Distributed Gaussian Process Mapping for Robot Teams with Time-varying Communication

James Di[1], Ehsan Zobeidi[2], Alec Koppel[3], Nikolay Atanasov[2]

***Abstract*— Multi-agent mapping is a fundamentally important capability for autonomous robot task coordination and execution in complex environments. While successful algorithms have been proposed for mapping using individual platforms, cooperative online mapping for teams of robots remains largely a challenge.A critical question to enabling this capability is how to process and aggregate incrementally observed local information among individual platforms, especially when their ability to communicate is intermittent. We employ truncated signed-distance field (TSDF) as the map representation, and propose an Incremental Sparse Gaussian Process (GP) methodology to regress over TSDF for multi-robot mapping. Doing so permits each robot in the network to track a local estimate of an approximated GP posterior and perform weighted averaging of its parameters with its (possibly time-varying) set of neighbors. We focus on probabilistic variants of mapping due to its potential utility in down-stream tasks such as uncertainty-aware path-planning. We establish conditions on the GP representation, as well as communications protocol, such that robots' local GPs converge to the one with globally aggregated information. We further provide experiments that corroborate our theoretical findings for probabilistic multi-robot mapping.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) refers to the ability of a robot to identify its location within an unknown environment while simultaneously constructing a map of its surroundings. SLAM is critically important to enable real-time robot operation, using only on-board sensing [2], [3]. We focus on a distributed mapping problem, where multiple robots acquire sensor data and seek to aggregate it to boost the statistical accuracy of their maps [4], [5]. This setting is important for reducing the amount of time to map an unknown environment at sufficiently high accuracy. However, collecting and optimizing the data at a central location requires every robot to send all of its observations followed by a large-scale optimization, rendering the process excessively slow, expensive, and brittle in robotics applications with challenging communication infrastructure [6]. Instead of using a centralized architecture, in this work we allow the robots to process their local observations incrementally and communicate in a *distributed* manner. The challenge then becomes mixing the information received at each robot from its neighbors that are within communication range to collaboratively build the map, without knowing the robot network topology a priori.

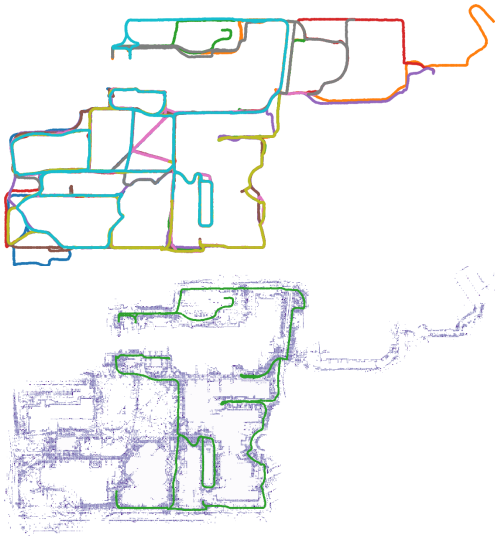


Fig. 1: This work tackles probabilistic continuous-space mapping using point-cloud observations from a robot team with time-varying communication. The trajectories of 10 sequences from the NCLT dataset [1], representing different robots, are shown at the top. The dataset provides 2D LiDAR scans and other sensor data collected in a square kilometer region of a university campus over several months. In our setting, the robot communication is intermittent as robots move in and out of communication range. Each robot collaborates with its teammates to reconstruct a probabilistic truncated signed distance field (TSDF) map of the environment. The TSDF map and trajectory of robot 3 is shown at the bottom. Even though robot 3 does not visit the entire environment, it obtains a complete TSDF map of the environment.

We use truncated signed distance field (TSDF) [7] as our environment representation, which intuitively measures the distance to the closest obstacle surface, up to a truncation value for each point in the environment. Compared to occupancy grid representations [8], TSDF quantifies both occupancy and distance to the closest object surface, and is more attractive for down-stream tasks such as collision avoidance and accurate surface reconstruction. However, updating a global map whenever new observations arrive is expensive. Hence, to keep the computational complexity in check, inspired by map decomposition methods such as Octomap

We gratefully acknowledge support from ARL DCIST CRA W911NF-17-2-0181 and NSF NRI CNS-1830399.

[1] James Di is with Treeswift Inc, Philadelphia, PA 19146, USA, james@treeswift.com.

[2] Ehsan Zobeidi and Nikolay Atanasov are with Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093, USA, {ezobeidi, natanasov}@ucsd.edu.

[3] Alec Koppel is with Supply Chain Optimization Technologies, Amazon, Bellevue, WA 98004, USA, aekoppel@amazon.com. Work completed while at the U.S. Army Research Laboratory in Adelphi, MD 20783.

[9] and voxel-hashing [10], we employ a QuadTree (Octree in 3D) data structure to reduce computation complexity, and only update maps of the relevant regions when new sensor observations are collected.

Furthermore, we focus on the case that individual platforms seek to construct a probabilistic map, as uncertainty quantification is important for collision avoidance. A Gaussian Process (GP) is a non-parametric model that provides a natural choice for tracking the posterior of the map while providing uncertainty information [11]. However, the training procedure of GP is cubic in the number of samples, and various methods [12], [13] have been proposed to strike a balance between computational effort and statistical accuracy in GP inference. We develop an approach based off *pseudo-point* approximations as in [14], which reduces the training complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(m^2 n)$, where n is the number of training samples, m is the number of pseudo-points, and whose convergence has recently been characterized [15]. When $m \ll n$ this could lead to significant computational savings.

We further develop a weighted averaging scheme for propagating distributions of TSDF estimated by individual GPs across the network, inspired by consensus protocols [16], [17]. While sub-optimal compared to approaches based on Lagrangian relaxation of consensus constraints, such as primal-dual [18], dual [19], and ADMM [20] methods, the proposed approach is simple and efficient, making it suitable for distributed probabilistic inference.

Thus, to achieve distributed, probablistic, online and efficient mapping with uncertainy information, we construct an algorithm based upon incremental sparse Gaussian Processes (GP) with pseudo-input approximations, which are regressed over sequentially observed TSDF measurements taken by each robot. Information mixing is executed through a parametric representations of the GP mean and covariance functions. A key point of departure of this work from our prior work in [21] is the consideration of a time-varying network. This is important in settings where communication is intermittent and dependent both on the robot locations and the environmental characteristics, e.g., common in underwater and underground [6] environments. In summary, the *contributions* of this paper are to:

- develop a distributed protocol for mixing incremental pseudo-points GP posterior of TSDF over a time-varying network,
- establish a convergence guarantee under suitable conditions on the pseudo-points, communications network, and input space,
- corroborate the proposed algorithm on two real-world LiDAR datasets, one of which is large-scale.

We demonstrate both theoretically and empirically that the proposed distributed mapping algorithm with time-varying communication converges asymptotically to a centralized estimator, which relies on the information of all robots.

II. Problem Statement

We consider the problem of mapping a d-dimensional environment ($d \in \{2, 3\}$ in practice) with occupied space $\Omega \subset \mathbb{R}^d$ and free space $\mathcal{F} \subset \mathbb{R}^d$. We aim to estimate a truncated signed distance function (TSDF) [10] as a continuous representation of the environment. The truncated signed distance from a point x to the boundary $\partial\Omega$ of the occupied space Ω is defined as:

$$g(x) = \begin{cases} \min(d(x, \partial\Omega), h), & \text{if } x \notin \Omega, \\ -\min(d(x, \partial\Omega), h), & \text{if } x \in \Omega, \end{cases} \tag{1}$$

where $h > 0$ is a pre-defined truncation value and:

$$d(x, \partial\Omega) = \inf_{y \in \partial\Omega} \|x - y\|_2. \tag{2}$$

The TSDF $g(x)$ provides the (truncated) minimum distance from x to the boundary of the occupied space and is negative if x is within the occupied space.

We employ a team of n robots to gather observations of the environment over a time horizon $0, \ldots, T$. The observation of robot i at time t is a point-cloud $Z_t^i \subset \mathbb{R}^d$ obtained, e.g., from a LiDAR scanner, depth camera, or another range sensor. We assume that the robot positions $p_t^i \in \mathbb{R}^d$ and orientations $R_t^i \in SO(d)$ are known for all i, t from an odometry algorithm, e.g. scan-matching [22] or pose graph optimization [23]. The world-frame coordinates of a point $z \in Z_t^i$ observed by the robot can be obtained via $p_t^i + R_t^i z$.

We assume the the team of robots are able to exchange the information over an undirected time-varying graph $G_t = (\mathcal{V}, \mathcal{E}_t)$ with nodes $\mathcal{V} = \{1, \ldots, n\}$, corresponding to the robots, and edges $\mathcal{E}_t \subseteq \mathcal{V} \times \mathcal{V}$. If two robots $i, j \in \mathcal{V}$ are able to communicate at time t, then an edge $(i, j) \in \mathcal{E}_t$ is present in the graph. The robots that robot i can communicate with at time t are called its neighbors and will be denoted by the set $\mathcal{N}_t^i = \{j \in \mathcal{V} | (i, j) \in \mathcal{E}_t\}$. We aim to design a fully distributed TSDF mapping approach, in which the robots communicate only with their neighbors and place minimal restrictions on the communication structure. We consider time-varying networks, in which the graph G_t may be instantaneously disconnected but the union of the graphs over a period of time B is connected. This assumption is much weaker than requiring the robots to be in constant communication and is utilized for many results in multi-agent coordination and distributed optimization [24], [25].

Assumption 1. The graph sequence $G_t = (\mathcal{V}, \mathcal{E}_t)$ is *uniformly connected*, i.e., there exists an integer $B > 0$ (potentially unknown to the robots) such that the graph with node set $\mathcal{V}$ and edge set $\mathcal{E}_k^B = \bigcup_{t=kB}^{(k+1)B-1} \mathcal{E}_t$ is connected for all $k = 0, 1, \ldots$.

For each robot i, our goal is to incrementally infer a posterior distribution over the TSDF representation g of the environment, conditioned on the sequential observations Z_t^i of robot i as well as the information received from its neighbors $\mathcal{N}_t^i$ for $t = 0, \ldots, T$. This amounts to an online distributed Bayesian inference problem over a time-varying network. Our approach to this problem is described in the following section.

III. Technical Approach

We organize our technical approaches into the following sections. Section III-A discusses the TSDF estimation

framework for a single agent, leveraging sparse pseudo-point Gaussian Process. Section III-B presents the distributed update protocol over time-varying network of robots, with proposition and proof over the convergence of the algorithm.

A. Regressing TSDF via pseudo-point Gaussian Processes

To estimate a TSDF $g(x)$ representation of the environment using a single agent, we leverage pseudo-point approximations of Gaussian Processes. This provides a way to infer the distribution in a parametrically efficient manner. Before doing so, we review the key steps of Gaussian Process regression and sparse pseudo-point GP approximation.

Gaussian Processes. A Gaussian Process (GP) $g(x) \sim \mathcal{GP}(\mu_0(x), k_0(x,x'))$ is a stochastic process such that any finite collection of its realizations $[g(x_1),\dots,g(x_K)]^\top$ is jointly Gaussian with mean $\mu(x) = [\mu_0(x_1),\dots,\mu_0(x_K)]^\top$ and covariance with elements $k_0(x_k, x_\ell)$. We employ this nonparametric model to hypothesize that the robot's observations are corrupted by zero-mean Gaussian noise: $y_i = g(x_i) + \epsilon_i$, where the noise satisfies $\epsilon_i \sim N(0,\sigma_i^2)$. In this work, $\{y_i\}$ are observations of the TSDF function g, which we estimate from the point cloud observations Z_t^i as described at the end of this section.

In our setting, the robot collects observations $\mathcal{D} = (\mathcal{X}, y)$, where $\mathcal{X} = \{x_i\}$ denotes the input vectors and $y = \{y_i\}$ denotes the corresponding TSDF values. The posterior of the GP is $g(x)|\mathcal{D} \sim \mathcal{GP}(\mu(x), k(x,x'))$. The associated conditional mean and covariance functions of g, estimated by the GP are:

$$\begin{aligned}\mu(x) &= \mu_0(x) + k_0(x,\mathcal{X})(k_0(\mathcal{X},\mathcal{X}) + \sigma^2 I)^{-1}(y - \mu_0(\mathcal{X}))\\ k(x,x') &= k_0(x,x') - k_0(x,\mathcal{X})(k_0(\mathcal{X},\mathcal{X}) + \sigma^2 I)^{-1}k_0(\mathcal{X},x')\end{aligned} \tag{3}$$

Complexity Reduction. The computational complexity of the training procedure is $O(n^3)$ with sample size $n = |\mathcal{D}|$, due to the inversion of the kernel matrix $k_0(\mathcal{X},\mathcal{X})$ in (3). We adopt an approximation based on a set of pseudo-points $\mathcal{P} \subset \mathcal{X}$, where $|\mathcal{P}| = k \ll n$, as in [14]. The key is that we parametrize the model with the pseudo-points $\mathcal{P}$ and g evaluated at $\mathcal{P}$, which we denote as pseudo targets $\bar{g}$. We can then obtain the distribution of g, by integrating out the derived distribution on $\bar{g}$ and the likelihood of the model.

The distribution of $g(x)$ conditioned on the input x, $\mathcal{P}$ and $\bar{g}$, i.e. $p(g(x)|\bar{g})$ is Gaussian with parameters:

$$\begin{aligned}\mu(x) &= \mu_0(x) + k_0(x,\mathcal{P})(k_0(\mathcal{P},\mathcal{P}) + \sigma^2 I)^{-1}(\bar{g} - \mu_0(\mathcal{P}))\\ k(x,x') &= k_0(x,x') - k_0(x,\mathcal{P})(k_0(\mathcal{P},\mathcal{P}) + \sigma^2 I)^{-1}k_0(\mathcal{P},x')\end{aligned} \tag{4}$$

Since we can assume the pseudo-targets come from the same distribution as the dataset $\mathcal{D}$, we place the same prior on $\bar{g} \sim \mathcal{N}(\mu_0, k_0)$, and after using Bayes Rules on (4) and the prior we can write $\bar{g}|\mathcal{X}, y, \mathcal{P}$ $\bar{g}|\mathcal{X}, y$ as:

$$\begin{aligned}\mu(\mathcal{P}) &= \mu_0(\mathcal{P}) + k_0(\mathcal{P},\mathcal{P})(k_0(\mathcal{P},\mathcal{P}) + \Gamma)^{-1}\gamma\\ \Sigma(\mathcal{P}) &= k_0(\mathcal{P},\mathcal{P})(k_0(\mathcal{P},\mathcal{P}) + \Gamma)^{-1}k_0(\mathcal{P},\mathcal{P})\end{aligned} \tag{5}$$

with weighting factors $\Gamma = k_0(\mathcal{P},\mathcal{X})(\Lambda + \sigma^2 I)^{-1}k_0(\mathcal{X},\mathcal{P})$, $\Lambda = k_0(\mathcal{X},\mathcal{X}) - k_0(\mathcal{X},\mathcal{P})k_0(\mathcal{P},\mathcal{P})^{-1}k_0(\mathcal{P},\mathcal{X})$, and $\gamma = k_0(\mathcal{P},\mathcal{X})(\Lambda + \sigma^2 I)^{-1}(y - \mu_0(\mathcal{X}))$.

Using the definition of info matrix $\Omega(\mathcal{P}) = \Sigma(\mathcal{P})^{-1}$ and information mean $\omega(\mathcal{P}) = \Omega\mu(\mathcal{P})$, equivalently to (5) the info mean and info matrix of $\mathcal{P}$ can be written as:

$$\begin{aligned}\omega(\mathcal{P}) &= \Omega\mu_0(\mathcal{P}) + k_0(\mathcal{P},\mathcal{P})^{-1}\gamma\\ \Omega(\mathcal{P}) &= k_0(\mathcal{P},\mathcal{P})^{-1}(k_0(\mathcal{P},\mathcal{P}) + \Gamma)k_0(\mathcal{P},\mathcal{P})^{-1}\end{aligned} \tag{6}$$

Lastly, integrating out $\bar{g}|\mathcal{X}, y$ in (6) and $g(x)|x, \mathcal{P}, \bar{g}$ $g(x)|\bar{g}$ in (4), the TSDF posterior $g(x)|\mathcal{X}, y$ is distributed as Gaussian $\mathcal{N}(\mu(x), k(x,x'))$ with parameters:

$$\begin{aligned}\mu(x) &= \mu_0(x) + k_0(x,\mathcal{P})k_0(\mathcal{P},\mathcal{P})^{-1}(\Omega^{-1}\omega - \mu_0(\mathcal{P}))\\ k(x,x') &= k_0(x,\mathcal{P})k_0(\mathcal{P},\mathcal{P})^{-1}\Omega^{-1}k_0(\mathcal{P},\mathcal{P})^{-1}k_0(\mathcal{P},x')\\ &\quad + k_0(x,x') - k_0(x,\mathcal{P})k_0(\mathcal{P},\mathcal{P})^{-1}k_0(\mathcal{P},x')\end{aligned} \tag{7}$$

This is a key complexity reduction of GP posterior computations. In general, the pseudo-points $\mathcal{P}$ could be arbitrary locations in the environment, and do not need to come from the dataset $\mathcal{X}$. However, in incremental and distributed settings, the team of robots may encounter repeated observations of the same locations. To keep the complexity of model proportional to complexity of environment, we pick the pseudo points out of a fixed grid of the environment. We incorporate the pseudo-point aggregation technique in [26] as the pseudo-point values $\bar{g}$, which is proven to have the same distribution as the pseudo-point GP mean and covariance in (7). The GP using the aggregated statistics have mean and covariance:

$$\begin{aligned}\mu(x) &= \mu_0(x) + k_0(x,\mathcal{P})Q(\zeta - \mu_0(\mathcal{P}))\\ k(x,x') &= k_0(x,x') - k_0(x,\mathcal{P})Qk_0(\mathcal{P},x')\end{aligned} \tag{8}$$

where $Q^{-1} \coloneqq k_0(\mathcal{P},\mathcal{P}) + \sigma^2 \text{diag}(m)^{-1}$

The locations of the pseudo-points are denoted by $\mathcal{P} \subset \mathcal{X}$, and ζ is a vector containing the arithmetic average of the observations for $\mathbf{p} \in \mathcal{P}$, and m is the vector containing count of observations for each $\mathbf{p} \in \mathcal{P}$.

Tree Data Structures. To keep the computational effort under control, for each agent i we use one QuadTree Tr_t^i (Octree in 3D) to keep track of the pseudo points at time t. Let $L_t^{i,k}$ denote the kth QuadTree *leaf* node for tree Tr_t^i, which may be the root when the tree has not been split, or a child of Tr_t^i. When new pseudo-points are inserted, if the number of pseudo-points in $L_t^{i,k}$ exceeds a threshold *maxLeafSize*, the node $L_t^{i,k}$ is split recursively until all leaves of $L_t^{i,k}$ have less than *maxLeafSize* pseudo-points.

Pseudo-point Placement. To place the pseudo-points in the environment, we adopt a grid-approach, and place a local frame $\mathcal{M}$ over the laser endpoints, as proposed by [26]. Since the laser beams are hit where there are obstacles, we can assume the endpoints of the laser beams have $g(x) = 0$. As illustrated in Fig.2, a 3x3 frame is placed over the laser endpoint, and the pseudo-points are selected from each point within the frame $\mathcal{M}$ such that pseudo-points of both positive and negative TSDF values are selected. Thus with the observations Z_t^i we locate the set of pseudo points $\tilde{\mathcal{P}}_t^i$.

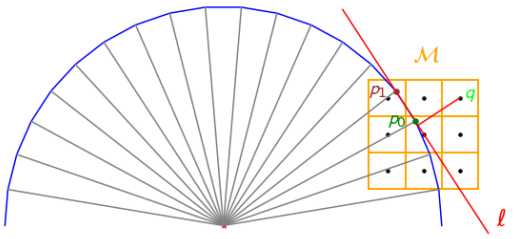


Fig. 2: Approximation of the TSDF values at a set of pseudo-points in 2D. The gray lines are LiDAR rays from the sensor. The blue line represents a surface boundary of the occupied space Ω. The green and brown points p_0, p_1 are hit locations on the boundary of two adjacent laser rays. Pseudo-points (black) are placed in a local frame $\mathcal{M}$ associated with point p_0. The TSDF of a pseudo-point q is approximated as the distance to the line l through the adjacent laser hit points.

A key observation is that for each robot i, $k_i = |\tilde{\mathcal{P}}_t^i|$ is much less than $n = |Z_t^i|$ as the robot team repeatedly explores the environment. k is controlled by the resolution of the environment, as pseudo-points are placed on the global map. n is controlled by the resolution, as well as the size of the frame placed on each laser hit point.

TSDF Approximation in 2D. For each pseudo-point $q \in \tilde{\mathcal{P}}_t^i$, to compute the TSDF corresponding to the recent observation Z_t^i, suppose that $p_1, p_2 \in X$ are two adjacent laser hit points, and $\vec{p_1}, \vec{p_2}$ denote the corresponding vectors starting from the origin O. We assume that p_1, p_2 belong to the same boundary surface $\partial\Omega_j$, and approximate the distance field of a pseudo-point $q \in \mathcal{M}(p_1)$ as the distance $d(q, \overline{p_1 p_2})$, where $\overline{p_1 p_2}$ denotes the line passing through $p_1 p_2$ in $2D$, or the plane containing p_1, p_2 in 3D. In 2D, $d(q, \overline{p_1 p_2})$ can be calculated using the following equation:

$$d(q, \overline{p_1 p_2}) = \frac{|(x_2 - x_1)(y_1 - y_0) - (x_1 - x_0)(y_2 - y_1)|}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \tag{9}$$

where $q = (x_0, y_0)$, $p_1 = (x_1, y_1)$, $p_2 = (x_2, y_2)$. We obtain $g(q) = \min(d(q, \overline{p_1 p_2}), h)$ if q is in the same halfspace of robot's position p_t, or $-\min(d(q, \overline{p_1 p_2}), h)$ otherwise.

Then we have a set of pseudo points $\tilde{\mathcal{P}}_t^i$ with their estimated TSDF values corresponding to observation Z_t^i. The transformed data are samples of g that are regressed with GP with mean $\mu(x)$ and covariance $k(x, x)$ defined in (8).

B. Distributed Updates in Time-Varying Network

To make the updates for the GP approximation scoped in the previous subsection executable in a distributed manner, we consider a setting where each robot acquires a local observation stream. Then, it must incorporate message passing with others in order to propagate its local information across the network and achieve a degree of coordination. Cooperation is defined from the perspective of whether individuals estimates approach that of a centralized estimator that theoretically has access to all robots' local information. In this work we show under Assumption 1, the GP parameters maintained by the individual robots converges to that of the central estimator.

In a distributed setting over a time-varying graph of robots, each individual platform i receives information from the neighboring platforms. For each robot i mixing the information from the neighbors $\mathcal{N}_t^i$ is usually achieved using a weighting scheme [17], [25]. Let W_t be the weighted adjacency matrix of the network at time t. We assume that $[W_t]_{ii} > 0$ for all $i \in \mathcal{V}$. Given the communication graph G_t, a weight matrix that is widely used in the distributed estimation literature is Metropolis weights [16] [25]:

$$[W_t]_{i,j} = \begin{cases} \frac{1}{(1+\max(\deg_t^i, \deg_t^j))}, & \text{if } (i,j) \in \mathcal{E}_t, \\ 0, & \text{if } (i,j) \notin \mathcal{E}_t, \\ 1 - \sum_{j \in \mathcal{N}_t^i} [W_t]_{i,j}, & \text{if } i = j, \end{cases} \tag{10}$$

where $deg_t^i = |\mathcal{N}_t^i|$ is the number of neighbors for robot i at t. As established in [25], an important property of the Metropolis weight matrix is that Assumption 1 ensures $\Pi_{t=\tau}^{\infty} W^t \to \frac{\mathbf{1}\mathbf{1}^\top}{n}$, i.e. the time-inhomogeneous Markov chain defined by the weights W_t has a uniform stationary distribution equal to $\mathbf{1}/n$.

Next, we introduce our method to update the robotic network. For each robot i at time t, we transform the observation $\{Z_t^i\}$ into a set of pseudo-point statistics, as explained in Sec.III-A. Using (9) for each new pseudo point $\mathbf{p} \in \tilde{\mathcal{P}}_t^i$ we average the TSDF values into $\tilde{\zeta}_t^i(\mathbf{p})$ and corresponding counting number of observation in $\tilde{m}_t^i(\mathbf{p})$. We also introduce a list ℓ_t^i containing the robots that received this message to enforce each robot receives each message exactly once. By combining all parameters, each observation $\tilde{\Theta}_t^i := \{\tilde{\mathcal{P}}_t^i, \tilde{m}_t^i(\tilde{\mathcal{P}}_t^i), \tilde{\zeta}_t^i(\tilde{\mathcal{P}}_t^i), \ell_t^i\}$ defines a message of observations for robot i observed at time t. In addition, $\tilde{B}_t^i$ and B_t^i define the set of messages robot i receives at t and retains at all time respectively. In practice, the distributed protocol would be mixed via weighting using (10); however, as $\Pi_{t=\tau}^{\infty} W^t$ converges to $\frac{\mathbf{1}\mathbf{1}^\top}{n}$, we define our distributed protocol based off the stationary distribution as the following:

$$\begin{aligned} \tilde{\mathcal{B}}_{t+1}^i &= \bigcup_{\tilde{\Theta}_\tau^j \in \mathcal{B}_t^r, r \in \mathcal{N}_t^i, i \notin \ell_\tau^j} \tilde{\Theta}_\tau^j \cup \tilde{\Theta}_{t+1}^i \\ \mathcal{B}_{t+1}^i &= \mathcal{B}_t^i \cup \tilde{\mathcal{B}}_{t+1}^i \\ \ell_\tau^j &= \ell_\tau^j \cup \{i\} \text{ for all } \tilde{\Theta}_\tau^j \in \tilde{\mathcal{B}}_{t+1}^i \\ \mathcal{P}_{t+1}^i &= \bigcup_{\tilde{\Theta}_\tau^j \in \tilde{\mathcal{B}}_{t+1}^i} \mathcal{P}_\tau^j \cup \mathcal{P}_t^i \end{aligned} \tag{11}$$

$$m_{t+1}^i(\mathbf{p}) = m_t^i(\mathbf{p}) + \sum_{\tilde{\Theta}_\tau^j \in \tilde{\mathcal{B}}_{t+1}^i} \frac{\tilde{m}_\tau^j(\mathbf{p})}{n}$$

$$\zeta_{t+1}^i(\mathbf{p}) = \frac{m_t^i(\mathbf{p})\zeta_t^i(\mathbf{p}) + \frac{1}{n}\sum_{\tilde{\Theta}_\tau^j \in \tilde{\mathcal{B}}_{t+1}^i} \tilde{m}_\tau^j(\mathbf{p})\tilde{\zeta}_\tau^j(\mathbf{p})}{m_{t+1}^i(\mathbf{p})}$$

where a message is dropped from robot i when ℓ_t^i is full.

To evaluate the convergence of our distributed protocol, we obtain the GP estimation of function g through a centralized estimator. A centralized estimator is defined as an imaginary robot that receives observations at each time t from all robots,

and the update of its GP parameters is defined as equal averaging of the statistics from all robots:

$$\begin{aligned}\mathcal{P}^{ctr}_{t+1} &= \cup_{i=1}^{n}\tilde{\mathcal{P}}^{i}_{t+1}\cup\mathcal{P}^{ctr}_{t},\\ m^{ctr}_{t+1}(\mathbf{p}) &= m^{ctr}_{t}(\mathbf{p})+\sum_{i=1}^{n}\frac{\tilde{m}^{i}_{t+1}(\mathbf{p})}{n},\\ \zeta^{ctr}_{t+1}(\mathbf{p}) &= \frac{m^{ctr}_{t}(\mathbf{p})\zeta^{ctr}_{t}(\mathbf{p})+\frac{1}{n}\sum_{i=1}^{n}\tilde{m}^{i}_{t+1}(\mathbf{p})\tilde{\zeta}^{i}_{t+1}(\mathbf{p})}{m^{ctr}_{t+1}(\mathbf{p})},\end{aligned} \tag{12}$$

Proposition 1. *For the data collected by the robot team by time $T<\infty$, at time $t=(\lceil\frac{T}{B}\rceil+(n-1))B$, the distributions $\mathcal{GP}(\mu^{i}_{t}(\mathbf{x}),k^{i}_{t}(\mathbf{x},\mathbf{x}'))$ maintained by each robot i, specified according to* (8) *with parameters in* (11) *are exactly equal to the distribution $\mathcal{GP}(\mu^{ctr}_{t}(\mathbf{x}),k^{ctr}_{t}(\mathbf{x},\mathbf{x}'))$ of the centralized estimator with parameters in* (12)*, i.e., $\mu^{i}_{t}(\mathbf{x})=\mu^{ctr}_{t}(\mathbf{x})$ and $k^{i}_{t}(\mathbf{x},\mathbf{x}')=k^{ctr}_{t}(\mathbf{x},\mathbf{x}')$ almost surely for all $i\in\mathcal{V}$, $\mathbf{x},\mathbf{x}'$.*

Proof. With regard to Eq (8), it is sufficient to show that at $t=(\lceil\frac{T}{B}\rceil+(n-1))B$, $m^{i}_{t}(\mathbf{p})=m^{ctr}_{t}(\mathbf{p})$ and $\zeta^{i}_{t}(\mathbf{p})=\zeta^{ctr}_{t}(\mathbf{p})$ for all $i\in\mathcal{V}$, $\mathbf{p}\in\mathcal{P}$. We express $m^{i}_{t}(\mathbf{p})$ and $\zeta^{i}_{t}(\mathbf{p})$ in terms of $\tilde{m}^{j}_{\tau}(\mathbf{p})$ and $\tilde{\zeta}^{j}_{\tau}(\mathbf{p})$ for arbitrary $\mathbf{p}\in\mathcal{P}$ and $\tau\le t$. The key is to check whether message $\tilde{\Theta}^{i}_{\tau}$ is received by robot j at time t. Since the message exchanges are happening based on the communication graph structure, the elements of $\Pi^{t}_{s=\tau}W_s$ determines which robots have received a message released at time τ by time t. Precisely, if $[\Pi^{t}_{s=\tau}W_s]_{ij}>0$ then robot i has received message $\tilde{\Theta}^{j}_{\tau}$ by time t and if $[\Pi^{t}_{s=\tau}W_s]_{ij}=0$ robot i has not received it. Let $sign(x)$ be the sign of a scalar x with $sign(0)=0$ and $sign(x)=1$ for $x>0$. Expanding (11) recursively leads to:

$$\begin{aligned}m^{i}_{t}(\mathbf{p}) &= \sum_{\tau=0}^{t}\sum_{i=1}^{n}sign([\Pi^{t}_{s=\tau}W_s]_{ij})\frac{\tilde{m}^{j}_{\tau}(\mathbf{p})}{n}\\ \zeta^{i}_{t}(\mathbf{p}) &= \frac{1}{m^{i}_{t}(\mathbf{p})}\sum_{\tau=0}^{t}\sum_{j=1}^{n}sign([\Pi^{t}_{s=\tau}W_s]_{ij})\frac{\tilde{m}^{j}_{\tau}(\mathbf{p})}{n}\tilde{\zeta}^{j}_{\tau}(\mathbf{p})\end{aligned} \tag{13}$$

Next we will show under Assumption 1, a message produced by robot j at time τ will be received by all robots in $(n-1)B$ steps. At time step t, let A_t be set of all robots who received the message produced by robot j at time step τ, and $\bar{A}_t$ be the set of the rest of robots who did not received it. At first $A_\tau=\{j\}$ and rest of robots are in $\bar{A}_\tau$ including i. Note that Assumption 1 means that in each B steps there is a time when a edge connects A_t and $\bar{A}_t$. At such time, we can consider the vertex of this edge in $\bar{A}_t$, remove it and add it to A_t. Since in the beginning there is $n-1$ robots in $\bar{A}_\tau$, at some point in $(n-1)B$ steps robot i will be added to A_t, and hence $sign([\Pi^{(n-1)B+\tau}_{s=\tau}W_s]_{ij})=1$, $\forall i,j\in V$.

For the pseudo-points $\mathbf{p}$ not observed by robot i we assume $\tilde{m}^{i}_{\tau}(\mathbf{p})=\tilde{\zeta}^{i}_{\tau}(\mathbf{p})=0$. Let us consider the non-zero terms $\frac{\tilde{m}^{i}_{\tau}(\mathbf{p})}{n}$, $\frac{\tilde{m}^{i}_{\tau}(\mathbf{p})}{n}\tilde{\zeta}^{i}_{\tau}(\mathbf{p})$, corresponding to observation by robot i at time step τ. It includes pseudo points $\mathbf{p}$ such that $\tilde{m}^{i}_{\tau}(\mathbf{p})>0$. We also have shown that the statistics in the corresponding message is observed by $t=(n-1)B+\tau$. Comparing (13) and (12) concludes the equality $\mu^{i}_{t}(\mathbf{x})=\mu^{ctr}_{t}(\mathbf{x})$ and $k^{i}_{t}(\mathbf{x},\mathbf{x}')=k^{ctr}_{t}(\mathbf{x},\mathbf{x}')$ at $t=(\lceil\frac{T}{B}\rceil+(n-1))B$. □

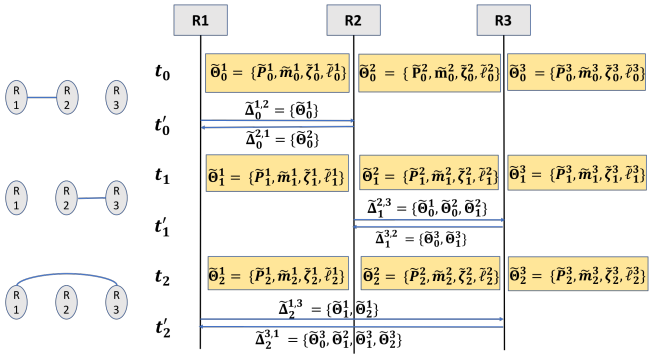


Fig. 3: Diagram for message passing among 3 agents with a time-varying communication graph G_t. The yellow box denotes the event when a new batch of pseudo-point and TSDF is observed, and a new message is constructed (Line 1-2 of Algorithm 1). At each step t new observations are collected, and at t' messages are sent. (Line 4-5).

Synthesizing Sec.III-A and the distributed protocol proposed in (11), we put forth the following algorithm for distributed update of pseudo-points statistics of the TSDF. An illustration for the message passing between the agents is in Fig 3. The GP parameters $\mathcal{GP}^{i}_{t+1}$ for the corresponding QuadTree Node Tr^{i}_{t+1} is updated according to (11) (Line 10 of Algorithm 1). When merging the pseudo-points statistics from the neighbors, if the QuadTree Node Tr^{i}_{t+1} has number of pseudo-points larger than *maxLeafSize*, the node including its pseudo-point statistics is split recursively (Line 10).

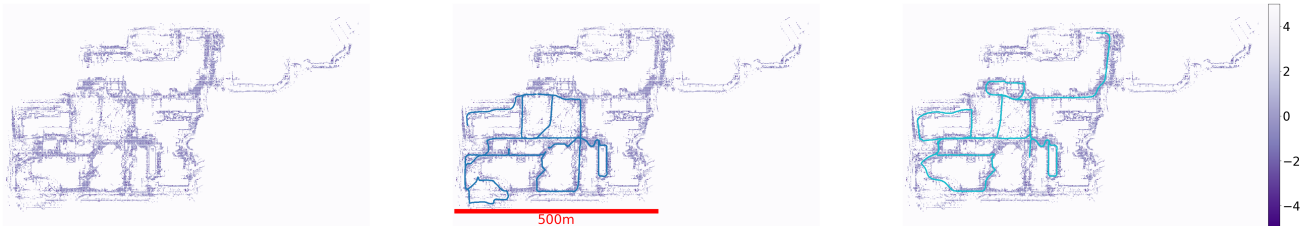
Algorithm 1 Distributed Mapping for robot i

Input: new local observations Z^{i}_{t+1}, robot position p^{i}_{t+1}
Octree Tr^{i}_{t}, $\mathcal{GP}^{i}_{t}$, Neighbors $\mathcal{N}^{i}_{t+1}$
Output: Octree Tr^{i}_{t+1}, $\mathcal{GP}^{i}_{t+1}$

1: $\tilde{P}^{i}_{t+1},\tilde{\zeta}^{i}_{t+1},\tilde{m}^{i}_{t+1}\leftarrow$ computePseudoPoints$(p^{i}_{t+1},Z^{i}_{t+1})$
2: $\tilde{\Theta}^{i}_{t+1}\leftarrow\{\tilde{\mathcal{P}}^{i}_{t+1},\tilde{m}^{i}_{t+1}(\tilde{\mathcal{P}}^{i}_{t+1}),\tilde{\zeta}^{i}_{t+1}(\tilde{\mathcal{P}}^{i}_{t+1}),\ell^{i}_{t+1}\}$
3: **for** $j\in\mathcal{N}^{i}_{t+1}$ **do**
4: $\quad\tilde{\Delta}^{i,j}_{t+1}=\tilde{\Theta}^{i}_{t+1}\bigcup_{\tilde{\Theta}^{k}_{\tau}\in\mathcal{B}^{i}_{t},j\notin l^{k}_{t}}\tilde{\Theta}^{k}_{\tau}$
5: $\quad$sendToNeighbor$(\tilde{\Delta}^{i,j}_{t+1})$
6: **end for**
7: $\bar{P}^{i}_{t+1}\leftarrow\tilde{P}^{i}_{t+1}\bigcup_{j\in\mathcal{N}^{i}_{t+1},\tilde{\Theta}^{k}_{\tau}\in\tilde{\Delta}^{j,i}_{t}}\tilde{P}^{k}_{\tau}$
8: $Tr^{i}_{t+1}=Tr^{i}_{t}$
9: **for** $p^{i}_{t+1}\in\bar{P}^{i}_{t+1}$ **do**
10: $\quad Tr^{i}_{t+1}$.insertAndSplit$(p^{i}_{t+1},\bar{\zeta}^{i}_{t+1}(p_{t+1}),\bar{m}^{i}_{t+1}(p_{t+1}))$
11: **end for**

IV. Experimental Results

We evaluate our algorithm on two public LiDAR datasets. The first dataset is from the Robotics Data Set Repository (Radish) [27], which contains LiDAR sequences and odometry recorded from different environments. The second dataset is the North Campus Long-Term (NCLT) dataset [1], which contains multiple long sessions for a single robot navigating on University of Michigan's campus. We show that using



(A) Centralized Estimator (B) Agent 1 (C) Agent 10

Fig. 4: TSDF estimates from the North-Campus Long-Term dataset [1].

the time-varying distributed mapping algorithm (Algorithm 1) the estimation of the *TSDF* by the distributed team of robots converge to that of the centralized estimator, both qualitatively and quantitatively.

To evaluate the accuracy of the TSDF mapping, we evaluate the predictions of each individual agent with respect to the prediction of the centralized estimator according to the following metric:

$$RMSE^i = \sqrt{\frac{\sum_{p\in\mathcal{X},\hat{z}(p)\in(-h,h)}(\hat{y}^i(p)-\hat{z}(p))^2}{\sum_{p\in X}\mathbb{1}(\hat{z}(p)\in(-h,h))}} \tag{14}$$

where $\hat{z}(p)$ is the prediction of the centralized estimator, h is the truncated distance, $\hat{y}^i(p)$ is the prediction of the ith agent, and $\mathcal{X}$ denotes the entire environment. To simulate the neighborhood $\mathcal{N}_t^i$ of the robot i, we compute the distance between all pairs of robots at all t, and connect two robots if the distance is lower than a threshold r.

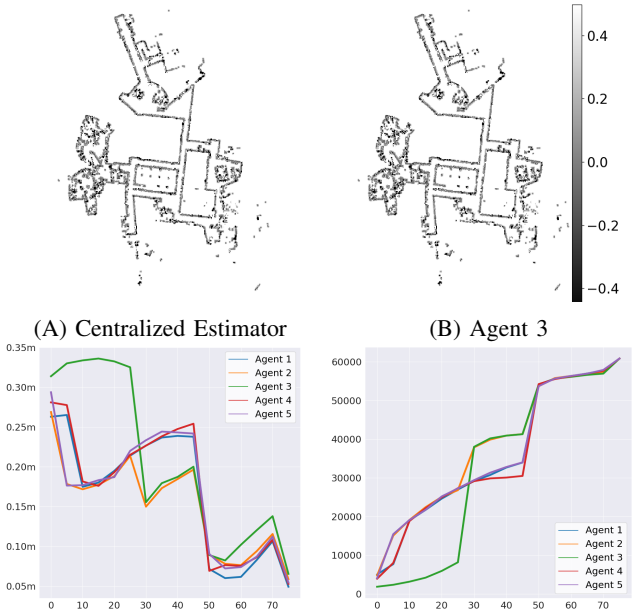
A. Radish Datasets

For the Radish [27] datasets, we selected the following three environments: Intel Research Lab, Orebro and MIT CSAIL. A single sequence was recorded for each environment, and we divide each sequence into five disjoint subsequences with equal length, and label them $1\ldots5$.

For each of the dataset, we set $h = 0.5$, and we assumed initially the entire environment to be in free space, i.e. $\mu_0 = 0.5$. $r = 20m$ is selected as the communication range based on the size of the environments. We used a grid size of 0.1 for all agents in each dataset, and used the following parameters for the Gaussian Process: $c = 1.0$, $l = 0.1$, $\sigma = 0.1$. In addition, we bound the max size of pseudo-points in each Quadtree leaf node to be 50 for efficient insertion and update of the pseudo-points. In Figure 5, we show the *TSDF* map of the central estimator and agent 3 on MIT CSAIL, as well as the estimated error with respect to the centralized estimator. We refer the readers to Appendix I in [28] for a complete list of our results of the agents, as well as on the other datasets.

B. NCLT Dataset

The North Campus Long Term Dataset [1] is a large-scale dataset both spatially and temporally. It covers roughly a square kilometer, includes challenging weather conditions and moving objects, and is collected over the time span of serveral months. The dataset provides 2D and 3D Lidar scans of the robot, camera images, odometry readings, as well as GPS positions of the robot. In our evaluation, we used the



(A) Centralized Estimator (B) Agent 3

(C) RMSE (D) Number of Pseudo-Points

Fig. 5: TSDF map and metrics for the MIT-CSAIL dataset from the Radish repository [27]. As the robots explore new parts of the environment, the centralized estimator accumulates more information than the individual robots because some newly acquired information is not passed to all robots instantaneously. Hence, the RMSE goes up for some periods of time. However, over the long-term run as information propagates throughout the network the RMSE decreases.

GPS readings as the robot's trajectory, and 2D Lidar scans as observations. We selected 10 sequences from different dates where a different trajectory and observations are collected. For the Gaussian Process, the following GP parameters are used: $\sigma = 0.1, c = 1.0, l = 0.2$. A grid size of 0.25 is used as the space covered is large-scale. We use 5 meters as the truncation value for TSDF, i.e. $\mu_0 = h = 5$.

We show the TSDF estimates of the centralized estimator, and agent 1 and 10 in Fig. 4. The RMSE and number of pseudo-points are shown in Fig. 6, with communication range of $100m$. The results for the rest of the agents are similar, and are included in Appendix II in [28]. When the robots are in isolated environments, the error with respect to the centralized estimator stagnated. However, when the robots are within communication range and exchange information, the pseudo-point statistics are exchanged, and the number of pseudo-points increase sharply, while the error

(A) RMSE (B) Number of Pseudo-Points

Fig. 6: Metrics over time for the NCLT dataset [1].

TABLE I: Distributed TSDF mapping performance for different communication range values. Each table entry shows the mean and standard deviation computed over the 10 agents at the end of the run on the NCLT dataset.

Range (m)	RMSE (m)	# Pseudo-points	# Leaves
50	1.65 ± 0.17	627684 ± 26247	31372 ± 1036
100	0.74 ± 0.09	712459 ± 4777	34837 ± 166
200	0.10 ± 0.02	719725 ± 22	35056 ± 3.56

drops sharply at these rendezvous. During some parts of the run the RMSE increases because the robot team continues to explore the environment, and may not have neighbors to exchange information. However, over the time-span T the TSDF estimate of each individual agent continues getting closer to that of the centralized estimator asymptotically.

In Table I, we show the statistics corresponding to different values of r. Larger value of r corresponds to larger communication range, and hence higher likelihood a larger subset of agents to exchange information with at each time t. As r increases, the mean and std of the RMSE both decrease at the end of the run. The number of pseudo-points and number of leaves both increase, as each robot has received more pseudo-points from a larger number of robots.

V. Conclusion

We developed a distributed, probabilistic, online and efficient algorithm for TSDF mapping using a robot team with time-varying communication. We provided a theoretical guarantee that the TSDF probability distributions maintained by individual agents converge to a common distribution, exactly equal to that obtained by centralized estimation. Our approach showed excellent performance in large-scale evaluation on two real-world datasets. Future work will consider incorporating robot pose estimation to achieve a fully distributed multi-agent SLAM algorithm. We also plan to utilize the covariance of the GP estimates for down-stream tasks, such as collision avoidance and motion planning.

References

[1] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of michigan north campus long-term vision and lidar dataset," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2016.

[2] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[3] S. Thrun, "Simultaneous localization and mapping," in *Robotics and cognitive approaches to spatial mapping*, pp. 13–41, Springer, 2007.

[4] P.-Y. 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.

[5] P. Schmuck and M. Chli, "Ccm-slam: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams," *Journal of Field Robotics*, vol. 36, no. 4, pp. 763–781, 2019.

[6] K. Alexis, "Resilient autonomous exploration and mapping of underground mines using aerial robots," in *2019 19th International Conference on Advanced Robotics (ICAR)*, pp. 1–8, IEEE, 2019.

[7] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *2011 10th IEEE international symposium on mixed and augmented reality*, pp. 127–136, IEEE, 2011.

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

[9] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, no. 3, pp. 189–206, 2013.

[10] 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.

[11] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer school on machine learning*, pp. 63–71, Springer, 2003.

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

[13] A. Koppel, H. Pradhan, and K. Rajawat, "Consistent online gaussian process regression without the sample complexity bottleneck," *Statistics and Computing*, vol. 31, no. 6, pp. 1–18, 2021.

[14] E. Snelson and Z. Ghahramani, "Sparse gaussian processes using pseudo-inputs," *Advances in neural information processing systems*, vol. 18, p. 1257, 2006.

[15] D. Burt, C. E. Rasmussen, and M. Van Der Wilk, "Rates of convergence for sparse variational gaussian process regression," in *International Conference on Machine Learning*, pp. 862–871, PMLR, 2019.

[16] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Gossip algorithms: Design, analysis and applications," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 3, pp. 1653–1664, IEEE, 2005.

[17] 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.

[18] A. Koppel, F. Y. Jakubiec, and A. Ribeiro, "A saddle point algorithm for networked online convex optimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 19, pp. 5149–5164, 2015.

[19] H. Terelius, U. Topcu, and R. M. Murray, "Decentralized multi-agent optimization via dual decomposition," *IFAC proceedings volumes*, vol. 44, no. 1, pp. 11245–11251, 2011.

[20] S. Boyd, N. Parikh, and E. Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[21] E. Zobeidi, A. Koppel, and N. Atanasov, "Dense incremental metric-semantic mapping for multi-agent systems via sparse gaussian process regression," *arXiv preprint arXiv:2103.16170*, 2021.

[22] A. Censi, "An ICP variant using a point-to-line metric," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[23] F. Dellaert and M. Kaess, *Factor graphs for robot perception*. Now Publishers, Inc., 2017.

[24] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 601–615, 2015.

[25] A. Nedić, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2597–2633, 2017.

[26] E. Zobeidi, A. Koppel, and N. Atanasov, "Dense incremental metric-semantic mapping via sparse gaussian process regression," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6180–6187, IEEE.

[27] A. Howard and N. Roy, "The robotics data set repository (radish)," 2003.

[28] J. Di, E. Zobeidi, A. Koppel, and N. Atanasov, "Technical Report: Distributed Gaussian Process Mapping for Robot Teams with Time-varying Communication," 2021.