (2) is never singular.

Proposition 1. *Consider $f(\mathbf{x}) \sim \mathcal{GP}(\mu_0(\mathbf{x}), k_0(\mathbf{x}, \mathbf{x}'))$. Let:*

$$\begin{aligned} \mathcal{X} &= \{\mathbf{x}_1 \;, \ldots, \mathbf{x}_1 \;, \mathbf{x}_2 \;, \ldots, \mathbf{x}_2 \;, \ldots, \mathbf{x}_n \;, \ldots, \mathbf{x}_n \;\} \\ \mathbf{y} &= [y_{1,1}, \ldots, y_{1,m_1}, y_{2,1}, \ldots, y_{2,m_2}, \ldots, y_{n,1}, \ldots, y_{n,m_n}]^\top \end{aligned}$$

be data generated from the model $y_{i,j} = f(\mathbf{x}_i) + \eta_{i,j}$ with $\eta_{i,j} \sim \mathcal{N}(0, \sigma^2)$ for $i = 1, \ldots, n$ and $j = 1, \ldots, m_i$. Let:

$$\mathcal{P} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}, \; \boldsymbol{\zeta} = \left[\frac{1}{m_1}\sum_{j=1}^{m_1} y_{1,j}, \ldots, \frac{1}{m_n}\sum_{j=1}^{m_n} y_{n,j}\right]^\top \tag{7}$$

be a compressed version of the data generated from $f(\mathbf{x}_i)$ with noise $\hat{\eta}_i \sim \mathcal{N}(0, \frac{\sigma^2}{m_i})$. Then, $f(\mathbf{x})|\mathcal{X}, \mathbf{y}$ and $f(\mathbf{x})|\mathcal{P}, \boldsymbol{\zeta}$ have the same Gaussian Process distribution $\mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ with:

$$\begin{aligned} \mu(\mathbf{x}) &= \mu_0(\mathbf{x}) + k_0(\mathbf{x}, \mathcal{P}) Z(\boldsymbol{\zeta} - \mu_0(\mathcal{P})), \\ k(\mathbf{x}, \mathbf{x}') &= k_0(\mathbf{x}, \mathbf{x}') - k_0(\mathbf{x}, \mathcal{P}) Z k_0(\mathcal{P}, \mathbf{x}'), \end{aligned} \tag{8}$$

where $Z^{-1} := k_0(\mathcal{P}, \mathcal{P}) + \sigma^2 \operatorname{diag}(\mathbf{m})^{-1}$ and $\mathbf{m}$ is a vector with elements m_i.

Proof. The distribution of $f(\mathbf{x})|\mathcal{X}, \mathbf{y}$ is provided in (2). Using the data $\mathcal{P}$, $\boldsymbol{\zeta}$, instead of $\mathcal{X}$, $\mathbf{y}$, to compute the posterior GP distribution of $f(\mathbf{x})$, according to (2), leads to the expression in (8). We need to show that (2) and (8) are equal given the relationship between $\mathcal{X}$, $\mathbf{y}$ and $\mathcal{P}$, $\boldsymbol{\zeta}$ in (7). Let E be a binary matrix defined such that $k_0(\mathcal{X}, \mathbf{x}) = Ek_0(\mathcal{P}, \mathbf{x})$. Note that $k_0(\mathcal{X}, \mathcal{X}) = Ek_0(\mathcal{P}, \mathcal{P})E^\top$, $k_0(\mathbf{x}, \mathcal{X}) = k_0(\mathbf{x}, \mathcal{P})E^\top$, $E^\top E = \operatorname{diag}(\mathbf{m})$, and $\boldsymbol{\zeta} = (E^\top E)^{-1}E^\top \mathbf{y}$. Using these expressions in (2) leads to:

$$\begin{aligned} \mu(\mathbf{x}) &= \mu_0(\mathbf{x}) + \\ &\quad k_0(\mathbf{x}, \mathcal{P})E^\top (Ek_0(\mathcal{P}, \mathcal{P})E^\top + \sigma^2 I)^{-1}(\mathbf{y} - E\mu_0(\mathcal{P})), \\ k(\mathbf{x}, \mathbf{x}') &= k_0(\mathbf{x}, \mathbf{x}') - \\ &\quad k_0(\mathbf{x}, \mathcal{P})E^\top (Ek_0(\mathcal{P}, \mathcal{P})E^\top + \sigma^2 I)^{-1}Ek_0(\mathcal{P}, \mathbf{x}'). \end{aligned} \tag{9}$$

An application of the matrix inversion lemma followed by algebraic manipulation shows that $E^\top(Ek_0(\mathcal{P}, \mathcal{P})E^\top + \sigma^2 I)^{-1} = \left(k_0(\mathcal{P}, \mathcal{P}) + \sigma^2(E^\top E)^{-1}\right)^{-1}(E^\top E)^{-1}E^\top = Z(E^\top E)^{-1}E^\top$. Replacing this and $\boldsymbol{\zeta} = (E^\top E)^{-1}E^\top \mathbf{y}$ in (9) shows that the GP distributions of $f(\mathbf{x})|\mathcal{X}, \mathbf{y}$ and $f(\mathbf{x})|\mathcal{P}, \boldsymbol{\zeta}$ are equal.

□

Prop. 1 allows us to summarize a training set $\mathcal{X}$, $\mathbf{y}$ by keeping the distinct points $\mathcal{P} \subset \mathcal{X}$ as well as the average observation value $\zeta(\mathbf{p})$ and number of times $m(\mathbf{p})$ that each point $\mathbf{p} \in \mathcal{P}$ has been observed. Given these statistics, the mean function $\mu(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$ of the posterior GP can be obtained according to (8) with $\boldsymbol{\zeta} := \zeta(\mathcal{P})$ and $\mathbf{m} := m(\mathcal{P})$. The number of observations $\mathbf{m}$ determines the value of the matrix Z in (9) and correctly scales the influence of the points that are observed more frequently. When the training points $\mathcal{X}$ contain many repetitions, the subset $\mathcal{P}$ of distinct points is a natural choice of pseudo-points (Sec. IV-A). In this case, Prop. 1 shows that the GP posterior obtained from training with $\mathcal{P}$ is *exactly equal* to the GP posterior obtained from training with $\mathcal{X}$. This is illustrated in Fig. 3, where a dataset with repeated observations is used for GP regression of $\sin(x)$ with or without the compression of Prop. 1. Our result is related to a general distribution-to-distribution regression formulation in [81]. It is interesting to consider whether this connection can allow removing the assumption that the training data contains observations from the same locations, while keeping the simplicity of computing summary statistics in (7).

The raw sensor data in the mapping problem does not directly satisfy the assumption of Prop. 1. In Sec. V-A, we first construct a set of pseudo-points with TSDF values computed from the sensor data and only then apply Prop. 1 to compute the GP posterior efficiently. We exploit this compression technique for efficient incremental GP training since the same pseudo-point locations are observed multiple times.

C. Incremental Compressed Sparse GP Regression

Suppose now that, instead of a single training set $\mathcal{D}$, the data are provided sequentially, i.e., an additional dataset $\tilde{\mathcal{D}}_t$ of points $\tilde{\mathcal{X}}_t$ with labels $\tilde{\mathbf{y}}_t$ is provided at each time step t. The cumulative data up to time t are $\mathcal{D}_t := \cup_{\tau=1}^{t} \tilde{\mathcal{D}}_\tau$. Based on Prop. 1, we can define an incrementally growing set of pseudo-points $\mathcal{P}_t$ with associated number of observations $m_t(\mathbf{p})$ and average observation $\zeta_t(\mathbf{p})$ for $\mathbf{p} \in \mathcal{P}_t$ and observation precision Z_t. We show how to update these statistics when a new dataset $\tilde{\mathcal{D}}_{t+1} = (\tilde{\mathcal{X}}_{t+1}, \tilde{\mathbf{y}}_{t+1})$ arrives at time $t + 1$.

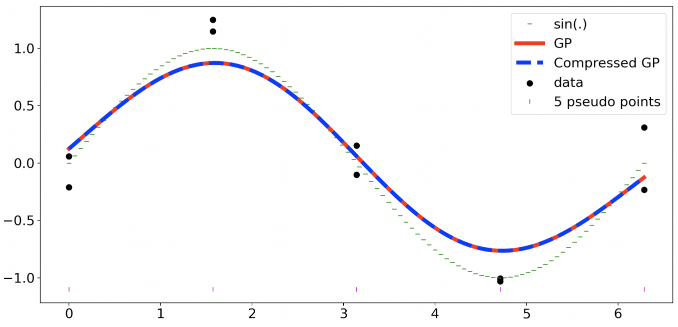


Fig. 3: Noisy data (red) obtained from a sine function (teal) at 5 equidistant pseudo-points (magenta). Two measurements are obtained at each pseudo-point. GP regression using all data is equivalent to GP regression with a compressed dataset described in Prop. 1.

Let $\tilde{\mathcal{P}}_{t+1}$ be the set of unique points in $\tilde{\mathcal{X}}_{t+1}$ with number of observations $\tilde{m}_{t+1}(\mathbf{p})$ and average observation $\tilde{\zeta}_{t+1}(\mathbf{p})$ for $\mathbf{p} \in \tilde{\mathcal{P}}_{t+1}$. The update of $\mathcal{P}_t$, $m_t(\mathbf{p})$ and $\zeta_t(\mathbf{p})$ is:

$$\begin{aligned} \mathcal{P}_{t+1} &= \mathcal{P}_t \cup \tilde{\mathcal{P}}_{t+1} \\ m_{t+1}(\mathbf{p}) &= \begin{cases} m_t(\mathbf{p}) + \tilde{m}_{t+1}(\mathbf{p}), & \text{if } \mathbf{p} \in \mathcal{P}_t, \\ \tilde{m}_{t+1}(\mathbf{p}), & \text{else,} \end{cases} \\ \zeta_{t+1}(\mathbf{p}) &= \begin{cases} \frac{m_t(\mathbf{p})\zeta_t(\mathbf{p}) + \tilde{m}_{t+1}(\mathbf{p})\tilde{\zeta}_{t+1}(\mathbf{p})}{m_{t+1}(\mathbf{p})}, & \text{if } \mathbf{p} \in \mathcal{P}_t, \\ \tilde{\zeta}_{t+1}(\mathbf{p}), & \text{else.} \end{cases} \end{aligned} \tag{10}$$

To update the observation precision Z_t, first consider the existing pseudo-points $\mathcal{P}_t$. Let l be the index of $\mathbf{p} \in \mathcal{P}_t$ in Z_t. Define $\epsilon_l := \sigma^2 \left(\frac{1}{m_{t+1}(\mathbf{p})} - \frac{1}{m_t(\mathbf{p})}\right)$, $B_0 := Z_t$, and for $l = 1, \ldots, |\mathcal{P}_t|$:

$$B_{l+1} = \left(B_l^{-1} + \epsilon_l \mathbf{e}_l \mathbf{e}_l^\top\right)^{-1} = B_l - \frac{B_l \mathbf{e}_l \mathbf{e}_l^\top B_l}{\frac{1}{\epsilon_l} + \mathbf{e}_l^\top B_l \mathbf{e}_l}. \tag{11}$$

This update is applied only to $\tilde{\mathcal{P}}_{t+1} \cap \mathcal{P}_t$ because the rest of the pseudo-points have $\epsilon_l = 0$ and, hence, $B_{l+1} = B_l$. With some abuse of notation, let $B := B_{|\mathcal{P}_t|}$ be the observation precision after all $\mathbf{p} \in \mathcal{P}_t$ have been updated. Finally, we update B by introducing the pseudo-points $\tilde{\mathcal{P}}_{t+1} \setminus \mathcal{P}_t$ that have been observed for the first time:

$$Z_{t+1} = \begin{bmatrix} B^{-1} & C \\ C^\top & D \end{bmatrix}^{-1} = \begin{bmatrix} B + BCSC^\top B & -BCS \\ -SC^\top B & S \end{bmatrix}, \tag{12}$$

where $C := k_0(\mathcal{P}_t, \tilde{\mathcal{P}}_{t+1} \setminus \mathcal{P}_t)$, $D := k_0(\tilde{\mathcal{P}}_{t+1} \setminus \mathcal{P}_t, \tilde{\mathcal{P}}_{t+1} \setminus \mathcal{P}_t) + \sigma^2 \operatorname{diag}(\tilde{m}_{t+1}(\tilde{\mathcal{P}}_{t+1} \setminus \mathcal{P}_t))^{-1}$, and $S := (D - C^\top BC)^{-1}$. The equality in (12) follows from the block matrix inversion lemma, which relates the blocks of a matrix inverse to the inverse of their Schur complement [82, Ch. 9.1]. By recursively tracking these matrix inverses, the posterior update can be executed efficiently every time a new observation arrives with complexity that is cubic in the number of new distinct points. This is a significant improvement over naïve GP training.

D. Hierarchical Tree Structure

Even after compressing the training data to a set of distinct pseudo-points using Prop. 1, the GP training complexity still scales cubically with the number of pseudo-points. In the

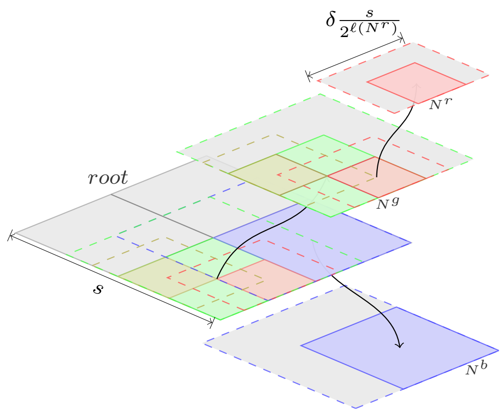


Fig. 4: Illustration of a hierarchical tree data structure, containing two pseudo-points (blue and cyan) in two dimensions. The support regions $\mathcal{S}(\cdot)$ and test regions $\mathcal{T}(\cdot)$ of three nodes N^r, N^g, N^b are shown as dashed and filled areas with red, green, and blue color, respectively. No pseudo-points are contained in the test region $\mathcal{T}(N^g)$ (filled green) of node N^g but two pseudo-points are in its support region $\mathcal{S}(N^g)$ (dashed green). In this example, the maximum number of allowable pseudo-points for each region is $max(N) = 1$, so node N^g is split into the red (N^r) and yellow (not labeled) regions. The cyan pseudo-point belongs to both $\mathcal{P}_t(N^b)$ and $\mathcal{P}_t(N^r)$.

mapping problem, the correlation between two sufficiently distant points is negligible. To ensure that online training is possible for large datasets, we develop a hierarchical tree data structure that decomposes the sample space into overlapping regions to store the pseudo-points. Our data structure is similar to an octree [4] but the regions associated with the tree nodes overlap. We train separate GPs in each region, which is efficient since the maximum number of pseudo-points per region is fixed. The resolution of the pseudo-points and the kernel parameters are fixed and are not different in different levels of the tree. The region overlap serves to eliminate discontinuities in the resulting GP estimate. The GPs associated with different regions are not statically independent because they may share some of the same pseudo-points and associated measurements. However, to enable efficient inference, we approximate the joint GP model over all pseudo-points with separately trained GP models, each using training data only from one region. At test time, the value of a query point is inferred using only the parameters of the corresponding region according to (8). The overlapping regions are illustrated in Fig. 4.

Formally, a hierarchical tree structure of pseudo-points in d dimensions is a tree data structure such that each internal node has 2^d children. Each node N is associated with a spatial region. The root is associated with a hypercube with side length $s > 0$, which is recursively subdivided into 2^d overlapping regions by the child nodes. Each node N maintains the following information:

1) $\ell(N) \geq 0$: level of N in the tree, starting from 0 at the root node.
2) $ctr(N) \in \mathbb{R}^d$: center of the region associated with N.
3) $\mathcal{S}(N) := \{\mathbf{x} \in \mathbb{R}^d \,|\, \|\mathbf{x} - ctr(N)\|_\infty \leq \delta\frac{s}{2^{\ell(N)+1}}\}$: support region of N with $\delta > 1$.
4) $\mathcal{T}(N) := \{\mathbf{x} \in \mathbb{R}^d \,|\, \|\mathbf{x} - ctr(N)\|_\infty \leq \frac{s}{2^{\ell(N)+1}}\}$: test region of N.
5) $\mathcal{P}(N) \subseteq \mathcal{S}(N) \cap \mathcal{P}_\#$: set of pseudo-points assigned to this node
6) $max(N)$: node N splits into 2^d children if the number of observed pseudo-points $\mathcal{P}(N)$ exceeds $max(N)$
7) $children(N)$: empty set if N is a leaf and, otherwise, a set of 2^d nodes at level $\ell(N)+1$ with centers in $\{ctr(N) + \mathbf{c} \mid c_i \in \{-\frac{s}{2^{\ell(N)+1}}, +\frac{s}{2^{\ell(N)+1}}\}\}$.

The pseudo-points $\mathcal{P}_t$ observed up to time t are stored in the hierarchical tree structure. The points assigned to node N at time t are $\mathcal{P}_t(N) := \mathcal{P}_t \cap \mathcal{S}(N)$. The pseudo-points $\mathcal{P}_t(N)$ of each leaf node N are used to train a separate GP. At time step t, prediction for test points in the region $\mathcal{T}(N)$ is performed by the GP associated with node N.

V. Probabilistic Metric-Semantic Mapping

In this section, we address the single-robot mapping problem using the incremental GP regression theory developed in Sec. IV. For simplicity, we suppress the robot index superscript i. We apply GP regression to estimate the TSDF $f_l(\mathbf{x})$ of each semantic class l. Since the sensor measurements $\{\lambda_{t,k}, c_{t,k}\}$ are not direct samples from the TSDFs, in Sec. V-A we transform them into training sets $\tilde{\mathcal{D}}_{t,l}$, suitable for updating the GP distributions of $f_l(\mathbf{x})$. In Sec. V-B, we apply incremental updates to GPs for each class and discuss how to predict the semantic class labels on the surfaces of the implicitly estimated object sets $\mathcal{O}_l$.

A. Training Set Construction

The class measurements allow us to associate the sensor data with particular semantic classes, while the distance measurements allow us to estimate the points where the sensor rays hit the object sets $\mathcal{O}_l$. We define the following point sets for each detected semantic class at time t:

$$\mathsf{G}_{t,l} = \{\hat{\mathbf{x}} \in \mathbb{R}^3 \,\big|\, \hat{\mathbf{x}} = \lambda_{t,k}\mathbf{R}_t\boldsymbol{\eta}_k + \mathbf{p}_t \text{ and } c_{t,k} = l\}. \quad (13)$$

The values $f_l(\hat{\mathbf{x}})$ of the TSDFs are close to zero at points $\hat{\mathbf{x}} \in \mathsf{G}_{t,l}$ because the sensor rays hit an object surface close to these locations.

As shown in Prop. 1, the complexity of online GP training can be improved by forcing the training data to repeatedly come from a finite set of points. We choose a grid discretization $\mathcal{P}_\#$ of the environment $\mathcal{O} \cup \mathcal{F}$ and construct a training set by selecting points $\mathbf{x} \in \mathcal{P}_\#$, that are at most $\epsilon > 0$ away from the points $\hat{\mathbf{x}} \in \mathsf{G}_{t,l}$, and approximating their TSDF values $f_l(\mathbf{x}) \approx g_{t,l}(\mathbf{x})$ (see Fig. 2). Precisely, the training data sets are constructed at time t as:

$$\tilde{\mathcal{D}}_{t,l} = \{(\mathbf{x}, g_{t,l}(\mathbf{x})) | \mathbf{x} \in \mathcal{P}_\#, \exists \hat{\mathbf{x}} \in \mathsf{G}_{t,l} \text{ s.t. } ||\mathbf{x} - \hat{\mathbf{x}}||_2 \leq \epsilon\}. \quad (14)$$

In the case of a camera sensor, the TSDF value $g_{t,l}(\mathbf{x})$ of a pseudo-point $\mathbf{x}$ is obtained by projecting $\mathbf{x}$ to the image plane and approximating its distance from the distance values of nearby pixels. In detail, suppose $\boldsymbol{\eta}_k$ is the unit vector

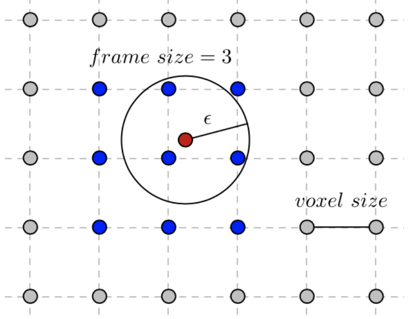


Fig. 5: Illustration of pseudo-point selection. For each sensor ray end point (red), we select pseudo-points (blue) from a regular grid $\mathcal{P}_\#$ (gray) according to (14). For efficiency, instead of choosing points within radius ϵ, we choose a rectangular region of $frame\ size = 3$ such that $(frame\ size - 1) \times voxel\ size \geq 2\epsilon$.

corresponding to the pixel closest to the projection of $\mathbf{x}$ (red pixel in Fig. 2) and let $\hat{\mathbf{x}} \in \mathsf{G}_{t,l}$ be the coordinates of its ray endpoint (blue point in Fig. 2). Let $\hat{\mathbf{x}}_{right}$ and $\hat{\mathbf{x}}_{up}$ (two cyan points in Fig. 2) be the ray endpoints of two adjacent pixels. Then, $g_{t,l}(\mathbf{x})$ is the signed distance from $\mathbf{x}$ to the plane defined by $\hat{\mathbf{x}}$, $\hat{\mathbf{x}}_{right}$, and $\hat{\mathbf{x}}_{up}$:

$$g_{t,l}(\mathbf{x}) := \mathbf{n}^\top(\mathbf{x} - \hat{\mathbf{x}}), \quad \mathbf{n} := \text{sign}(\mathbf{q}^\top(\mathbf{p}_t - \hat{\mathbf{x}}))\mathbf{q},$$
$$\mathbf{q} = \frac{(\hat{\mathbf{x}}_{right} - \hat{\mathbf{x}}) \times (\hat{\mathbf{x}}_{up} - \hat{\mathbf{x}})}{\|(\hat{\mathbf{x}}_{right} - \hat{\mathbf{x}}) \times (\hat{\mathbf{x}}_{up} - \hat{\mathbf{x}})\|}, \tag{15}$$

where $\mathbf{q}$ is the normal of the plane and the signed distance from $\mathbf{p}_t$ to the plane is positive because the sensor is known to be outside of the object set $\mathcal{O}_l$. Approximating the pseudo-point SDF values by measuring the distances to triangles formed by the nearest pixels means that the SDF values may be larger or smaller than the ground-truth SDF and using a Gaussian noise model is reasonable. The pseudo-point locations in the world frame are known since the sensor position $\mathbf{p}_t$ and orientation $\mathbf{R}_t$ are assumed known in this paper. If sensor pose uncertainty is considered, the Gaussian noise model for the measured SDF values needs to be modified to account for localization errors.

In the experiments, we discretize the workspace as a grid $\mathcal{P}_\#$ with resolution $voxel\ size$. Given a sensor ray end point $\hat{\mathbf{x}}$ in (13), instead of a sphere (circle) with radius ϵ, we choose a cubic (square) region of pseudo-points from $\mathcal{P}_\#$ around $\hat{\mathbf{x}}$. This is illustrated in Fig. 5. The cubic region is parameterized by the number of pseudo-points on its the edge, called $frame\ size$. This parameter is such that $(frame\ size - 1) \times voxel\ size \geq 2\epsilon$. We do not use ϵ but only $frame\ size$ as one of the hyperparameters of our method.

B. TSDF Mapping and Semantic Class Prediction

Given the transformed TSDF training data $\tilde{\mathcal{D}}_{t,l}$ obtained from the sensor measurement $\{\lambda_{t,k}, c_{t,k}\}_k$ at time t, we update the GP distribution of the TSDF $f_l(\mathbf{x})$ for each class l using the approach in Sec. IV-C. At time t, the new data are $\tilde{\mathcal{D}}_{t,l} = (\tilde{\mathcal{X}}_{t,l}, \tilde{\mathbf{y}}_{t,l})$, the new pseudo-points are $\tilde{\mathcal{P}}_{t,l} = \tilde{\mathcal{X}}_{t,l} \setminus \mathcal{P}_{t-1,l}$, and we update $\mathcal{P}_{t,l}$, $\zeta_{t,l}$, $m_{t,l}$ via (10). If online prediction is required, we can also update the precision matrix $Z_{t,l}$ using (11) and (12). Once we have the GPs of all classes updated, and can predict the TSDF at any query point according to (8). Next, we discuss how to predict the semantic class labels on the surfaces of the implicitly estimated object sets $\mathcal{O}_l$.

Semantic segmentation algorithms applied to camera images may produce incorrect pixel-level classification. This leads to some observations $\lambda_{t,k}$, $c_{t,k}$ being incorrectly included into the training set $\tilde{\mathcal{D}}_{t,l}$ of a different semantic class. We model the classification error as one of the sources of noise in the measured TSDF. This is motivated by the following observations. First, the GP models for different classes are trained using the TSDF values of the pseudo-points rather than the sensor ray endpoints, which semantic segmentation is based on. A pseudo-point has TSDF values for all semantic classes so it should not be considered to belong to a specific class. The association between pseudo-points and ray endpoints is based on the TSDF value so a misclassification of a ray endpoint may be interpreted as noise on the measured TSDF value. Second, the misclassification probability is not uniform across the semantic categories. Rather, it is larger when two objects of different classes are close to each other in 3-D space. In the region around the object surfaces, the pseudo-points will have very small and very similar TSDF values. To predict the correct semantic class, we compare the likelihoods of the different classes at surface points using the posterior GP distributions of the TSDFs $f_l(\mathbf{x})$.

Proposition 2. *Let $\mathcal{GP}(\mu_{t,l}(\mathbf{x}), k_{t,l}(\mathbf{x}, \mathbf{x}'))$ be the distributions of the truncated signed distance functions $f_l(\mathbf{x})$ at time t, determined according to* (8)*. Consider an arbitrary point $\mathbf{x} \in \partial\mathcal{O}$ on the surface of the obstacle set, i.e., $\mathbf{x}$ is such that $f_l(\mathbf{x}) = 0$ for some class $l \in \{1, \ldots, \mathcal{C}\}$. Then, the probability that the true class label of $\mathbf{x}$ is $c \in \{1, \ldots, \mathcal{C}\}$ is:*

$$\mathbb{P}\left(\arg\min_l |f_l(\mathbf{x})| = c \;\middle|\; \min_l |f_l(\mathbf{x})| = 0\right) = \frac{\frac{1}{\sigma_{t,c}(\mathbf{x})}\phi\left(\frac{\mu_{t,c}(\mathbf{x})}{\sigma_{t,c}(\mathbf{x})}\right)}{\sum_l \frac{1}{\sigma_{t,l}(\mathbf{x})}\phi\left(\frac{\mu_{t,l}(\mathbf{x})}{\sigma_{t,l}(\mathbf{x})}\right)},$$

where $\phi(\cdot)$ is the probability density function of the standard normal distribution and $\sigma_{t,l}(\mathbf{x}) := \sqrt{k_{t,l}(\mathbf{x}, \mathbf{x})}$.

Proof. Let $\mathbf{x}$ be an arbitrary (test) point. Denote the probability that closest surface at $\mathbf{x}$ is of class c and distance $|z|$ away by $l_c(z) := \mathbb{P}\left(\arg\min_l |f_l(\mathbf{x})| = c \text{ and } \min_l |f_l(\mathbf{x})| \leq |z|\right)$. Since $\mathbb{P}(\min_l |f_l(\mathbf{x})| \leq |z|) = \sum_l l_l(z)$, we have:

$$\mathbb{P}\left(\arg\min_l |f_l(\mathbf{x})| = c \;\middle|\; \min_l |f_l(\mathbf{x})| \leq |z|\right) = \frac{l_c(z)}{\sum_l l_l(z)}.$$

The term we are interested in computing is $\lim_{z\to 0} \frac{l_c(z)}{\sum_l l_l(z)}$. Define $\mu_l := \mu_{t,l}(\mathbf{x})$ and $\sigma_l := \sigma_{t,l}(\mathbf{x})$ for $l = 1, \ldots, \mathcal{C}$. The GP distribution of f_l stipulates that its value at $\mathbf{x}$ has a density function $p(z) = \frac{1}{\sigma_l}\phi\left(\frac{z-\mu_l}{\sigma_l}\right)$. Hence, $\mathbb{P}(|f_l(\mathbf{x})| \geq z) = 1 - \Phi\left(\frac{|z|-\mu_l}{\sigma_l}\right) + \Phi\left(\frac{-|z|-\mu_l}{\sigma_l}\right)$. Note that $l_c(z)$ corresponds to the probability that $|f_c(\mathbf{x})| \leq |f_l(\mathbf{x})|$ for all l. Since all f_l are independent of each other:

$$l_c(z) = \frac{1}{\sigma_c}\int_{-z}^{z} \phi\left(\frac{\zeta - \mu_c}{\sigma_c}\right)\prod_{l\neq c}\left(1 - \Phi\left(\frac{|\zeta| - \mu_l}{\sigma_l}\right) + \Phi\left(\frac{-|\zeta| - \mu_l}{\sigma_l}\right)\right)d\zeta$$

The claim is concluded by $\lim_{z\to 0} \frac{l_c(z)}{2z} = \frac{1}{\sigma_c}\phi\left(\frac{-\mu_c}{\sigma_c}\right)$. $\square$

The class distribution for an arbitrary point $\mathbf{x} \in \mathcal{O} \cup \mathcal{F}$, not lying on an object surface, may also be obtained, as shown in the proof of Prop. 2 but is both less efficient to compute and rarely needed in practice.

VI. Distributed Incremental Sparse GP Regression

In this section, we develop a distributed version of the incremental sparse GP regression in Sec. IV. Suppose that the n robots communicate over a network, represented as a strongly connected directed graph $G = (\mathcal{V}, \mathcal{E})$ with vertices $\mathcal{V} := \{1, ..., n\}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. An edge $(i, j) \in \mathcal{E}$ from robot i to robot j exists if the two robots can communicate. The robots directly connected to robot i are called *neighbors* and will be denoted by $\mathcal{N}e_i = \{j \in \mathcal{V} \mid (i, j) \in \mathcal{E}\}$. Let $W \in \mathbb{R}^{n \times n}$ be a weighted adjacency matrix such that $W_{i,j} > 0$ if $j \in \mathcal{N}e_i$ and $W_{i,j} = 0$, otherwise. Assume that W is a row-stochastic nonnegative and primitive matrix [72] and, hence, has a stationary distribution. The stationary distribution $\boldsymbol{\pi}$ is specified by the left eigenvector of W associated with the eigenvalue 1 and satisfies $\sum_{i=1}^{n} \pi_i = 1$. This weight matrix construction is common in consensus and distributed gradient descent algorithms [72], [83], [84]. Relying on consensus results for switching networks [72], [85], [86], our results may be generalized to time-varying graphs assuming that the graph sequence is uniformly strongly connected, i.e., there exists an integer $T > 0$ such that the union of the edges over any time interval of length T is strongly connected. However, we leave such an extension for future work.

Each robot $i \in \mathcal{V}$ receives its own local observations $\tilde{\mathcal{D}}_t^i = (\tilde{\mathcal{X}}_t^i, \tilde{\mathbf{y}}_t^i)$ at time t and extracts newly observed pseudo-points $\tilde{\mathcal{P}}_t^i$, with associated number of observations $\tilde{\mathbf{m}}_t^i$ and average values $\tilde{\boldsymbol{\zeta}}_t^i$, as detailed in Sec. IV-C. This information is used to update the complete set of pseudo-points $\mathcal{P}_t^i$ observed up to time t, along with the number of observations $\mathbf{m}_t^i$ and average values $\boldsymbol{\zeta}_t^i$, according to (10). These parameters $\Theta_t^i := \{\mathcal{P}_t^i, \mathbf{m}_t^i, \boldsymbol{\zeta}_t^i\}$, maintained by robot i, define a complete GP distribution for the function $f(\mathbf{x})$, with mean and covariance functions in (8).

While each robot can estimate $f(\mathbf{x})$ individually, we consider how the robots may exchange information to estimate $f(\mathbf{x})$ collaboratively. We observe that the continuous-space GP distribution of $f(\mathbf{x})$ is induced by the statistics $\mathbf{m}_t^i$, $\boldsymbol{\zeta}_t^i$ of the pseudo-points $\mathcal{P}_t^i$. Hence, if the robots exchange information about and agree on these finite-dimensional parameters, then the corresponding GP distributions of $f(\mathbf{x})$ at each robot will agree. Our *main innovation* is a distributed algorithm for updating the sparse GP parameters of one robot using the parameters of its one-hop neighbors' distributions. While existing results apply to fixed finite-dimensional parameter estimation, our approach considers function estimation with an infinite-dimensional GP distribution, updated via consensus over an incrementally growing set of pseudo-point parameters.

To gain intuition about the construction of consensus schemes over GP posteriors, we first review distributed Kalman filtering for fixed-dimensional parameter estimation in Sec. VI-A. Then, we use the connection between the GP posterior induced by pseudo-points and the joint Gaussian distribution over the pseudo-points described from Sec. IV-A to develop distributed GP regression in Sec. VI-B. In Sec. VI-C, we prove that the distributed algorithm converges to the same posterior GP distribution as a centralized sparse GP regression that uses the observations from all robots. Finally, in Sec. VI-D, we provide an approach to label the messages that the robots exchange in order to avoid repeated communication of the same information.

A. Distributed Kalman Filtering

Suppose that the robots aim to estimate a fixed (finite-dimensional) vector $\mathbf{f}$ cooperatively using local observations $\mathbf{y}_t^i$, generated according to a linear Gaussian model:

$$\mathbf{y}_t^i = H^i \mathbf{f} + \boldsymbol{\eta}_t^i, \qquad \boldsymbol{\eta}_t^i \sim \mathcal{N}(0, V^i). \tag{16}$$

Assume that the observations $\mathbf{y}_t^i$ received by robot i are independent over time and from the observations of all other robots. Assume also that the graph G is connected and that $\mathbf{f}$ is observable if one has access to the observations received by all robots, i.e., the matrix $\begin{bmatrix} H^1 & \cdots & H^n \end{bmatrix}$ has rank equal to the dimension of $\mathbf{f}$. Since individual observations $\mathbf{y}_t^i$ alone may be insufficient to estimate $\mathbf{f}$, the robots need to exchange information. We suppose that each robot starts with a prior probability density function $p_0^i(\mathbf{f})$ over the unknown vector $\mathbf{f}$ and updates it over time, relying on its local observations $\mathbf{y}_t^i$ as well as communication with one-hop neighbors in G.

Rahnama Rad and Tahbaz-Saleh [73] developed a consistent distributed estimation algorithm, in which each agent i uses standard Bayesian updates with its local observations $\mathbf{y}_{t+1}^i$ but, instead of its own prior p_t^i, each agent uses a weighted geometric average of its neighbors' priors:

$$p_{t+1}^i(\mathbf{f}) \propto p^i(\mathbf{y}_{t+1}^i | \mathbf{f}) \prod_{i=1}^{n} (p_t^i(\mathbf{f}))^{W_{ij}}, \tag{17}$$

where $p^i(\mathbf{y}_{t+1}^i | \mathbf{f})$ is an observation model, such as (16), that should satisfy certain regularity conditions [73]. Atanasov et al. [74] showed that if the prior distributions p_0^i are Gaussian and the observation models are linear Gaussian as in (16), the resulting distributed Kalman filter is mean-square consistent (the estimates $\arg\max_{\mathbf{f}} p_t^i(\mathbf{f})$ of all agents i converge in mean square to the true $\mathbf{f}$). Specifically, if the priors are $\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}_0^i, \Sigma_0^i)$ with information matrix $\Omega_0^i := (\Sigma_0^i)^{-1}$ and information mean $\boldsymbol{\omega}_0^i := \Omega_0^i \boldsymbol{\mu}_0^i$, the Gaussian version of the distributed estimator in (17) is:

$$\begin{aligned} \boldsymbol{\omega}_{t+1}^i &= \sum_{i=1}^{n} W_{ij} \boldsymbol{\omega}_t^j + H^{i\top} {V^i}^{-1} \mathbf{y}_{t+1}^i \\ \Omega_{t+1}^i &= \sum_{j=1}^{n} W_{ij} \Omega_t^j + H^{i\top} {V^i}^{-1} H^i \end{aligned} \tag{18}$$

because geometric averaging and Bayesian updates with Gaussian densities lead to a Gaussian posterior density [74]. The relationship between geometric means being used for belief propagation in (17) and weighted averaging via mixing matrix W forms the conceptual basis for message passing in the more general GP posterior inference setting which we detail next.

B. Distributed Incremental Sparse GP Regression

The distributed estimation algorithm in (18) does not directly apply to GP regression because the estimation target $f(\mathbf{x})$ is infinite-dimensional. However, the sparse GP regression, described in Sec. IV, relies on a finite (albeit incrementally growing) set of pseudo-points $\mathcal{P}_t$, and we show that it is possible to obtain distributed incremental sparse GP regression based on (18). As discussed earlier, each robot i maintains parameters $\Theta_t^i := \{\mathcal{P}_t^i, \mathbf{m}_t^i, \boldsymbol{\zeta}_t^i\}$ based on its local observations $\tilde{\mathcal{D}}_t^i = (\tilde{\mathcal{X}}_t^i, \tilde{\mathbf{y}}_t^i)$. Our key idea is to perform weighted geometric averaging over local posteriors, which translates to simple weighted averaging of the means and covariances in (3) of f at a finite set of points $\mathcal{Q} \supseteq \mathcal{P}_t^i$, which will be specified precisely below. The parameters Θ_t^i induce a GP distribution over f in (8), which in turn provides a Gaussian probability density function $p_t^i(\mathbf{f}) := p(\mathbf{f}|\Theta_t^i)$ over the vector $\mathbf{f} := f(\mathcal{Q})$ with mean $\mu_t^i(\mathcal{Q})$ and covariance $\Sigma_t^i(\mathcal{Q})$, obtained from (8). In order to derive decentralized updates for GPs akin to (18), we first present the iterative updates associated with the robots' local posteriors in terms of their information mean and information matrix corresponding to the mean and covariance of $p_t^i(\mathbf{f})$.

Lemma 1. *The information mean $\omega_t^i(\mathcal{Q}) := \Omega_t^i(\mathcal{Q})\boldsymbol{\mu}_t^i(\mathcal{Q})$ and information matrix $\Omega_t^i(\mathcal{Q}) := (\Sigma_t^i(\mathcal{Q}))^{-1}$ of the Gaussian probability density function $p_t^i(\mathbf{f}) := p(\mathbf{f}|\Theta_t^i)$ of $\mathbf{f} := f(\mathcal{Q})$ with parameters $\Theta_t^i := \{\mathcal{P}_t^i, \mathbf{m}_t^i, \boldsymbol{\zeta}_t^i\}$ are:*

$$\begin{aligned}\omega_t^i(\mathcal{Q}) &= k_0^i(\mathcal{Q},\mathcal{Q})^{-1}\mu_0^i(\mathcal{Q}) + \sigma^{-2}\operatorname{diag}(m_t^i(\mathcal{Q}))\zeta_t^i(\mathcal{Q})\\ \Omega_t^i(\mathcal{Q}) &= k_0^i(\mathcal{Q},\mathcal{Q})^{-1} + \sigma^{-2}\operatorname{diag}(m_t^i(\mathcal{Q})),\end{aligned} \tag{19}$$

where, similar to Sec. IV-C, $m_t^i(\mathbf{p})$ and $\zeta_t^i(\mathbf{p})$ denote the number of observations and average observation, respectively, for $\mathbf{p} \in \mathcal{P}_t^i$ and their domains have been extended to $\mathcal{Q} \supseteq \mathcal{P}_t^i$ by defining $m_t^i(\mathbf{q}) = \zeta_t^i(\mathbf{q}) = 0$ for $\mathbf{q} \in \mathcal{Q} \setminus \mathcal{P}_t^i$.

Proof. Similar to the proof of Prop. 1, let E be a binary matrix such that $k_0^i(\mathcal{P}_t^i, \mathbf{x}) = Ek_0^i(\mathcal{Q}, \mathbf{x})$, i.e., E selects the points from the superset $\mathcal{Q}$ which correspond to $\mathcal{P}_t^i$. Note that $k_0^i(\mathcal{Q},\mathcal{P}_t^i) = k_0^i(\mathcal{Q},\mathcal{Q})E^\top$, $k_0^i(\mathcal{P}_t^i,\mathcal{Q}) = Ek_0^i(\mathcal{Q},\mathcal{Q})$, and $k_0^i(\mathcal{P}_t^i,\mathcal{P}_t^i) = Ek_0^i(\mathcal{Q},\mathcal{Q})E^\top$. The expression for $\Omega_t^i(\mathcal{Q})$ follows from the matrix inversion lemma applied to the covariance matrix $\Sigma_t^i(\mathcal{Q})$ and noting that $E^\top \operatorname{diag}(\mathbf{m}_t^i)E = \operatorname{diag}(m_t^i(\mathcal{Q}))$. Then, note that:

$$\begin{aligned}&\Omega_t^i(\mathcal{Q})k_0^i(\mathcal{Q},\mathcal{P}_t^i)Z_t^i\\ &\quad= \left(I + \sigma^{-2}E^\top \operatorname{diag}(\mathbf{m}_t^i)Ek_0^i(\mathcal{Q},\mathcal{Q})\right)E^\top Z_t^i\\ &\quad= \sigma^{-2}E^\top \operatorname{diag}(\mathbf{m}_t^i)(Z_t^i)^{-1}Z_t^i = \sigma^{-2}E^\top \operatorname{diag}(\mathbf{m}_t^i).\end{aligned}$$

Thus, the information mean is:

$$\begin{aligned}\omega_t^i(\mathcal{Q}) &= \Omega_t^i(\mathcal{Q})\left(\mu_0^i(\mathcal{Q}) + k_0^i(\mathcal{Q},\mathcal{P}_t^i)Z_t^i\left(\boldsymbol{\zeta}_t^i - \mu_0^i(\mathcal{P}_t^i)\right)\right)\\ &= \Omega_t^i(\mathcal{Q})\mu_0^i(\mathcal{Q}) + \sigma^{-2}E^\top \operatorname{diag}(\mathbf{m}_t^i)\left(\boldsymbol{\zeta}_t^i - \mu_0^i(\mathcal{P}_t^i)\right)\\ &= k_0^i(\mathcal{Q},\mathcal{Q})^{-1}\mu_0^i(\mathcal{Q}) + \sigma^{-2}E^\top \operatorname{diag}(\mathbf{m}_t^i)\boldsymbol{\zeta}_t^i\\ &= k_0^i(\mathcal{Q},\mathcal{Q})^{-1}\mu_0^i(\mathcal{Q}) + \sigma^{-2}\operatorname{diag}(m_t^i(\mathcal{Q}))\zeta_t^i(\mathcal{Q}). \qquad \square\end{aligned}$$

With the expression for the parametric updates associated with the posterior inference defined by observations acquired locally at robot i only, we next detail how to augment this update with neighboring robots' information.

1) Distributed updates with a fixed pseudo-point set: To begin, suppose that the pseudo-point sets are fixed across all robots, i.e., $\mathcal{P} \equiv \mathcal{P}_t^i$, and the local observations $\tilde{\mathcal{D}}_{t+1}^i = (\tilde{\mathcal{X}}_{t+1}^i, \tilde{\mathbf{y}}_{t+1}^i)$ satisfy $\tilde{\mathcal{X}}_{t+1}^i \subseteq \mathcal{P}$ for all t, i. Then, the information means and matrices in (19) have equal dimensions across the robots. By defining $H_{t+1}^i := k_0^i(\tilde{\mathcal{X}}_{t+1}^i, \mathcal{P})k_0^i(\mathcal{P},\mathcal{P})^{-1}$, $\boldsymbol{\omega}_t^i := \omega_t^i(\mathcal{P})$, and $\Omega_t^i := \Omega_t^i(\mathcal{P})$ we can apply the update in (18) directly. The information means and matrices have a simple structure, and, similar to (10), it is sufficient to track only the number of observations $\mathbf{m}_t^i$ and the average observations $\boldsymbol{\zeta}_t^i$ over time:

$$\begin{aligned}\boldsymbol{\omega}_{t+1}^i &= \sum_{i=1}^n W_{ij}\boldsymbol{\omega}_0^j + \frac{1}{\sigma^2}\sum_{i=1}^n W_{ij}\operatorname{diag}(\mathbf{m}_t^j)\boldsymbol{\zeta}_t^j + \frac{1}{\sigma^2}\operatorname{diag}(\tilde{\mathbf{m}}_{t+1}^i)\tilde{\boldsymbol{\zeta}}_{t+1}^i\\ \Omega_{t+1}^i &= \sum_{j=1}^n W_{ij}\Omega_0^j + \frac{1}{\sigma^2}\sum_{i=1}^n W_{ij}\operatorname{diag}(\mathbf{m}_t^j) + \frac{1}{\sigma^2}\operatorname{diag}(\tilde{\mathbf{m}}_{t+1}^i),\end{aligned} \tag{20}$$

where $\tilde{\mathbf{m}}_{t+1}^i$ and $\tilde{\boldsymbol{\zeta}}_{t+1}^i$ are the number of new observations and new observation averages received by robot i of the pseudo-points $\mathcal{P}$ at time $t+1$. We consider the case with incrementally growing pseudo-point sets that are potentially different across the robots before presenting the final distributed update equations for $\mathbf{m}_t^i$ and $\boldsymbol{\zeta}_t^i$. This is the focus of the following subsection.

2) Distributed updates with dynamic pseudo-point sets: Consider the general case where each robot maintains its own pseudo-point set $\mathcal{P}_t^i$ and the observations $\tilde{\mathcal{D}}_{t+1}^i = (\tilde{\mathcal{X}}_{t+1}^i, \tilde{\mathbf{y}}_{t+1}^i)$ may introduce new pseudo-points $\tilde{\mathcal{P}}_{t+1}^i \not\subseteq \mathcal{P}_t^i$. Our key observation is that the parameters Θ_t^i induce a GP distribution over the whole function f and, hence, can be used to obtain a Gaussian distribution over a pseudo-point set that is larger than $\mathcal{P}_t^i$ according to (19):

$$\mathcal{P}_{t+1}^i = \bigcup_{j\in\mathcal{N}e_i\cup\{i\}} \mathcal{P}_t^j \cup \tilde{\mathcal{P}}_{t+1}^i. \tag{21}$$

Note that the structure of the information mean and information matrix in (19) was derived for an arbitrary pseudo-points set $\mathcal{Q}$. First, without considering the observations $\tilde{\mathcal{D}}_{t+1}^i$, we let $\mathcal{Q} = \mathcal{P}_{t+1}^i$ and calculate $\omega_t^j(\mathcal{P}_{t+1}^i)$, $\Omega_t^j(\mathcal{P}_{t+1}^i)$. Then, the distributed averaging in (18) can be performed over the information means and information matrices in (19) with $\mathcal{Q} = \mathcal{P}_{t+1}^i$ and $H_{t+1}^i := k_0^i(\tilde{\mathcal{X}}_{t+1}^i, \mathcal{P}_{t+1}^i)k_0^i(\mathcal{P}_{t+1}^i,\mathcal{P}_{t+1}^i)^{-1}$:

$$\begin{aligned}\omega_{t+1}^i(\mathcal{P}_{t+1}^i) &= \sum_{i=1}^n W_{ij}\omega_t^j(\mathcal{P}_{t+1}^i) + H_{t+1}^{i\top}(\sigma^2 I)^{-1}\tilde{\mathbf{y}}_{t+1}^i,\\ \Omega_{t+1}^i(\mathcal{P}_{t+1}^i) &= \sum_{i=1}^n W_{ij}\Omega_t^j(\mathcal{P}_{t+1}^i) + H_{t+1}^{i\top}(\sigma^2 I)^{-1}H_{t+1}^i.\end{aligned} \tag{22}$$

We may rewrite the preceding expressions in terms of the number of observations $m_{t+1}^i(\mathbf{p})$ and average observations $\zeta_{t+1}^i(\mathbf{p})$ for any $\mathbf{p} \in \mathcal{P}_{t+1}^i$, akin to (10), by following the steps in (20) for the dynamic pseudo-point case, leading to:

$$\begin{aligned}m_{t+1}^i(\mathbf{p}) &= \sum_{j\in\mathcal{N}e_i\cup\{i\}} W_{ij}m_t^j(\mathbf{p}) + \tilde{m}_{t+1}^i(\mathbf{p}),\\ \zeta_{t+1}^i(\mathbf{p}) &= \frac{\sum_{j\in\mathcal{N}e_i\cup\{i\}} W_{ij}m_t^j(\mathbf{p})\zeta_t^j(\mathbf{p}) + \tilde{m}_{t+1}^i(\mathbf{p})\tilde{\zeta}_{t+1}^i(\mathbf{p})}{m_{t+1}^i(\mathbf{p})}.\end{aligned} \tag{23}$$

With the updates for robot i in terms of its local observations and message passing with its neighbors $\mathcal{N}e_i$ specified, we shift in the following subsection to establishing its statistical properties.

C. Theoretical Guarantee for Consistent Estimation

We show that the proposed distributed incremental sparse GP regression defined by (21), (23), and (8) converges to a centralized sparse GP regression, which uses the observation data $\cup_t \cup_i \tilde{\mathcal{D}}_t^i$ from all robots. At each time step t, the centralized estimator receives data $\cup_i \tilde{\mathcal{D}}_t^i$, and, as discussed in Sec. IV-C, updates a global set of pseudo-points $\mathcal{P}_t^{ctr}$, the number of times $m_t^{ctr}(\mathbf{p})$ each pseudo-point $\mathbf{p} \in \mathcal{P}_t^{ctr}$ has been observed, and the average observation $\zeta_t^{ctr}(\mathbf{p})$ of $\mathbf{p} \in \mathcal{P}_t^{ctr}$. In order to show that the GP maintained by each robot i eventually agrees with the centralized GP, the centralized estimator should also be affected by the Perron weight matrix W. If $W = \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, the information provided by different robots is equally credible and the centralized estimator can use the combined set of observations $\cup_i \tilde{\mathcal{D}}_t^i$ directly. If, however, the left eigenvector $\boldsymbol{\pi}$ of W is not $\mathbf{1}$, then its elements π_i specify different credibility for the different robots. More precisely, the centralized estimator should treat the measurements $\tilde{\mathcal{D}}_t^i$ of robot i as if they were generated with noise variance σ^2/π_i, instead of the true noise variance σ^2. This is equivalent to scaling the number of observations $\tilde{m}_t^i$ provided by robot i by its "credibility" π_i, leading to the following update for the centralized sparse GP regression parameters:

$$\begin{aligned}
\mathcal{P}_{t+1}^{ctr} &= \cup_{i=1}^n \tilde{\mathcal{P}}_{t+1}^i \cup \mathcal{P}_t^{ctr}, \\
m_{t+1}^{ctr}(\mathbf{p}) &= m_t^{ctr}(\mathbf{p}) + \sum_{i=1}^n \pi_i \tilde{m}_{t+1}^i(\mathbf{p}), \\
\zeta_{t+1}^{ctr}(\mathbf{p}) &= \frac{m_t^{ctr}(\mathbf{p})\zeta_t^{ctr}(\mathbf{p}) + \sum_{i=1}^n \pi_i \tilde{m}_{t+1}^i(\mathbf{p})\tilde{\zeta}_{t+1}^i(\mathbf{p})}{m_{t+1}^{ctr}(\mathbf{p})},
\end{aligned} \tag{24}$$

for all $\mathbf{p} \in \mathcal{P}_{t+1}^{ctr}$. The next result shows that the individual GP distributions maintained by each robot using the distributed updates in (23) converge to the centralized GP distribution determined by the parameters above.

Proposition 3. *Let $\tilde{\mathcal{D}}_t^i = (\tilde{\mathcal{X}}_t^i, \tilde{\mathbf{y}}_t^i)$ be the data received by robot i at time t, associated with pseudo-points $\tilde{\mathcal{P}}_t^i \subset \mathcal{P}_\#$ and number of observations $\tilde{m}_t^i(\mathbf{p})$ and average observation $\tilde{\zeta}_t^i(\mathbf{p})$ for $\mathbf{p} \in \mathcal{P}_\#$. If the data streaming stops at some time $T < \infty$, then as $t \to \infty$, the distributions $\mathcal{GP}(\mu_t^i(\mathbf{x}), k_t^i(\mathbf{x}, \mathbf{x}'))$ maintained by each robot i, specified according to* (8) *with parameters $\mathcal{P}_t^i$, $m_t^i(\mathbf{p})$, $\zeta_t^i(\mathbf{p})$ in* (21) *and* (23) *converge to the distribution $\mathcal{GP}(\mu_t^{ctr}(\mathbf{x}), k_t^{ctr}(\mathbf{x}, \mathbf{x}'))$ of the centralized estimator with parameters $\mathcal{P}_t^{ctr}$, $m_t^{ctr}(\mathbf{p})$, $\zeta_t^{ctr}(\mathbf{p})$ in* (24)*, i.e., $|\mu_t^i(\mathbf{x}) - \mu_t^{ctr}(\mathbf{x})| \to 0$ and $|k_t^i(\mathbf{x}, \mathbf{x}') - k_t^{ctr}(\mathbf{x}, \mathbf{x}')| \to 0$ almost surely for all $i \in \mathcal{V}$, $\mathbf{x}, \mathbf{x}'$.*

Proof. Since the distributions $\mathcal{GP}(\mu_t^i(\mathbf{x}), k_t^i(\mathbf{x}, \mathbf{x}'))$ and $\mathcal{GP}(\mu_t^{ctr}(\mathbf{x}), k_t^{ctr}(\mathbf{x}, \mathbf{x}'))$ are completely determined by the parameters $\mathcal{P}_t^i$, $m_t^i(\mathbf{p})$, $\zeta_t^i(\mathbf{p})$ and $\mathcal{P}_t^{ctr}$, $m_t^{ctr}(\mathbf{p})$, $\zeta_t^{ctr}(\mathbf{p})$, respectively, it is sufficient to show that $|m_t^i(\mathbf{p}) - m_t^{ctr}(\mathbf{p})| \to 0$ and $|\zeta_t^i(\mathbf{p}) - \zeta_t^{ctr}(\mathbf{p})| \to 0$ for all $i \in \mathcal{V}$, $\mathbf{p} \in \mathcal{P}_\#$. Let $\mathbf{p} \in \mathcal{P}_\#$ be arbitrary and note that $m_0^i(\mathbf{p}) = m_0^{ctr}(\mathbf{p}) = 0$ and $\zeta_0^i(\mathbf{p}) = \zeta_0^{ctr}(\mathbf{p}) = 0$ since no pseudo-points have been observed initially. Expand (24) recursively to obtain $m_t^{ctr}(\mathbf{p})$ and $\zeta_t^{ctr}(\mathbf{p})$ in terms of the observation statistics:

$$\begin{aligned}
m_t^{ctr}(\mathbf{p}) &= \sum_{\tau=0}^t \sum_{i=1}^n \pi_i \tilde{m}_\tau^i(\mathbf{p}), \\
\zeta_t^{ctr}(\mathbf{p}) &= \frac{1}{m_t^{ctr}(\mathbf{p})} \sum_{\tau=0}^t \sum_{i=1}^n \pi_i \tilde{m}_\tau^i(\mathbf{p})\tilde{\zeta}_\tau^i(\mathbf{p}).
\end{aligned} \tag{25}$$

Similarly, expand (23) to obtain $m_t^i(\mathbf{p})$ and $\zeta_t^i(\mathbf{p})$ in terms of the observation statistics:

$$\begin{aligned}
m_t^i(\mathbf{p}) &= \sum_{\tau=0}^t \sum_{j=1}^n \left[W^{t-\tau}\right]_{ij} \tilde{m}_\tau^j(\mathbf{p}), \\
\zeta_t^i(\mathbf{p}) &= \frac{1}{m_t^i(\mathbf{p})} \sum_{\tau=0}^t \sum_{j=1}^n \left[W^{t-\tau}\right]_{ij} \tilde{m}_\tau^j(\mathbf{p})\tilde{\zeta}_\tau^j(\mathbf{p}),
\end{aligned} \tag{26}$$

where the weights $\left[W^{t-\tau}\right]_{ij}$ appear since the data $\tilde{m}_\tau^j(\mathbf{p})$ and $\tilde{\zeta}_\tau^j(\mathbf{p})$ propagate through the network with weight matrix W and reach robot i via all paths of length $t - \tau$. Alternatively, (26) can be viewed as the solution of the discrete-time linear time-invariant system in (23) with transition matrix $\Phi(t, \tau) = W^{t-\tau}$, $t \geq \tau$. Since the data collection stops at some finite time T, $\tilde{m}_t^i(\mathbf{p}) = \tilde{\zeta}_t^i(\mathbf{p}) = 0$ for all $t > T$, $i \in \mathcal{V}$. The convergence of (26) to (25) is concluded from the fact that $\left[W^t\right]_{ij} \to \pi_j > 0$ since W is a row-stochastic nonnegative and primitive matrix. □

Prop. 3 is a similar result to [73, Thm. 3], where it is shown that, if the weight matrix W is doubly stochastic, a distributed parameter estimator is as efficient as any centralized parameter estimator. However, Prop. 3 applies to distributed function estimation using an incrementally growing set of parameters and re-weights the observations used by the centralized estimator via the stationary distribution $\boldsymbol{\pi}$ of W to ensure convergence even when W is not doubly stochastic.

D. Echoless Distributed GP Regression

The distributed pseudo-point update we derived in (23) is not efficient for two reasons. First, convergence to the central GP estimate is guaranteed only in the limit, as $t \to \infty$ (Prop. 3). Second, every time robots exchange messages, all information they have must be sent. This is inefficient as may be seen in the proof of Prop. 3, the observations are exchanged an infinite number of times (echos in the network). To address these limitations, we label the communication messages with the list of robots that have already received them and show that convergence to the centralized estimate can, in fact, be achieved in finite time. Let $\tilde{\Theta}_t^i := \{\tilde{\mathcal{P}}_t^i, \tilde{m}_t^i(\tilde{\mathcal{P}}_t^i), \tilde{\zeta}_t^i(\tilde{\mathcal{P}}_t^i), \ell_t^i\}$ define a communication package which contains the new observations $\tilde{\mathcal{P}}_t^i$, $\tilde{m}_t^i(\tilde{\mathcal{P}}_t^i)$, $\tilde{\zeta}_t^i(\tilde{\mathcal{P}}_t^i)$ of robot i at time t as well as a list of robots ℓ_t^i that have already received this package. The list ℓ_t^i is initialized at time t with $\{i\}$. For each robot i, define also a set of packages $\mathcal{B}_{t+1}^i$ that the robot should use at time t to update its GP parameters. The set $\mathcal{B}_t^i$ from the previous time step contains old packages that robot i should transmit to its neighbors. Inspired by the similarity of (25) and (26), we propose a distributed protocol which ensures that:

- each package, which contains the processed observations of robot i at time t, visits each robot once rather than echoing in the network, relying on ℓ_t^i to keep track of visited robots,
- convergence to the centralized GP distribution is achieved in finite and minimum time by using the *stationary distribution* (left eigenvector) $\boldsymbol{\pi}$ of W as the coefficient in (26).

The distributed parameter update for robot i at time t is:

$$\begin{aligned}
\mathcal{B}_{t+1}^i &= \bigcup_{\tilde{\Theta}_\tau^j \in \mathcal{B}_t^r, r\in \mathcal{N}e_i, i\notin \ell_\tau^j} \tilde{\Theta}_\tau^j \cup \tilde{\Theta}_{t+1}^i,\\
\ell_\tau^j &= \ell_\tau^j \cup \{i\} \text{ for all } \tilde{\Theta}_\tau^j \in \mathcal{B}_{t+1}^i,\\
\mathcal{P}_{t+1}^i &= \bigcup_{\tilde{\Theta}_\tau^j \in \mathcal{B}_{t+1}^i} \mathcal{P}_\tau^j \cup \mathcal{P}_t^i,\\
m_{t+1}^i(\mathbf{p}) &= m_t^i(\mathbf{p}) + \sum_{\tilde{\Theta}_\tau^j \in \mathcal{B}_{t+1}^i} \pi_j \tilde{m}_\tau^j(\mathbf{p}),\\
\zeta_{t+1}^i(\mathbf{p}) &= \frac{m_t^i(\mathbf{p})\zeta_t^i(\mathbf{p}) + \sum_{\tilde{\Theta}_\tau^j \in \mathcal{B}_{t+1}^i} \pi_j \tilde{m}_\tau^j(\mathbf{p})\tilde{\zeta}_\tau^j(\mathbf{p})}{m_{t+1}^i(\mathbf{p})}.
\end{aligned} \tag{27}$$

We prove that this distributed update rule converges in finite time to the centralized GP distribution. Compared with (23), the distributed update in (27) is able to achieve finite-time convergence because it uses the weights $\boldsymbol{\pi}$ from the stationary distribution of W right away, instead of processing the same information an infinite number of times to determine $\boldsymbol{\pi}$. Moreover, (23) stipulates that two robots should exchange all of their information at each time step, which is very inefficient in practice. The messages in (27) allow the robots to exchange only the latest information and guarantee that each observation reaches each robot once.

Proposition 4. *Let $\tilde{\mathcal{D}}_t^i = (\tilde{\mathcal{X}}_t^i, \tilde{\mathbf{y}}_t^i)$ be the data received by robot i at time t, associated with pseudo-points $\tilde{\mathcal{P}}_t^i \subset \mathcal{P}_\#$ and number of observations $\tilde{m}_t^i(\mathbf{p})$ and average observation $\tilde{\zeta}_t^i(\mathbf{p})$ for $\mathbf{p} \in \mathcal{P}_\#$. If the data streaming stops at some time $T < \infty$, then at time $t = T + n - 1$, the distributions $\mathcal{GP}(\mu_t^i(\mathbf{x}), k_t^i(\mathbf{x}, \mathbf{x}'))$ maintained by each robot i, specified according to* (8) *with parameters in* (27) *are exactly equal to the distribution $\mathcal{GP}(\mu_t^{ctr}(\mathbf{x}), k_t^{ctr}(\mathbf{x}, \mathbf{x}'))$ of the centralized estimator with parameters in* (24)*, i.e., $\mu_t^i(\mathbf{x}) = \mu_t^{ctr}(\mathbf{x})$ and $k_t^i(\mathbf{x}, \mathbf{x}') = k_t^{ctr}(\mathbf{x}, \mathbf{x}')$ almost surely for all $i \in \mathcal{V}$, $\mathbf{x}, \mathbf{x}'$.*

Proof. As in the proof of Prop. 3, it is sufficient to show that at $t = T + n - 1$, $m_t^i(\mathbf{p}) = m_t^{ctr}(\mathbf{p})$ and $\zeta_t^i(\mathbf{p}) = \zeta_t^{ctr}(\mathbf{p})$ for all $i \in \mathcal{V}$, $\mathbf{p} \in \mathcal{P}_\#$. As before, we express $m_t^i(\mathbf{p})$ and $\zeta_t^i(\mathbf{p})$ in terms of $\tilde{m}_\tau^j(\mathbf{p})$ and $\tilde{\zeta}_\tau^j(\mathbf{p})$ for arbitrary $\mathbf{p} \in \mathcal{P}_\#$ and $\tau \le t$. The key is to decide whether package $\tilde{\Theta}_\tau^j$ is received by robot i. Since the package exchanges are happening based on the communication graph structure, the elements of $W^{t-\tau}$ determine which robots have received a package released at time τ by time t. Precisely, if $[W^{t-\tau}]_{ij} > 0$, then robot i has received package $\tilde{\Theta}_\tau^j$ by time t and otherwise, if $[W^{t-\tau}]_{ij} = 0$, it has not received it. Let $\text{sign}(x)$ denote the sign of a scalar x with $\text{sign}(0) = 0$. Expanding (27) recursively leads to:

$$\begin{aligned}
m_t^i(\mathbf{p}) &= \sum_{\tau=0}^{t}\sum_{i=1}^{n} \text{sign}([W^{t-\tau}]_{ij})\pi_j \tilde{m}_\tau^j(\mathbf{p})\\
\zeta_t^i(\mathbf{p}) &= \frac{1}{m_t^i(\mathbf{p})}\sum_{\tau=0}^{t}\sum_{j=1}^{n} \text{sign}([W^{t-\tau}]_{ij})\pi_j \tilde{m}_\tau^j(\mathbf{p})\tilde{\zeta}_\tau^j(\mathbf{p})
\end{aligned} \tag{28}$$

Since the data collection stops at some finite time T, $\tilde{m}_\tau^i(\mathbf{p}) = \tilde{\zeta}_\tau^i(\mathbf{p}) = 0$ for all $\tau > T$, $i \in \mathcal{V}$. Comparing (26) and (25), equality of $\mu_t^i(\mathbf{x})$ and $\mu_t^{ctr}(\mathbf{x})$ and $k_t^i(\mathbf{x}, \mathbf{x}')$ and $k_t^{ctr}(\mathbf{x}, \mathbf{x}')$ at $t = T + n - 1$ is concluded by the fact that $[W^{n-1}]_{ij} > 0$ because the network is connected. $\square$

Algorithm 1 Distributed Metric-Semantic Mapping

```
 1: Routine for robot $i$ at time step $\tau$:
 2: Input: sensor observation: $\{\lambda_{\tau,k}^i, c_{\tau,k}^i\}_k$
 3: Construct training set via Sec. V-A:
        $\left\{\lambda_{\tau,k}^i, c_{\tau,k}^i\right\}_k \to \left\{\tilde{\mathcal{D}}_{\tau,l}^i\right\}_l \to \left\{\tilde{\Theta}_{\tau,l}^i\right\}_l$
 4: Receive packages $\mathcal{B}_\tau^j$ from neighbors $j \in \mathcal{N}e_i$
 5: Update the pseudo-point parameters via (27)
 6: Update the GP tree data structure for each class $l$ via Sec. IV-D
 7: for each leaf $N$ of each tree do
 8:     Update precision matrix $Z_{\tau,l}^i(N)$ in Sec. V-B via (11), (12)
 9: if TSDF and class evaluation for a set of points is requested then
10:     Evaluate the class of each point via Prop. 2 in Sec. V-B
11:     Evaluate the TSDF of each point via (8)
12:     if mesh reconstruction is requested then
13:         Apply Marching Cubes [87] to the TSDF values
```

To ensure that each package is received by each robot once, we assumed that the package keeps a list of visited robots. Since a package may be received by several robots at the same time, the robots should keep a list of received packages. More precisely, each package should be labeled based on the robot that observed it and the time slot the package is observed. Then, each robot keeps the list of received packages and always removes the packages with time label earlier than $n-1$.

VII. Distributed Metric-Semantic Mapping

We apply the distributed GP regression technique developed in Sec. VI to the multi-robot metric-semantic TSDF mapping problem. Each robot i receives local distance and class observations $\left\{\lambda_{t+1,k}^i, c_{t+1,k}^i\right\}$, which are transformed using the procedure in Sec. V-A into training data sets $\tilde{\mathcal{D}}_{t+1,l}^i = \left(\tilde{\mathcal{X}}_{t+1,l}^i, \tilde{\mathbf{y}}_{t+1,l}^i\right)$ for estimating the TSDFs $\{f_l(\mathbf{x})\}$ of the different object classes. Each dataset $\tilde{\mathcal{D}}_{t+1,l}^i$ is compressed into a set of pseudo-points $\tilde{\mathcal{P}}_{t+1,l}^i$ with associated number of observations $\tilde{m}_{t+1,l}^i(\mathbf{p})$ and average observation $\tilde{\zeta}_{t+1,l}^i(\mathbf{p})$ for $\mathbf{p} \in \tilde{\mathcal{P}}_{t+1,l}^i$. Each robot maintains a separate GP $\mathcal{GP}(\mu_{t,l}^i(\mathbf{x}), k_{t,l}^i(\mathbf{x}, \mathbf{x}')$ for each class TSDF $f_l(\mathbf{x})$. In the multi-robot case, the GP distributions of robot i are updated simultaneously and separately for all classes using the new class-specific observation data $\tilde{\mathcal{P}}_{t+1,l}^i$, $\tilde{m}_{t+1,l}^i(\tilde{\mathcal{P}}_{t+1,l}^i)$, $\tilde{\zeta}_{t+1,l}^i(\tilde{\mathcal{P}}_{t+1,l}^i)$ as well as information from the neighboring robots in the form of class-specific packages $\mathcal{B}_{t+1,l}^i$ as described in (27). To make the GP models scalable to large

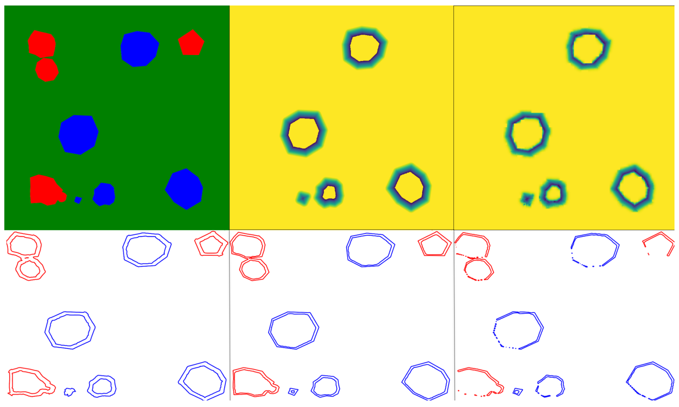


Fig. 6: Ground-truth 2-D simulated environment (top left) with two object classes (red, blue), ground-truth TSDF for the blue class (top middle), and reconstructed TSDF with $frame\ size = 10$ (top right). The reconstructed TSDF boundaries are shown for three different $frame\ size$ parameters on the bottom row: 10 (bottom left), 3 (bottom middle), 2 (bottom right). Sharp edges are captured better with $frame\ size$ 3 vs. 10 but using $frame\ size$ less that 3 caused missing parts at the boundaries.

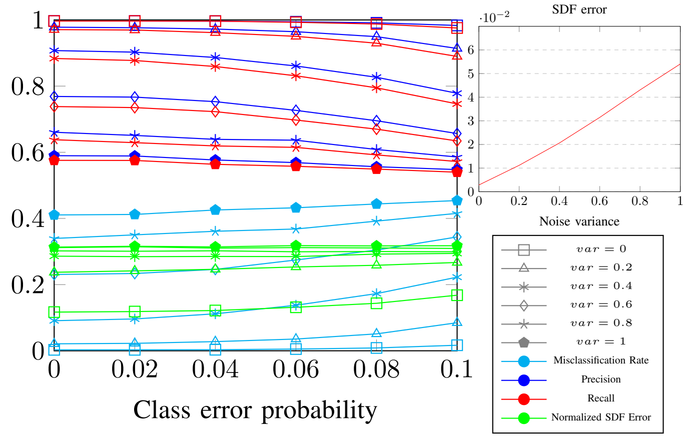


Fig. 7: Misclassification Rate, Precision, Recall, and Normalized SDF Error for different class error probability and distance noise variance. The top right plot shows the average SDF error over 10 random 40×40 maps with a 100 random observations each, with $voxel\ size = 0.1$, $max(N) = 100$, $\delta = 1.2$.

environments, we organize the pseudo-points $\mathcal{P}^i_{t,l}$ for each robot i and class l in a hierarchical tree data structure, as in Sec. IV-D, and predict the class of a query point via the method in Sec. V-B. Prop. 4 guarantees that the local TSDF GPs at each robot converge to a common GP, which is equivalent to the one that would be obtained by centralized sparse GP regression. Moreover, when the streaming of new observations stops, the convergence happens in finite time as soon as each observation is received by each robot exactly once. There is no unnecessary communication in the network. The algorithm is summarized in Alg. 1.

VIII. Evaluation using 2-D Simulated Data

In this section, we evaluate our semantic TSDF mapping approach in 2-D simulated environments. We first demonstrate the qualitative and quantitative performance of the single-robot approach of Sec. V. Then, we report results for the multi-robot approach of Sec. VII using three robots to map the same environment collaboratively. In all experiments, we use an isotropic sparse Matérn kernel ($\nu = 3/2$) [26]. Since a TSDF value of zero indicates an object surface, while an unknown environment is predominantly empty, we use a constant GP prior equal to the truncation value, $\mu^i_{0,l}(\mathbf{x}) = \bar{d} > 0$.

A. *Single-Robot 2-D Evaluation*

We generate random 2-D environments (see Fig. 6) and robot trajectories by sampling poses sequentially and keeping the ones that are in free space. Observations are obtained along the robot trajectories using a simulated distance-class sensor. We apply our incremental sparse GP regression method to obtain a probabilistic TSDF map and compare it with the ground truth TSDF.

1) TSDF Accuracy: A sample environment from our 2-D simulation with ground-truth and reconstructed TSDF and boundaries is shown in Fig. 6. Our method provides continuous probabilistic TSDF estimates. The choice of $frame\ size$ is dependent on the desired truncation value for the SDF reconstruction. Larger $frame\ size$ allows estimating larger truncation values but incurs additional computation cost. The precision and resilience to measurement noise of our method are evaluated in Fig. 7. The test points are chosen from a grid with resolution $0.5 \times voxel\ size$ within the truncation distance from the ground-truth object boundaries.

2) Classification Accuracy: We evaluate the average precision and recall of our posterior classification over 50 random 2-D maps. In each map, we pick uniformly distributed random points along the obstacle boundaries, and calculate the SDF error and the class-detection accuracy. Since the values are symmetric for binary classification, we present the average precision and recall over the two classes in Fig. 7. The figure shows that the misclassification rate, precision, recall, and SDF error are not very sensitive to class error probability. The misclassification rate is the ratio of all to the misclassified test points. The SDF error is the average absolute value difference between the estimated and ground-truth SDF values. We report normalized SDF error: $\frac{\text{SDF error}}{voxel\ size}$. Fig. 8 investigates the effect of the parameters of our algorithm on misclassification rate, normalized SDF error, False Discovery Rate (FDR $:= 1 -$ Precision), and False Negative Rate (FNR $:= 1 -$ Recall). We see that the misclassification rate, FNR, and FDR respond similarly to parameter variations.

Increasing the maximum number of pseudo-points per support region, $max(N)$, in the hierarchical tree structure improves the (normalized) SDF error. The improvement is significant at first but after a certain support region size, even exponential increases in $max(N)$ do not significantly affect the SDF error. The geometric prediction improvement caused by an initial increase in $max(N)$, improves the classification measures at first. The classification noise is iid and, hence, the training data have a Bernoulli misclassification probability and are roughly uniformly distributed in space. Further increase of $max(N)$ results in larger pseudo-point region being used for GP training. Since the misclassified samples are more and not concentrated, the classification measures deteriorate slightly

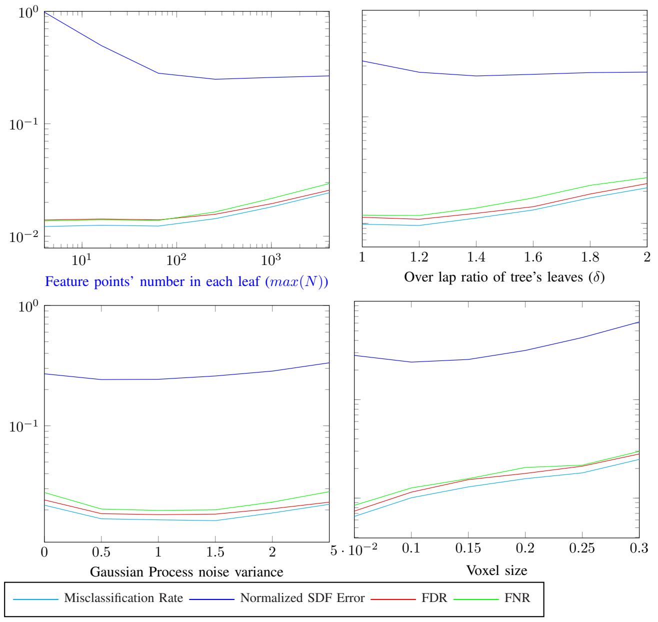


Fig. 8: Misclassification rate, normalized SDF error, False Discovery Rate (FDR), and False Negative Rate (FNR) as a function of the number of pseudo-points per tree support region ($max(N)$), support region overlap ratio (δ), GP noise variance σ^2, and workspace discretization ($voxel\ size$). The default parameter values are $\delta = 1.5$, $max(N) = 100$, $\sigma^2 = 1$, $voxel\ size = 0.1$. Class and distance measurements with class error probability of 0.05 and distance noise variance 0.5 are obtained from 100 random observations in each of 50 random 2-D maps. Test points are selected within a threshold of 0.05 from the ground truth class boundaries.

with the increase of $max(N)$. Nevertheless, our approach remains accurate in the classification errors are low in all cases. Increasing δ has a similar effect on all performance measures. Increasing the GP noise variance σ^2 improves all the measures at first but then worsens them. A correct choice of σ^2 is critical to the method but affects the misclassification rate smoothly so as long as the value of σ^2 is in the right ballpark, choosing the optimal σ^2 is not critical.

B. Multi-Robot 2-D Evaluation

Next, we evaluate the distributed GP regression in a three-robot simulation and investigate the convergence of the local GP estimates of each robot to a centralized GP estimate. We use the same random polygonal 2-D environments with two object classes but this time generate trajectories for three different robots (see Fig. 9). The robots communicate with each other over a graph with a fixed weight matrix:

$$W = \begin{bmatrix} 0.5 & 0.25 & 0.25 \\ 0.25 & 0.75 & 0 \\ 0.25 & 0 & 0.75 \end{bmatrix}. \tag{29}$$

The GP regression parameters at each robot are the same as the defaults in Sec. VIII-A. To verify Prop. 4 empirically, we compare the mean absolute error (MAE) between the GP prediction of an individual robot i and the centralized estimator ctr using all observations as described in Sec. VI-C. Specifically, at each t, we consider all classes l and associated

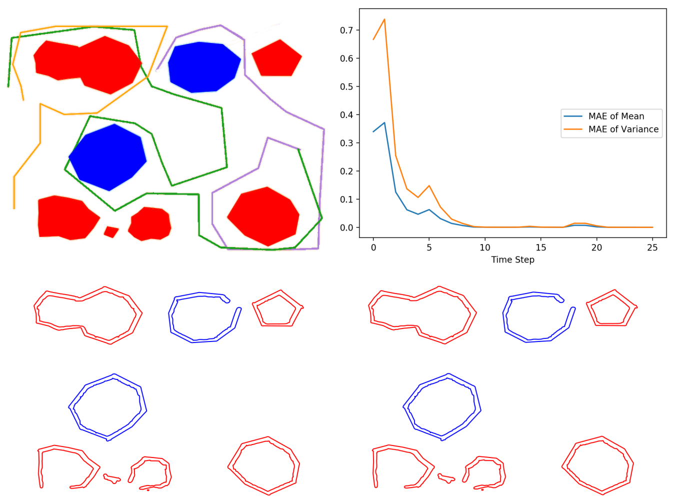


Fig. 9: Three robot trajectories (green, orange, purple) in a 2-D simulated environment (top left) with two object classes (red, blue). The zero level-sets of the TSDF reconstructions for the two classes by centralized GP regression (bottom left) and distributed GP regression from the perspective of the orange robot (bottom right) are shown. As expected, due to Prop. 2, the centralized and individual robot reconstructions are identical. This is verified quantitatively in the GP mean and variance mean absolute error (MAE) plot (top right). The initial GP parameters for each robot and object class were $\mu^i_{0,l}(\mathbf{x}) = 0.5, k^i_{0,l}(\mathbf{x}, \mathbf{x}) = 1$.

pseudo-points $\mathcal{P}^{ctr}_{t,l}$ that have been observed by the centralized estimator and calculate the mean MAE as:

$$MAE_t = \frac{1}{L_t |\mathcal{P}^{ctr}_{t,l}|} \sum_{\ell} \sum_{\mathbf{p} \in \mathcal{P}^{ctr}_{t,l}} \left| \mu^i_{t,l}(\mathbf{p}) - \mu^{ctr}_{t,l}(\mathbf{p}) \right|, \tag{30}$$

where L_t is the number of observed object classes by time t. The variance MAE is computed equivalently to (30) with $\mu^i_{t,l}(\mathbf{p})$ and $\mu^{ctr}_{t,l}(\mathbf{p})$ replaced by $k^i_{t,l}(\mathbf{p}, \mathbf{p})$ and $k^{ctr}_{t,l}(\mathbf{p}, \mathbf{p})$.

Fig. 9 shows the final reconstructions of one robot and the centralized estimator. As expected, the final reconstructions are identical and convergence happens in finite time. The behavior of the mean and variance MAE curves is similar. This is expected because the distance between the local and centralized GP parameters is due to unobserved information rather than stochastic noise. We see that the MAE curves approach 0 quickly. Several peaks are observed in the curves when new sections of the environment that are not visible to robot i are observed by another robot in the network. The new information disseminates in the network and the MAE curves approach zero again.

IX. Evaluation using 3-D Real Data

In this section, we evaluate our approach using real RGB-D data from physical 3-D environments. As before, we use an isotropic sparse Matérn kernel ($v = 3/2$) and a grid of potential pseudo-points $\mathcal{P}_\#$ with resolution $voxel\ size$. Given a query point $\hat{\mathbf{x}}$, we choose a cubic region around it such that $(frame\ size - 1) \times voxel\ size \geq 2 \times \epsilon$ to construct the training data in (13). All points from $\mathcal{P}_\#$ that lie in the cubic region are chosen as pseudo-points associated with $\hat{\mathbf{x}}$.

A. Single-Robot 3-D Evaluation

We compare our method to the incremental Euclidean signed distance mapping method Fiesta [44] on the Cow and Lady dataset [43]. We also demonstrate 3-D semantic reconstruction on the SceneNN dataset [88].

1) Cow and Lady Dataset: The reconstruction of the Cow and Lady dataset with 829 depth images and known camera trajectory by the single-robot TSDF GP regression of Sec. V is shown in Fig. 13. A triangular mesh is extracted from the mean TSDF prediction using the Marching Cubes algorithm [89]. The reconstruction time and error with respect to the ground-truth scene point cloud are reported in Fig. 10. The error of Fiesta with default parameters is shown as well. Similar to the 2-D simulations, increasing the maximum number of pseudo-points $max(N)$ per tree support region improves the SDF error of our approach. The improvement is significant at first and less pronounced afterwards. Conversely, the computation time decreases at first because the number of leaves in the hierarchical tree structure decreases and then increases afterwards as the GP covariance matrices get larger. Increasing δ leads to an insignificant improvement in the SDF error at the expense of a significant reconstruction time increase. Increasing the GP noise variance improves the SDF error at first (especially when the error is close to zero) but worsens it afterwards without significant impact on time. As $voxel\ size$ varies, our method outperforms Fiesta noticeably.

2) SceneNN Dataset: We evaluate the classification accuracy of our method on the SceneNN dataset in Fig. 11. The GP posterior is evaluated on a test grid with resolution $0.5 \times voxel\ size$. The test points with posterior variance less than a threshold are used to reconstruct a triangular mesh via Marching Cubes [89]. We use Prop. 2 for classification. The effect of the different parameters is illustrated in Fig. 11. Increasing $max(N)$ improves both classification and TSDF reconstruction results. The improvement after $max(N) = 100$ is negligible but time increases significantly. Increasing δ improves the TSDF reconstruction significantly at first. After $\delta = 1.4$, the improvement is negligible. As seen in the 2-D simulations, choosing a correct magnitude for the GP noise variance σ^2 is very important for both the classification and TSDF reconstruction but choosing the optimal value for σ^2 is not critical. Our method provides continuous TSDF estimates but this does not mean that the zero-level set of the estimated TSDF is necessarily continuous. A positive TSDF prediction by the GP means that the corresponding location is empty. Choosing the truncation value as the prior GP mean results in assuming that space is empty by default. The presence of pseudo-points, generated around observed surfaces, is what introduces occupied space in the GP estimation. For example, the grid-like empty spots in Fig. 11, resulting when the parameter choices are suboptimal, are due to the choice of pseudo-points on a regular grid.

B. Parameter Selection

This section discusses parameter selection for our algorithm, based on the results in Sec. IX-A. With parameter selection

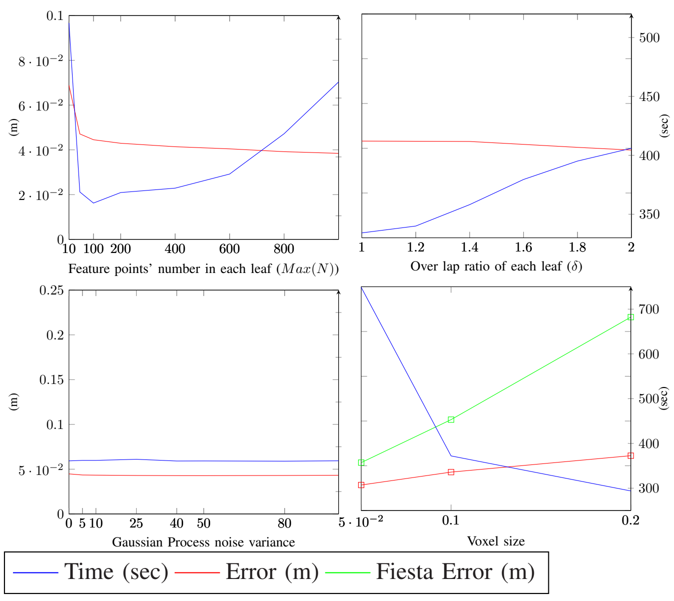


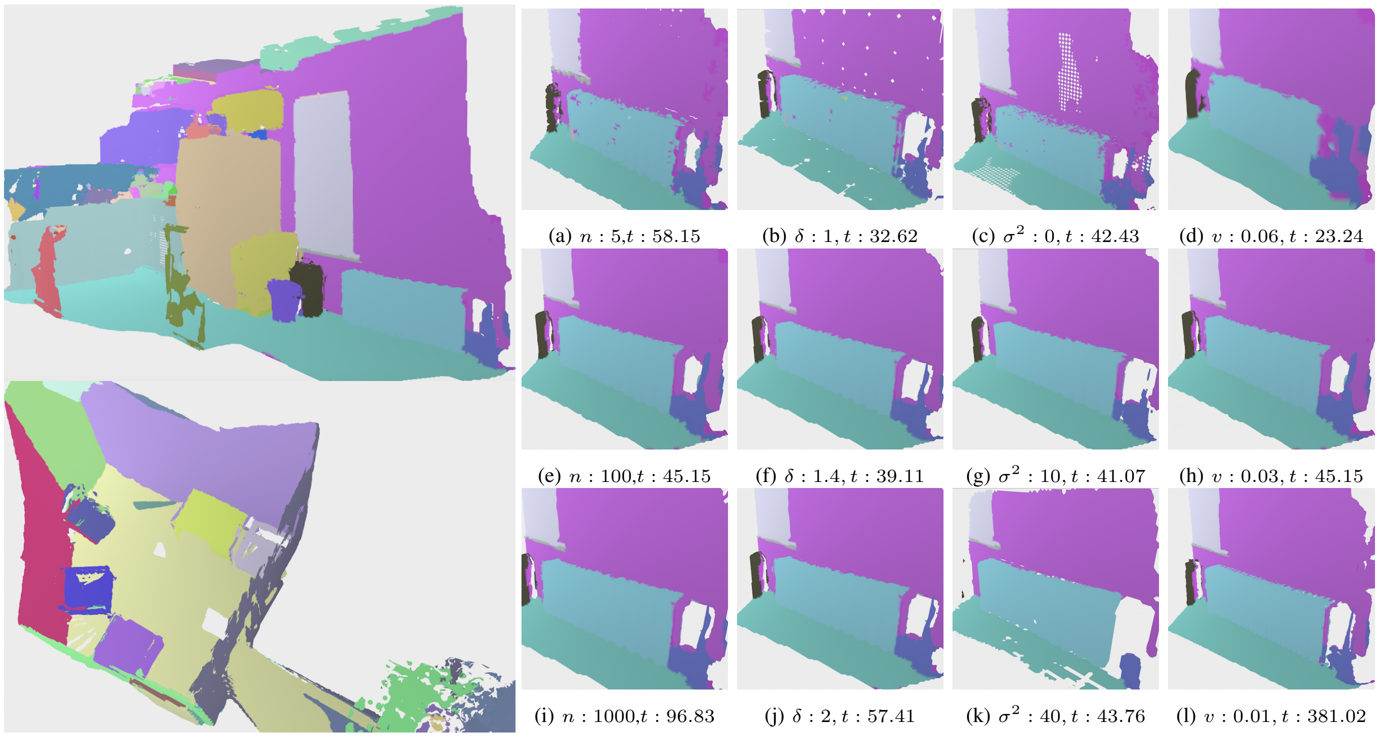
Fig. 10: Evaluation of the SDF reconstruction time (sec) and error (m) of our incremental sparse GP regression algorithm on the Cow and Lady dataset [43] and in comparison with Fiesta [44]. The errors are evaluated with respect to the ground-truth scene point cloud provided by the dataset. Training is done with 829 depth images and known camera trajectory. The default parameters for our algorithm are $max(N) = 200$, $\delta = 1.5$, $\sigma^2 = 25$, $voxel\ size = 0.1$, $frame\ size = 5$, and SDF truncation value $3 \times voxel\ size$.

there is a trade-off between accuracy and computational efficiency. For the hierarchical tree structure, as observed in Sec. IX-A2, it is critical to select the maximum number of pseudo-points per node as $max(N) > 100$ and the size of the node support region as $\delta > 1.4$ to ensure continuous surface representation, robustness to noise, and sufficient accuracy of the decomposition into separate GPs. The larger these two parameters are, however, the more computationally expensive our algorithm becomes. Our experiments suggest optimal choices around $\delta = 1.5$ and $max(N) = 200$.

The noise parameter σ^2 of the GP model depends on the measurement data. For the RGB-D data used in our experiments $\sigma^2 = 3$ was a good choice. The $voxel\ size$ parameter depends on the level of detail that needs to be captured in the reconstruction, with smaller values increasing the accuracy but also the computational complexity. In our experiments, $voxel\ size = 0.03$ was a good choice. A trade-off between accuracy and computation time is associated with the choice of $frame\ size$, which determines how many pseudo-points are added per sensor ray (Fig. 5). While $frame\ size = 2$ is very efficient and captures fine details, as discussed in Fig. 6 and Sec. IX-C, it might miss surface boundaries. Choosing $frame\ size = 3$ alleviates this issue while keeping the approach efficient. If TSDF reconstruction accuracy close to the surface, in addition to the surface extraction itself, is important, then $frame\ size \geq 5$ is a good choice (Fig. 10).

C. Comparison with Deep TSDF Reconstruction

This section compares our sparse GP approach to IGR [32], a deep learning approach for SDF reconstruction introduced in Sec. II. We evaluated the two approaches on five car


(a) $n: 5, t: 58.15$ (b) $\delta: 1, t: 32.62$ (c) $\sigma^2: 0, t: 42.43$ (d) $v: 0.06, t: 23.24$

(e) $n: 100, t: 45.15$ (f) $\delta: 1.4, t: 39.11$ (g) $\sigma^2: 10, t: 41.07$ (h) $v: 0.03, t: 45.15$

(i) $n: 1000, t: 96.83$ (j) $\delta: 2, t: 57.41$ (k) $\sigma^2: 40, t: 43.76$ (l) $v: 0.01, t: 381.02$

Fig. 11: Single-robot reconstructions of sequence 255 (top left), containing 2450 RGB-D images and 85 semantic categories (in random colors), and sequence 011 (bottom left), containing 3700 RGB-D images and 61 semantic categories (in random colors), of the SceneNN dataset [88]. The incremental sparse GP TSDF mapping process took 1040.41 sec. for sequence 255 and 1885.72 sec. for sequence 011. The following default parameters were used for the hierarchical tree structure: $\delta = 1.5$, $n = max(N) = 100$ and the GP training: $\sigma^2 = 3$, $v = voxel\ size = 0.03$, $frame\ size = 2$. On the right we see the effect of these parameters (t is time in seconds) on the metric-semantic reconstruction over 140 RGB-D images.

instances from the ShapeNet dataset [90]. Both methods were trained using 100 depth images of size 512×512, obtained from a sphere around the object with camera orientation facing the object. To make the measurement noise independent of the object scale, each element of the depth image was multiplied with Gaussian noise with mean 1 and three levels of standard deviation: 0 (no noise), 0.025 (low noise), and 0.05 (high noise). Our method was trained on a single CPU with parameters: $\sigma^2 = 3$, $voxel\ size = 0.03$, $frame\ size = 2$, and tree support regions with $\delta = 1.5$ and $max(N) = 200$. IGR was trained without a latent shape vector (single shape estimation) on an Nvidia GTX 1080 Ti GPU with initial learning rate 0.005 and decay factor of 2 every 200 steps for $2k$ iterations. Mesh reconstruction from the SDF values was performed using Marching Cubes [87] with the same resolution of $voxel\ size = 0.015$ for both methods.

Qualitative results for three of the object instances are shown in Fig. 12. The color patterns observed in the variance prediction of our method correspond to convex object parts which more visible (so the uncertainty is lower) from various camera poses. The meshes in ShapeNet are combinations of flat faces and the object surfaces are not smooth. Quantitative results for noise evaluation are shown in Table I. In the absence of noise IGR is more accurate but the difference is minor. Examining the qualitative results in Fig. 12 suggests that our method is able to capture fine details more precisely. As the measurement noise increases, our method's reconstruction accuracy remains robust while IGR's accuracy deteriorates. Fig. 12 shows that with large measurement noise the reconstruction from IGR may sometimes fail, while our method is still able to capture the overall object shape.

TABLE I: Quantitative comparison between our approach and IGR [32] for TSDF reconstruction, averaged over 5 car instances from ShapeNet [90] and evaluated using the metrics from [91]. The metrics are computed after the reconstructed meshes are normalized in a unit-length bounding box.

Method	Noise st. dev.	Chamfer-L_2	Chamfer-L_1	Accuracy	Completeness
Ours	0	0.0010	0.0091	0.0078	0.0105
IGR		0.0007	0.0071	0.0072	0.0070
Ours	0.025	0.0007	0.0104	0.0114	0.0093
IGR		0.0014	0.0262	0.0269	0.0255
Ours	0.05	0.0013	0.0198	0.0248	0.0148
IGR		0.0072	0.0677	0.0863	0.0492

The total training time per instance for IGR, including mesh reconstruction, was 783.27 sec. Timing results for our method are provided in Table II. Our method was able to process depth images for training set construction and incremental GP training at roughly 6.2 Hz. In robotics applications, such as autonomous navigation, only the dataset construction and GP training steps need to be performed online and a small local portion of the TSDF values may be predicted for collision checking. Prediction at a single test-point took 7.7 μs. If complete TSDF reconstruction and mesh extraction are considered, our method took 66.4 sec to process 100 images.

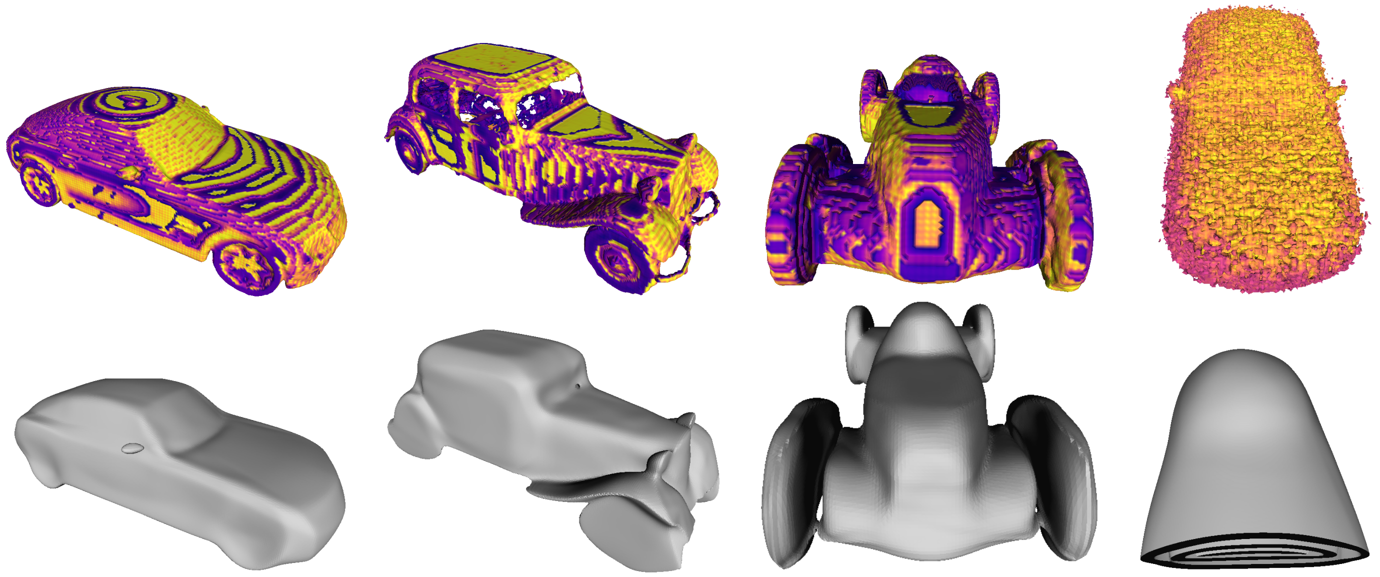


Fig. 12: Reconstruction of car instances from the ShapeNet dataset using our incremental GP approach (top row) and IGR [32] (bottom row). The color indicates the reconstruction variance provided by our model, ranging from cold (low confidence) to hot (high confidence). The first three car instances are reconstructed from RGB-D images with zero measurement noise, while the last instance is obtained with noise standard deviation of 0.05.

TABLE II: Profiling our python implementation of Alg. 1 with average times obtained from 5 ShapeNet [90] with 100 RGB-D images each.

Online TSDF mapping without mesh reconstruction	
Data construction and training (42214 pseudo-points, 944970 updates)	16.02 s
Single pseudo-point update	17.13 μs
Single test-point prediction	7.68 μs
Image processing and training rate	**6.24 Hz**
Online TSDF mapping with mesh reconstruction	
Data construction and training (42214 pseudo-points, 944970 updates)	16.02 s
TSDF prediction (4138478 test points)	31.78 s
Marching cubes mesh reconstruction (944 × 969 × 1066 grid)	11.00 s
Other operations	7.58 s
Total reconstruction time (100 RGB-D images)	66.38 s
Image processing, training, and reconstruction rate	**1.51 Hz**

D. Multi-Robot 3-D Evaluation

Finally, we evaluate our distributed GP regression on the Cow and Lady and SceneNN datasets. We split the RGB-D image sequences into equal parts and consider each as data obtained by a different robot. As in the 2-D simulation, we use three robots with communication weight matrix W in (29). Each robot uses the distributed update rule in (27) and communication continues for 2 rounds after the last RGB-D image from the individual robot sequences is received. The individual robot parameters are the same as in the single-robot experiments in Sec. IX-A. The choice of additional communication rounds is due to Prop. 4, where we showed theoretically that $T + n - 1$ rounds are needed for the local GP distributions to agree with the centralized GP estimator. As in the 2-D simulations, to verify Prop. 4 empirically, we compare the mean absolute error (MAE) in (30) between the GP mean and variance of an individual robot and the centralized estimator.

The results from the Cow and Lady dataset are reported in Fig. 13 and Fig. 14, while those from the SceneNN dataset are in Fig. 15 and Fig. 16. The local and centralized reconstruction results are identical in both data sets, which confirms the expected theoretical consistency. The mean and variance MAE curves also behave similarly in both data sets because the errors in the local GP regression are due to unobserved information, that has not yet been received by the robot, rather than measurement noise. As in the 2-D simulation, the peaks in the MAE curves are due to another robot in the network observing a new region that has not yet been observed by this robot. These peaks quickly decrease, which indicates the fast empirical convergence of the distributed sparse GP algorithm.

X. Conclusion

This paper developed an online Gaussian Process regression method that enables a robot team to build metric-semantic maps collaboratively. A theorem to compress repeated observations before GP training, combined with hierarchical data decomposition, allows scaling to large domains. The presence of distance information allows GP regression instead of computationally challenging GP classification to recover the semantic class distribution. Our approach achieves comparable accuracy to deep neural network techniques for scene reconstruction but offers better robustness to noise, incremental training, and uncertainty quantification. It also enables collaborative inference through distributed GP regression with guaranteed finite-time convergence to the distribution of a centralized estimator. Our probabilistic metric-semantic mapping results offer a promising direction for future research in semantic task specifications and uncertainty-aware task planning.

References

[1] B. Curless and M. Levoy, “A volumetric method for building complex models from range images,” in *Conference on Computer Graphics and Interactive Techniques*, 1996, pp. 303–312.

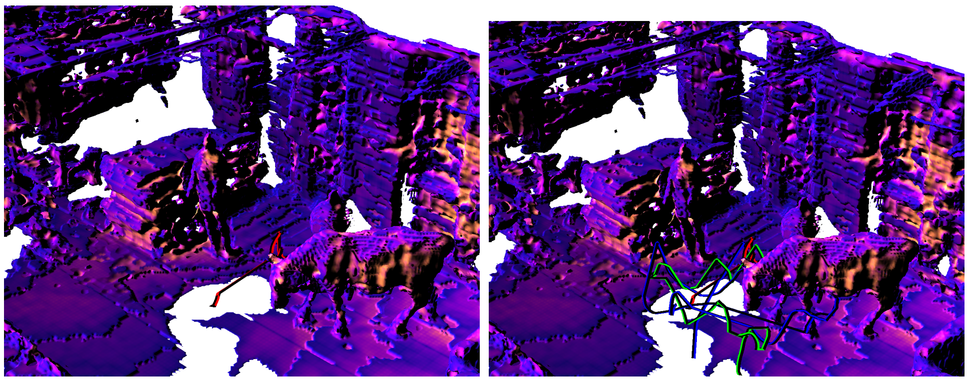


Fig. 13: The Cow and Lady dataset [43] is divided into three equal sequences of about 275 depth images, and each is considered data obtained by one robot. The three camera trajectories are shown in red, green, and blue on the right. The left plot shows the final reconstruction obtained by the first robot. The right plot shows the final reconstruction obtained by centralized GP regression using the observations of all three robots. The orange hues indicate smaller variance. As expected, due to Prop. 4, the reconstruction of robot one is identical with that of the centralized estimator. The initial GP parameters for each robot and object class were $\mu^i_{0,l}(\mathbf{x}) = 0.15$ and $k^i_{0,l}(\mathbf{x}, \mathbf{x}) = 5$.

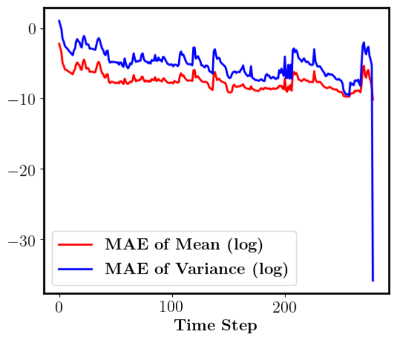


Fig. 14: Log-space plot of the mean absolute error (MAE) between the mean (red) and variance (blue) predictions of robot 1 and centralized GP regression for the sequence in Fig. 13. When the data streaming stops at the end, the MAE approaches zero ($-\infty$ in log space).

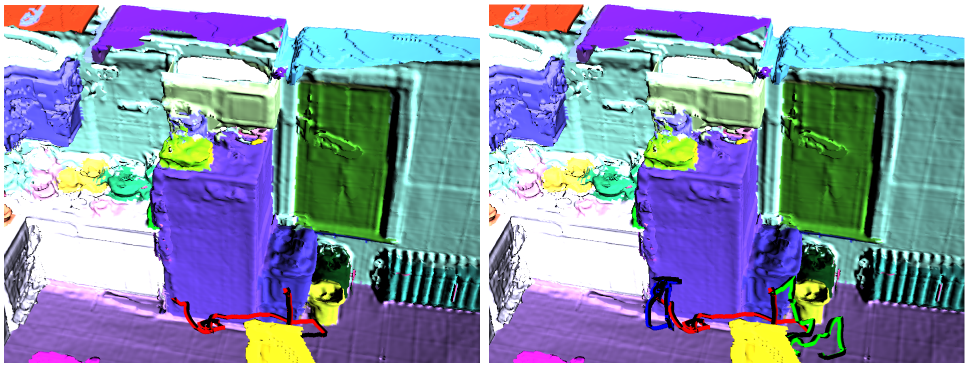


Fig. 15: Sequence 255 of the SceneNN dataset [88] is divided into three equal sequences of about 800 RGB-D images, and each is considered data obtained by a different robot. The three camera trajectories are shown in red, green, and blue on the right. The left plot shows the final metric-semantic reconstruction obtained by the first robot. The right plot shows the final reconstruction obtained by centralized GP regression using the observations of all three robots. As expected, due to Prop. 4, the reconstruction of robot one is identical with that of the centralized estimator. The initial GP parameters for each robot and object class were $\mu^i_{0,l}(\mathbf{x}) = 0.09$ and $k^i_{0,l}(\mathbf{x}, \mathbf{x}) = 5$.

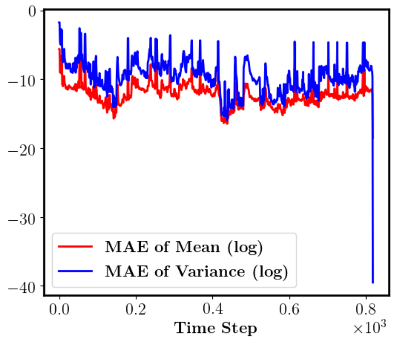


Fig. 16: Log-space plot of the mean absolute error (MAE) between the mean (red) and variance (blue) predictions of robot 1 and centralized GP regression for the sequence in Fig. 15. When the data streaming stops at the end, the MAE approaches zero ($-\infty$ in log space).

[2] M. Kazhdan, M. Bolitho, and H. Hoppe, “Poisson surface reconstruction,” in *Eurographics Symposium on Geometry Processing*, 2006.

[3] X. Wang, M. R. Oswald, I. Cherabier, and M. Pollefeys, “Learning 3d semantic reconstruction on octrees,” in *German Conference on Pattern Recognition*. Springer, 2019, pp. 581–594.

[4] 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, 2013.

[5] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, “Real-time 3d reconstruction at scale using voxel hashing,” *ACM Transactions on Graphics (ToG)*, vol. 32, no. 6, pp. 1–11, 2013.

[6] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, “Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping,” in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.

[7] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, “Semanticfusion: Dense 3d semantic mapping with convolutional neural networks,” in *2017 IEEE International Conference on Robotics and automation (ICRA)*. IEEE, 2017, pp. 4628–4635.

[8] A. Hermans, G. Floros, and B. Leibe, “Dense 3D Semantic Mapping of Indoor Scenes from RGB-D Images,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 2631–2638.

[9] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg, “Joint semantic segmentation and 3d reconstruction from monocular video,” in *European Conference on Computer Vision*. Springer, 2014, pp. 703–718.

[10] J. Zhang, C. Zhu, L. Zheng, and K. Xu, “Fusion-aware point convolution for online semantic 3d scene segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4534–4543.

[11] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[12] C. E. Rasmussen, “Gaussian processes in machine learning,” in *Summer School on Machine Learning*. Springer, 2003, pp. 63–71.

[13] S. O’Callaghan and F. Ramos, “Gaussian process occupancy maps,” *The International Journal of Robotics Research (IJRR)*, vol. 31, no. 1, pp. 42–62, 2012.

[14] S. Kim and J. Kim, “Occupancy Mapping and Surface Reconstruction Using Local Gaussian Processes With Kinect Sensors,” *IEEE Trans. on Cybernetics*, vol. 43, no. 5, pp. 1335–1346, 2013.

[15] M. G. Jadidi, J. V. Miró, R. Valencia, and J. Andrade-Cetto, “Exploration on Continuous Gaussian Process Frontier Maps,” in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 6077–6082.

[16] R. Senanayake and F. Ramos, “Building continuous occupancy maps with moving robots,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[17] M. Ghaffari Jadidi, L. Gan, S. Parkison, J. Li, and R. Eustice, “Gaussian Processes Semantic Map Representation,” *arXiv:1707.01532*, 2017.

[18] J. Hensman, A. Matthews, and Z. Ghahramani, “Scalable Variational Gaussian Process Classification,” in *International Conference on Artificial Intelligence and Statistics*, 2015, pp. 351–360.

[19] T. Galy-Fajou, F. Wenzel, C. Donner, and M. Opper, “Multi-class

gaussian process classification made conjugate: Efficient inference via data augmentation," in *Uncertainty in Artificial Intelligence Conference*, 2020, pp. 755–765.

[20] D. Hernández-Lobato, J. Hernández-lobato, and P. Dupont, "Robust multi-class gaussian process classification," *Advances in neural information processing systems*, vol. 24, pp. 280–288, 2011.

[21] E. Snelson and Z. Ghahramani, "Sparse gaussian processes using pseudo-inputs," in *Advances in neural information processing systems*, 2006, pp. 1257–1264.

[22] J. Hensman, N. Durrande, and A. Solin, "Variational Fourier features for Gaussian processes," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 5537–5588, 2017.

[23] A. Koppel, "Consistent online Gaussian Process regression without the sample complexity bottleneck," in *American Control Conference (ACC)*, 2019, pp. 3512–3518.

[24] A. Koppel, A. S. Bedi, K. Rajawat, and B. M. Sadler, "Optimally compressed nonparametric online learning," *IEEE Signal Processing Magazine*, 2020.

[25] V. Tresp, "A bayesian committee machine," *Neural computation*, vol. 12, no. 11, pp. 2719–2741, 2000.

[26] S. Kim and J. Kim, "Recursive Bayesian Updates for Occupancy Mapping and Surface Reconstruction," in *Australasian Conference on Robotics and Automation (ACRA)*, 2014.

[27] M. Bauer, M. van der Wilk, and C. E. Rasmussen, "Understanding probabilistic sparse gaussian process approximations," in *Advances in neural information processing systems*, 2016, pp. 1533–1541.

[28] C. E. Rasmussen and Z. Ghahramani, "Infinite mixtures of gaussian process experts," in *Advances in neural information processing systems*, 2002, pp. 881–888.

[29] A. Nedić, A. Olshevsky, and C. A. Uribe, "Distributed learning for cooperative inference," *arXiv preprint:1704.02718*, 2017.

[30] A. Jadbabaie, P. Molavi, A. Sandroni, and A. Tahbaz-Salehi, "Non-bayesian social learning," *Games and Economic Behavior*, vol. 76, no. 1, pp. 210–225, 2012.

[31] E. Zobeidi, A. Koppel, and N. Atanasov, "Dense incremental metric-semantic mapping via sparse gaussian process regression," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[32] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, "Implicit geometric regularization for learning shapes," *arXiv preprint arXiv:2002.10099*, 2020.

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

[34] E. Vespa, N. Nikolov, M. Grimm, L. Nardi, P. H. J. Kelly, and S. Leutenegger, "Efficient Octree-Based Volumetric SLAM Supporting Signed-Distance and Occupancy Mapping," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1144–1151, 2018.

[35] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *European Conf. on Computer Vision*, 2014.

[36] R. Dubé, A. Cramariuc, D. Dugas, H. Sommer, M. Dymczyk, J. Nieto, R. Siegwart, and C. Cadena, "SegMap: Segment-based mapping and localization using data-driven descriptors," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 339–355, 2020.

[37] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 647–663, 2012.

[38] J. Behley and C. Stachniss, "Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments," in *Robotics: Science and Systems*, 2018.

[39] L. Teixeira and M. Chli, "Real-time mesh-based scene estimation for aerial inspection," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4863–4869.

[40] E. Piazza, A. Romanoni, and M. Matteucci, "Real-time cpu-based large-scale three-dimensional mesh reconstruction," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1584–1591, 2018.

[41] R. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-Time Dense Surface Mapping and Tracking," in *IEEE Int. Symposium on Mixed and Augmented Reality*, 2011, pp. 127–136.

[42] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion: Real-time dense slam and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.

[43] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017.

[44] L. Han, F. Gao, B. Zhou, and S. Shen, "Fiesta: Fast incremental euclidean distance fields for online motion planning of aerial robots," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2019.

[45] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. J. Leonard, and J. McDonald, "Real-time large-scale dense RGB-D SLAM with volumetric fusion," *The International Journal of Robotics Research (IJRR)*, vol. 34, no. 4-5, pp. 598–626, 2015.

[46] M. Klingensmith, I. Dryanovski, S. S. Srinivasa, and J. Xiao, "Chisel: Real Time Large Scale 3D Reconstruction Onboard a Mobile Device using Spatially Hashed Signed Distance Fields," in *Robotics: science and systems*, vol. 4. Citeseer, 2015, p. 1.

[47] L. Han and L. Fang, "FlashFusion: Real-time Globally Consistent Dense 3D Reconstruction using CPU Computing," in *Robotics: Science and Systems (RSS)*, 2018.

[48] O. Kähler, V. A. Prisacariu, and D. W. Murray, "Real-time large-scale dense 3d reconstruction with loop closure," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 500–516.

[49] V. Reijgwart, A. Millane, H. Oleynikova, R. Siegwart, C. Cadena, and J. Nieto, "Voxgraph: Globally Consistent, Volumetric Mapping Using Signed Distance Function Submaps," *IEEE Robotics and Automation Letters*, 2020.

[50] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto, "Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3037–3044, 2019.

[51] E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers, "Real-time camera tracking and 3d reconstruction using signed distance functions." in *Robotics: Science and Systems*, 2013.

[52] H. Oleynikova, M. Burri, Z. Taylor, J. I. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online uav replanning," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2016.

[53] K. Saulnier, N. Atanasov, G. Pappas, and V. Kumar, "Information theoretic active exploration in signed distance fields," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2020.

[54] S. Kim and J. Kim, "GPmap: A Unified Framework for Robotic Mapping Based on Sparse Gaussian Processes," in *International Conference on Field and Service Robotics*, 2015.

[55] J. Wang and B. Englot, "Fast, accurate gaussian process occupancy maps via test-data octrees and nested bayesian fusion," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016, pp. 1003–1010.

[56] F. Ramos and L. Ott, "Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1717–1730, 2016.

[57] W. Zhi, L. Ott, R. Senanayake, and F. Ramos, "Continuous occupancy map fusion with fast bayesian hilbert maps," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 4111–4117.

[58] R. Senanayake and F. Ramos, "Bayesian Hilbert Maps for Continuous Occupancy Mapping in Dynamic Environments," in *Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 78, 2017, pp. 458–471.

[59] R. Senanayake, S. O'Callaghan, and F. Ramos, "Learning highly dynamic environments with stochastic variational inference," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2532–2539.

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

[61] S. Guo and N. A. Atanasov, "Information filter occupancy mapping using decomposable radial kernels," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 7887–7894.

[62] V. Vineet, O. Miksik, M. Lidegaard, M. Nießner, S. Golodetz, V. A. Prisacariu, O. Kähler, D. W. Murray, S. Izadi, P. Pérez, and P. H. S. Torr, "Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 75–82.

[63] S. Sengupta and P. Sturgess, "Semantic octree: Unifying recognition, reconstruction and representation via an octree constrained higher order mrf," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 1874–1879.

[64] S. Yang, Y. Huang, and S. Scherer, "Semantic 3D occupancy mapping through efficient high-order CRFs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 590–597.

[65] Z. Zhao and X. Chen, "Building 3D semantic maps for mobile robots using RGB-D camera," *Intelligent Service Robotics*, vol. 9, no. 4, pp. 297–309, 2016.

[66] L. Gan, R. Zhang, J. W. Grizzle, R. M. Eustice, and M. Ghaffari Jadidi, "Bayesian spatial kernel smoothing for scalable dense semantic

mapping," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 790–797, 2020.

[67] K. Zheng and A. Pronobis, "From pixels to buildings: End-to-end probabilistic deep networks for large-scale semantic mapping," *arXiv preprint arXiv:1812.11866*, 2018.

[68] Y. Siddiqui, J. Thies, F. Ma, Q. Shan, M. Nießner, and A. Dai, "RetrievalFuse: Neural 3D Scene Reconstruction with a Database," *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[69] R. Chabra, J. E. Lenssen, E. Ilg, T. Schmidt, J. Straub, S. Lovegrove, and R. Newcombe, "Deep local shapes: Learning local sdf priors for detailed 3d reconstruction," in *European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 608–625.

[70] C. Jiang, A. Sud, A. Makadia, J. Huang, M. Nießner, T. Funkhouser *et al.*, "Local implicit grid representations for 3d scenes," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6001–6010.

[71] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 523–540.

[72] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[73] K. Rahnama Rad and A. Tahbaz-Salehi, "Distributed parameter estimation in networks," in *IEEE Conference on Decision and Control (CDC)*, 2010, pp. 5050–5055.

[74] N. Atanasov, R. Tron, V. M. Preciado, and G. J. Pappas, "Joint estimation and localization in sensor networks," in *IEEE Conference on Decision and Control (CDC)*, 2014, pp. 6875–6882.

[75] A. Nedić, A. Olshevsky, and C. A. Uribe, "Distributed Gaussian learning over time-varying directed graphs," in *Asilomar Conference on Signals, Systems and Computers*, 2016, pp. 1710–1714.

[76] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. I. Christensen, and F. Dellaert, "Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models," *The International Journal of Robotics Research*, vol. 36, no. 12, pp. 1286–1311, 2017.

[77] P. Koch, S. May, M. Schmidpeter, M. Kühn, C. Pfitzner, C. Merkl, R. Koch, M. Fees, J. Martin, D. Ammon, and A. Nüchter, "Multi-robot localization and mapping based on signed distance functions," *Journal of Intelligent & Robotic Systems*, vol. 83, no. 3-4, pp. 409–428, 2016.

[78] P. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "DOOR-SLAM: Distributed, Online, and Outlier Resilient SLAM for Robotic Teams," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1656–1663, 2020.

[79] A. Milioto and C. Stachniss, "Bonnet: An Open-Source Training and Deployment Framework for Semantic Segmentation in Robotics using CNNs," in *IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019.

[80] T. D. Bui, J. Yan, and R. E. Turner, "A unifying framework for sparse gaussian process approximation using power expectation propagation," *stat*, vol. 23, no. 1050, 2016.

[81] J. Oliva, B. Póczos, and J. Schneider, "Distribution to distribution regression," in *International Conference on Machine Learning*. PMLR, 2013, pp. 1049–1057.

[82] K. B. Petersen and M. S. Pedersen, "The matrix cookbook," 2008.

[83] A. Tahbaz-Salehi and A. Jadbabaie, "A Necessary and Sufficient Condition for Consensus Over Random Networks," *IEEE Transactions on Automatic Control*, vol. 53, no. 3, pp. 791–795, 2008.

[84] A. Nedic and A. Ozdaglar, "Distributed Subgradient Methods for Multi-Agent Optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[85] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 169–182, 2005.

[86] F. Saadatniaki, R. Xin, and U. A. Khan, "Decentralized optimization over time-varying directed graphs with row and column-stochastic matrices," *IEEE Transactions on Automatic Control*, vol. 65, no. 11, pp. 4769–4780, 2020.

[87] T. S. Newman and H. Yi, "A survey of the marching cubes algorithm," *Computers & Graphics*, vol. 30, no. 5, pp. 854–879, 2006.

[88] B.-S. Hua, Q.-H. Pham, D. T. Nguyen, M.-K. Tran, L.-F. Yu, and S.-K. Yeung, "Scenenn: A scene meshes dataset with annotations," in *International Conference on 3D Vision (3DV)*, 2016.

[89] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *Computer Graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[90] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," *arXiv:1512.03012*, 2015.

[91] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3D reconstruction in function space," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4460–4470.

Ehsan Zobeidi is a PhD student in Electrical and Computer Engineering at University of California San Diego (UCSD). He received a B.S. degree in Electrical Engineering (Communications) and a B.S. degree in Computer Science from Sharif University of Technology, Tehran, Iran in 2017. He received his M.S. degree in Electrical Engineering at UCSD in the beginning of 2020. His research interests include statistics, machine learning and their application to robotics and computer vision.

Alec Koppel is a Research Scientist at the U.S. Army Research Laboratory in the Computational and Information Sciences Directorate since September of 2017. He completed his Master's degree in Statistics and Doctorate in Electrical and Systems Engineering, both at the University of Pennsylvania (Penn) in August of 2017. Before coming to Penn, he completed his Master's degree in Systems Science and Mathematics and Bachelor's Degree in Mathematics, both at Washington University in St. Louis (WashU), Missouri. He is a recipient of the 2016 UPenn ESE Dept. Award for Exceptional Service, an awardee of the Science, Mathematics, and Research for Transformation (SMART) Scholarship, a co-author of Best Paper Finalist at the 2017 IEEE Asilomar Conference on Signals, Systems, and Computers, a finalist for the ARL Honorable Scientist Award 2019, an awardee of the 2020 ARL Director's Research Award Translational Research Challenge (DIRA-TRC), a 2020 Honorable Mention from the IEEE Robotics and Automation Letters, and mentor to the 2021 ARL Summer Symposium Best Project Awardee. His research interests are in optimization and machine learning. Currently, he focuses on approximate Bayesian inference, reinforcement learning, and decentralized optimization, with an emphasis on applications in robotics and autonomy.

Nikolay Atanasov is an Assistant Professor of Electrical and Computer Engineering at the University of California San Diego. He obtained a B.S. degree in Electrical Engineering from Trinity College, Hartford, CT, in 2008 and M.S. and Ph.D. degrees in Electrical and Systems Engineering from the University of Pennsylvania, Philadelphia, PA, in 2012 and 2015, respectively. His research focuses on robotics, control theory, and machine learning, applied to active sensing using ground and aerial robots. He works on probabilistic perception models that unify geometry and semantics and on optimal control and reinforcement learning approaches for minimizing uncertainty in these models. Dr. Atanasov's work has been recognized by the Joseph and Rosaline Wolf award for the best Ph.D. dissertation in Electrical and Systems Engineering at the University of Pennsylvania in 2015, the best conference paper award at the International Conference on Robotics and Automation in 2017, and the NSF CAREER award in 2021.