Exploration, Mapping and Scalar Field Estimation using a Swarm of Resource-Constrained Robots

by

Ragesh Kumar Ramachandran

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved August 2018 by the
Graduate Supervisory Committee:

Spring Berman, Chair
Marc Mignolet
Panagiotis Artemiadis
Hamid Marvi
Michael Robinson

ARIZONA STATE UNIVERSITY

December 2018

ABSTRACT

Robotic swarms can potentially perform complicated tasks such as exploration and mapping at large space and time scales in a parallel and robust fashion. This thesis presents strategies for mapping environmental features of interest – specifically obstacles, collision-free paths, generating a metric map and estimating scalar density fields– in an unknown domain using data obtained by a swarm of resource-constrained robots. First, an approach was developed for mapping a single obstacle using a swarm of point-mass robots with both directed and random motion. The swarm population dynamics are modeled by a set of advection-diffusion-reaction partial differential equations (PDEs) in which a spatially-dependent indicator function marks the presence or absence of the obstacle in the domain. The indicator function is estimated by solving an optimization problem with PDEs as constraints. Second, a methodology for constructing a topological map of an unknown environment was proposed, which indicates collision-free paths for navigation, from data collected by a swarm of finite-sized robots. As an initial step, the number of topological features in the domain was quantified by applying tools from *algebraic topology*, to a probability function over the explored region that indicates the presence of obstacles. A topological map of the domain is then generated using a graph-based wave propagation algorithm. This approach is further extended, enabling the technique to construct a metric map of an unknown domain with obstacles using uncertain position data collected by a swarm of resource-constrained robots, filtered using intensity measurements of an external signal. Next, a distributed method was developed to construct the occupancy grid map of an unknown environment using a swarm of inexpensive robots or mobile sensors with limited communication. In addition to this, an exploration strategy which combines information theoretic ideas with Lévy walks was also proposed. Finally,

the problem of reconstructing a two-dimensional scalar field using observations from a subset of a sensor network in which each node communicates its local measurements to its neighboring nodes was addressed. This problem reduces to estimating the initial condition of a large interconnected system with first-order linear dynamics, which can be solved as an optimization problem.

To Achan, Amma, Preethi and Roopesh

ACKNOWLEDGMENTS

I would first like to thank my advisor Professor Spring Berman whose hard work, patience, and guidance has provided me with opportunities and experiences that have pushed me to excel during this process. I would also like to thank Prof. Marc Mignolet, Prof. Panagiotis Artemiadis, and Prof. Hamid Marvi for teaching me inside and outside the classroom, serving on my dissertation committee, and providing me valuable feedback that improved the work presented in this thesis. A special thanks go to Prof. Michael Robinson for his valuable inputs and serving on my dissertation committee in spite of the distance and his medical condition.

I would like to thank everyone in the Mechanical and Aerospace Engineering(MAE) department and acknowledge all the staff members who have helped me with my work. There are no words to express my sincere gratitude and respect towards MAE graduate chair Prof. Marc Mignolet for helping me overcome various crisis while at ASU.

My special thanks goes to the Wikimedia Foundation for providing an easy and quick reference to a vast amount of knowledge through Wikipedia. I would also like to thank Dr. N. J. Wildberger, UNSW, for his online video lectures on algebraic topology.

I would like to credit my roommates old and new who have provided me with chances to relax and step away from the stress caused during graduate work. I'd like to thank all my lab mates who made working in the lab extremely entertaining and who have had conversations with me that have spawned great and terrible ideas. Special thanks to my friend and colleague Karthik for the long discussions we had. These discussions had not only helped improve the dissertation considerably but also shaped my outlook towards mathematics.

I am grateful to my high school computer science teacher Gomathi, who has

helped me strengthen my computer programming basics. She is always a source of inspiration and has left a lasting impression on me.

I am indebted to my parents Ramachandran and Geetha, for their love and care. Many good times spent were with my brother Rajesh. His company has been a pleasant break from my work. Finally, a special thanks goes to my wife Aishwarya for her understanding, love, and patience over the years of my graduate school life no matter the distance.

कर्मण्येवाधिकारस्ते मा फलेषु कदाचन।
मा कर्मफलहेतुर्भूर्मा ते सङ्गोऽस्त्वकर्मणि॥

Karmanye vadhikaraste Ma Phaleshu Kadachana,
Ma Karmaphalaheturbhurma Te Sangostvakarmani

You have the right to work only, never to its fruits.
Let not the fruits of action be your motive,
nor let your attachment be to inaction.

— **Bhagavad gita**

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

Chapter 1

INTRODUCTION

1.1 Swarm Robotics

In recent years, there has been an increasing focus on the development of robot platforms that can be deployed in *swarms* to perform tasks autonomously over large spatial and temporal scales. In fact, robotic swarms have already become a reality with large numbers of robots being deployed to perform desired tasks. The Kiva System uses hundreds to thousands of autonomous robots to manage storage warehouses [30]. In addition, swarms of nanoscale structures and devices such as nanoparticles, molecular machines, and magnetic nanocarriers are being developed for biomedical applications such as imaging and targeted drug delivery [143]. Many potential applications for robotic swarms, including exploration, environmental monitoring, disaster response, search-and-rescue, mining, and intelligence-surveillance-reconnaissance will require the robots to operate in dynamic, uncertain environments. Moreover, the robots' highly restricted onboard power may preclude the use of GPS and communication devices, or the robots may be located in GPS-denied environments where communication is impractical or unreliable. Despite these limitations, it may still be necessary for the swarm to characterize its surroundings; for instance, to map obstacles, payloads for transport, or hazardous areas to avoid. Nanoscale swarms, which will have extremely limited capabilities, may be used to map cellular structures inside the human body.

A major challenge in designing any kind of control strategies for swarm robotic systems is to make the strategies scale with the number of agents, i.e., the com-

plexity of the strategy should not increase considerably for a larger number of robots. This challenge has motivated researchers to develop scalable strategy design frameworks for swarm robotic systems. In 2004, Erol Şahin proposed the following definition of swarm robotics:

Definition 1.1.1. *Swarm robotics is the study of how a large number of relatively simple physically embodied agents can be designed such that a desired collective behavior emerges from the local interactions among agents and between the agents and the environment [118].*

Şahin's criterion in [118] has motivated many researchers to develop scalable control strategies for robotic swarms. [13, 36, 94, 95]. Swarm robotic systems are capable of executing tasks in parallel and adapt to their surroundings; enabling them to solve problems in large, unknown, possibly hazardous environments. The thesis also introduces and solves the problem of reconstructing a two-dimensional scalar field using measurements from a subset of a network with local communication between nodes in a sensor network.

1.2 Robotic Mapping Techniques

A common task involved in most of the robotic applications is generating a map of the environment where the swarm of robots operate. Size and cost constraints limit individual robots in the swarm from having sufficient resources to map the entire unknown environment using existing mapping techniques collectively known as simultaneous localization and mapping (SLAM) [142]. SLAM is the process of mapping an unknown domain and constructing a map consistent with the domain through an appropriate fusion of robot's sensory data. Although numerous algorithms for SLAM have been proposed, in general, all these algorithms can be

categorized into the following main categories:

Feature-based mapping: [126] also known as *landmark based mapping* is a method in which the environment is represented using a list of global positions of various features or landmarks present in the environment. Consequently, the algorithms in this category require feature extraction and data association.

Occupancy grid mapping uses an array of cells to represent the unknown environment. This class of algorithms was first introduced in Elfes et al. [44] and is the most commonly used method in robotics mapping applications. Occupancy grid maps are very effective in representing 2D environment, but it suffers from the curse of dimensionality. The cells in an occupancy grid map are modeled as binary random variables that give probability of occupancy of a cell by an object.

Topological mapping [25] procedures generate a *topological map* which is a compact sparse representation of an environment. A topological map encodes all of an environment's topological features such as holes that signify the presence of obstacles; and provides a collision-free path through the environment in the form of a roadmap. A topological map is generally a graph in which the vertices correspond to particular obstacle-free locations in the domain and the edges correspond to collision-free paths between these locations. Figure 1.1 shows an example of a topological map in the form of a Generalized Voronoi Diagram (GVD) .

Chapter 3, Chapter 4, Chapter 5, Chapter 6 and Chapter 7 in this thesis presents various scalable strategies for mapping and exploring unknown environments using a swarm of resource-constrained robots. The thesis primarily focuses on

generating occupancy grid maps and topological maps in a scalable fashion. In Chapter 7, a novel exploration strategy for exploration of unknown environments by a robotic swarm is presented. The exploration technique combines information theoretic concepts with Lévy walks.

1.3 Scalar field estimation

Large networks of robots or sensors can perform a range of distributed sensing and estimation tasks such as environmental monitoring, field surveillance and reconstruction, multi-target tracking, and geo-scientific exploration [68, 4]. Applications of sensor networks which require scalar field estimation are so wide and important that it is essential to consider them as a problem to be solved in itself. The environment to be sampled by the network may be remote or hazardous, allowing measurements to be directly accessed from only a subset of the robots/sensors at any given time. A common characteristic of most scalar fields is that the sensor measurements are valid only locally. In other words, the correlation in measurements between sensor nodes is inversely proportional to the distance between them [161]. In general, estimation of scalar fields where no sensor nodes are deployed is performed by interpolation measurements from the neighboring sensor nodes. The estimation accuracy can be further improved if the

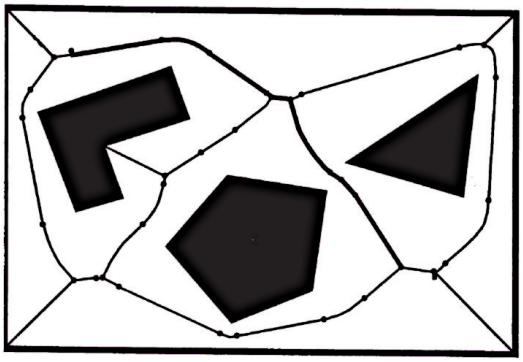


Figure 1.1: Image from [59]. The figure shows the topological map of a domain with three obstacles (the solid black regions). The black lines represent the topological map, constructed as a Generalized Voronoi Diagram.

sensors shared their measurements with their neighbors. The important question to ask is: "what network topology of the sensor network would increase the accuracy of the estimation of a scalar field?". The primary motivation of this work is to quantify the fundamental performance limitations that emerge in these scenarios due to the chosen *inter-robot/sensor communication topology* of the network. This topology can be implemented in stationary networks through the configuration of the robots/sensors. Also it can be enforced in mobile networks using strategies such as formation control [31] or in traffic control for platoons [6].This thesis not only solves the problem of scalar field estimation by posing it as an optimization problem with linear network dynamics as constraints but also proposes a metric to measure the scalar field estimation capability of a given sensor network.

1.4 Contributions of the thesis

The novel contributions of this work can be summarized as follows:

- A partial differential equation (PDEs) based approach to mapping a region of interest using observations from a robotic swarm without localization was introduced. The robots have local sensing capabilities, no communication, and they exhibit stochasticity in their motion. The swarm population dynamics is modeled with a set of advection-diffusion-reaction partial differential equations. The map of the environment is incorporated into this model using a spatially-dependent indicator function that marks the presence or absence of the region of interest throughout the domain. To estimate this indicator function, the solution of an optimization problem is defined in which we minimize an objective functional that is based on temporal robot data. The optimization is performed numerically offline using a standard gradient descent algorithm. The approach is validated through simulations.

- A novel automated approach to quantifying the topological features of an unknown environment using a swarm of robots with local sensing and limited or no access to global position information was proposed. The robots randomly explore the environment and record a time series of their estimated position and the covariance matrix associated with this estimate. After the robots' deployment, a point cloud indicating the free space of the environment is extracted from their aggregated data. Tools from topological data analysis, in particular the concept of persistent homology, are applied to a subset of the point cloud to construct barcode diagrams, which are used to determine the numbers of different types of features in the domain. It is demonstrated that the approach can correctly identify the number of topological features in simulations with zero to four features and in multi-robot experiments with one to three features.

- A novel procedure is presented for constructing a topological map of an unknown environment from data collected by a swarm of robots with limited sensing capabilities and no communication or global localization. Topological maps are sparse roadmap representations of environments that can be used to identify collision-free trajectories for robots to navigate through a domain. Like the previous contribution, this method also uses uncertain position data obtained by robots during the course of random exploration to construct a probability function over the explored region that indicates the presence of obstacles. Techniques from topological data analysis, in particular the concept of persistent homology, are again applied in a different manner to the probability map to segment the obstacle regions. Finally, a graph-based wave propagation algorithm is applied to the obstacle-free region to construct the

topological map of the domain in the form of an approximate Generalized Voronoi Diagram. The effectiveness of the approach is demonstrated in a variety of simulated domains and in multi-robot experiments on a domain with two obstacles, and an analysis of its computational time complexity is also performed.

- The procedure mentioned in the previous contribution was extended to construct a metric map of an unknown domain with obstacles using uncertain position data collected by a swarm of resource-constrained robots. The robots obtain this data during random exploration of the domain by combining onboard odometry information with noisy measurements of signals received from transmitters located outside the domain. This data is processed offline to compute a density function of the free space over a discretization of the domain. Persistent homology techniques from topological data analysis are used to estimate a value for thresholding the density function, thereby segmenting the obstacle-occupied region in the unknown domain. The approach is substantiated with theoretical results to prove its completeness and to analyze its time complexity. The effectiveness of the procedure is illustrated with numerical simulations conducted on six different domains, each with two signal transmitters.

- A novel distributed method to construct occupancy grid map of an unknown environment using a swarm of inexpensive robots or mobile sensors with limited communication was proposed. A team of robots that explore an unknown domain and create local maps based on robots' laser range sensor measurement was considered. Simultaneously, each robot updates its local occupancy grid map using its laser range sensor measurements and using

map information broadcasted by robots residing in the robot's neighborhood. Thus each robot's occupancy grid map eventually converges to the global map of the unknown environment. Under reasonable assumptions on the connectivity of robot interaction graph, the distributed mapping strategy is proved to converge each robot's occupancy grid asymptotically to the global map. An exploration strategy which combines information theoretic ideas with Lévy walks was also proposed. In addition, it was demonstrated that a *topological data analysis* technique developed as part of earlier contribution for generating topological maps can be easily extended for adaptive thresholding of occupancy grid maps. The effectiveness of the distributed mapping and exploration strategy was validated through a series of numerical simulations and experiments.

- Formulated the problem of reconstructing a two-dimensional scalar field using measurements from a subset of a network with local communication between nodes as an optimization problem that is constrained by first-order linear dynamics on a large interconnected system. A gradient descent approach was used to solve the optimization problem. The gradient of the objective function for the optimization problem was derived analytically. Bounds on the trace of the observability Gramian of the large interconnected system was also derived, this can be used to quantify and compare the field estimation capabilities of any undirected networks. The derived bounds were used to compare the field estimation capabilities of chain and grid networks. A comparison based on a performance measure related to the $\mathcal{H}_2$ norm of the system is also used to study the robustness of the network topologies. The results are validated in simulation using both Gaussian scalar fields and

actual ocean salinity data.

1.5 Organization of the thesis

This section outlines the contents described in various chapters of this report. Each chapter from Chapter 3 to Chapter 8 in the thesis talk about separate problem statements and literature review associated with each problem statement is detailed in the corresponding chapter. It is to be noted that the notations used in each chapter are local to the chapter and are not be confused with notations used in other chapters. The reminder of the dissertation is organized as follows: Chapter 2 provides a concise mathematical background for in-depth understanding of the various techniques used in this work. Chapter 3 proposes an optimal control of partial differential equation approach for mapping a feature of interest in an unknown environment using a swarm of resources constrained robots. A probabilistic topological technique to quantify the number of features in an unknown environment is introduced in Chapter 4. The technique uses uncertain position data obtained from robots that explored the unknown domain using random walk. Tools from algebraic topology were used to quantify the number of features from the uncertain position data. Chapter 5 extends the approach presented in Chapter 4 to generate a topological map of the domain using the uncertain position data. The approach is further extended in Chapter 6 to generate a metric map of the explored environment. Position data from robots with bounded uncertainty were used for metric map generation of the environment. It is also proven in Chapter 6 that a metric map of the environment can also be constructed from the position data of robots as long as its uncertainty is bounded. A distributed mapping approach for mapping an unknown environment using a swarm of robots is detailed in Chapter 7. The chapter also describes a novel exploration strategy for robotic swarms which

combines information theoretic concepts with Lévy walks. Chapter 8 investigates the problem of reconstructing a two-dimensional scalar field using measurements from a subset of a network with local communication between nodes. Finally, conclusions and future works are provided in Chapter 9.

Chapter 2

BACKGROUND

In this chapter, the basic concepts of optimization theory, topology and information theory are introduced. The material presented in this chapter does not constitute any novel research contribution by this dissertation.

2.1 Topology

This section outlines some basic concepts from the mathematical field known as topology. The topics described in this section provide the necessary background to understand the topological techniques presented in Chapter 4, Chapter 5, Chapter 6 and Chapter 7.

2.1.1 Point-set Topology

Definition 2.1.1. *A set is a collection of objects. Mathematical analysis mostly deals with sets having infinite uncountable number of objects. For example, the set of real numbers.*

However, set theory alone does not provide any information about the relationship between elements in a set. In other words, a set is just a collection of elements and does not provide information like whether a particular element in the set is closer to the second element in comparison to another element in the set. This is where the notion of topology comes to the rescue. A *topology* of a set gives the set an additional structure which helps to define the notion of closeness of elements in the set.

Definition 2.1.2. *Topology [91]. A topology on a set* $\mathbf{X}$ *is a collection,* $\mathbf{T}$*, of subsets of* $\mathbf{X}$*, containing both* $\mathbf{X}$ *and* $\emptyset$*, and closed under the operations of intersection and union.*

The tuple $(\mathbf{X}, \mathbf{T})$ *is referred as a topological space, open sets of a topological space are the elements of* $\mathbf{T}$.

The concept topology also helps in defining the notion of *continuity* of a function between sets.

Definition 2.1.3. *Continuity of function between topological spaces [91]. Let* $f : \mathbf{X}_1 \to \mathbf{X}_1$ *be a function between the topological spaces,* $(\mathbf{X}_1, \mathbf{T}_1)$ *and* $(\mathbf{X}_2, \mathbf{T}_2)$*. Then* f *is a continuous function if for every open set* $V \in \mathbf{T}_2$*, the set* $f^{-1}(V) \in \mathbf{T}_2$

Using these basic definitions, one can construct a topological space from another. The same idea can also be used to establish relationships between topological space such as equivalence relations and embedding. This is the primary goal of *point-set topology*. Sometimes the relationship can be gauged much more easily by performing a little algebra. This is where *algebraic topology* gets introduced.

Now some fundamental equivalence relations among topological space can be introduced. The first to introduce is the notion called *Homeomorphism* . Topologically speaking, two topological spaces are homeomorphic if they basically the same topological space. Informally, two topological spaces are homeomorphic if one can be continuously deformed into the other without causing cuts or tears in the space. In other words, under a homeomorphic map open sets remain as open sets.

Definition 2.1.4. *Homeomorphism [91]. Let* $\mathbf{X}_1$ *and* $\mathbf{X}_1$ *be two topological spaces. Then these spaces are said to homeomorphic, if there exists a bijective function such that both the function and its inverse are continuous. The function is referred as a homeomorphism between the spaces.*

Homotopy is a fundamental equivalence relation that exists among continuous functions defined between two fixed topological spaces.

Definition 2.1.5. *Homotopy [60]. Let $f_1, f_2 : \mathbf{X}_1 \to \mathbf{X}_1$ be continuous functions between the topological space $\mathbf{X}_1$ and $\mathbf{X}_2$. If there exists a continuous function $\mathbf{F} : \mathbf{X}_1 \times [0,1] \to \mathbf{X}_1$ such that $F(x_1, 0) = f_1(x_1)$ and $F(x_1, 1) = f_2(x_1), \forall x_1 \in \mathbf{X}_1$ then the functions f_1 and f_2 are homotopic. The function F is called a homotopy between the functions.*

An interesting question to ask is: given a topological space **X** and a subspace **Y** (subspace is a subset endowed with *subspace topology*), can the space **X** be continuously deformed to its subspace **Y** without tearing or cutting. This question motives the idea of a *deformation retract*.

Definition 2.1.6. *Deformation Retract [60, 26]. A subspace **Y** is called a deformation retract of a space **X** if there exists a continuous map $D : \mathbf{X} \times [0,1] \to \mathbf{X}$ such that*

1. $D(x, 0) = x, \ \forall x \in \mathbf{X}$
2. $D(y, t) = y, \ \forall y \in \mathbf{Y}, t \in [0,1]$, *and*
3. $D(x, 1) \in \mathbf{Y}, \ \forall x \in \mathbf{X}$.

*The map F becomes the deformation retraction from **X** to **Y**. F can also be interpreted as a homotopy between the identity map and the map $F(\cdot, 1)$ whose image is in **Y**. This is because **Y** is a subspace of **X**.*

The idea of deformation retract requires that the topological space **Y** is a subspace of **X**. This concept can be generalized to define equivalence relation between arbitrary spaces and generalization is called *homotopy equivalence*. Intuitively speaking, the basic idea behind homotopy equivalence is that instead of explicitly referring a subspace **Y** of **X**, continuous identity functions of **X** through a second space **Z** is examined. If this combination map is homotopic to the identity map on **X**. Also consider the same process with the role of **X** and **Z** interchanged. If the answer

is positive in both scenarios then the topological spaces **X** and **Z** are homotopy equivalents.

Definition 2.1.7. *Homotopy Equivalence [60]. Two topological spaces* **A** *and* **B** *are homotopy equivalent if there exists continuous functions* $a : \mathbf{A} \to \mathbf{B}$ *and* $b : \mathbf{B} \to \mathbf{A}$ *such that* $b \circ a$ *is homotopic to the identity map on* **A**, *and* $a \circ b$ *is homotopic to the identity map on* **Y**.

If a topological space is a deformation retract of another topological space, then they are homotopy equivalents. However the converse is not true. A special kind of homotopy equivalent topological space is a *contractible space*.

Definition 2.1.8. *Contractible space [60]. A topological space* **X** *is called contractible if the identity map on* **X** *is homotopic to a constant map.*

The intuition behind the concept of contractible space is that the topological space can be contracted continuously to a point contained in the space.

In this subsection a number of equivalence relationships between topological spaces were stated which can be used to classify these spaces. In general it is difficult to come up with functions to prove homeomorphism or homotopic equivalents for arbitrary topological space. Algebraic topology tries to circumvent the problem of analyzing topological spaces by imparting certain algebraic structures(primarily group) to a topological space, and allows interpretation of the structure of the topological space by analysis of the algebraic structures. In this thesis an applied version of algebraic topology known as *topological data analysis*(TDA) is used. For this reason a detailed background of algebraic topology is not introduced instead only the concepts required for TDA are discussed in Chapter 4, Chapter 5, Chapter 6 and Chapter 7. An in-depth treatment of algebraic topology can be found in [60].

2.2 Optimization Theory for Infinite dimensional spaces

This section introduces some of the mathematical concepts required for the in-depth understanding of materials presented in Chapter 3. The reader would require concepts from functional analysis, partial differential equations and optimization theory described in the following subsection in order to follow the derivation of the gradient in Chapter 3. The information presented in section has been adapted from [46, 145, 84] and [79].

2.2.1 Functional Analysis

Definition 2.2.1. *A normed space is a vector space with a metric defined by a norm. A norm generalizes the notion of length of a vector in a plane or in three-dimensional space.*

Definition 2.2.2. *A normed space* $\{\mathbf{X}, \|\cdot\|\}$ *is said to be a complete metric space if every Cauchy sequence in* $\mathbf{X}$ *converges to an element in* $\mathbf{X}$*. A Banach space is a normed space which is a complete metric space.*

Definition 2.2.3. *A Hilbert space is a Banach space which is endowed with an inner product* $< \cdot, \cdot >$*. The inner product induces the norm for space defined as* $\|x\| = (< x, x >)^{\frac{1}{2}}$

Definition 2.2.4. *A linear mapping* $\mathbf{A}$ *from a normed space* $\mathbf{X}$ *into a normed space* $\mathbf{Y}$ *is called an linear operator if*

$$A(ax + by) = aA(x) + bA(y) \tag{2.1}$$

$\forall x, y \in \mathbf{X}$ *and* $a, b \in \mathbb{R}$

Definition 2.2.5. *A linear functional is a linear mapping from a normed space into a scaler field* $\mathbb{R}$ *or* $\mathbb{C}$*.*

Bounded linear operators and *bounded linear functionals* are of particular importance since they are continuous and take advantage of the vector space structure.

Definition 2.2.6. *A linear operator* $\mathbf{A} : \mathbf{X} \to \mathbf{Y}$ *is said to be bounded if*

$$\|\mathbf{A}\| := \sup \{\|\mathbf{A}(x)\|_{\mathbf{Y}} \mid \|x\|_{\mathbf{X}} \leq 1\} < \infty \tag{2.2}$$

It is a well known theorem that a linear operator is continuous if and only if it is bounded[79].

Definition 2.2.7. $\mathcal{L}(\mathbf{X}, \mathbf{Y})$ *denotes the normed vector space of all linear bounded mappings from* $\mathbf{X}$ *to* $\mathbf{Y}$*. If* $\mathbf{X} = \mathbf{Y}$*, then we write* $\mathcal{L}(\mathbf{X}, \mathbf{Y}) := \mathcal{L}(\mathbf{X})$*.*

Definition 2.2.8. *Dual space of* $\mathbf{X}$ *is the space* $\mathbf{X}^*$ *of linear functionals on* $\mathbf{X}$*,* $\mathbf{X}^* := \mathcal{L}(\mathbf{X}, \mathbb{R})$*. Dual space is also a normed space with the associated norm,*

$$\left\|f\right\|_{\mathbf{X}^*} = \sup_{\|x\|=1} \left|f(x)\right| \tag{2.3}$$

The notation $< \cdot, \cdot >_{\mathbf{X}^*,\mathbf{X}}$ refers to the dual pairing between $\mathbf{X}^*$ and $\mathbf{X}$ defined as $< f, x >_{\mathbf{X}^*,\mathbf{X}} = f(x)$.

Theorem 1. *Reisz Representation theorem. Let* $\{\mathbf{H}, < \cdot, \cdot >\}$ *be a real Hilbert space, then for any element* $G \in \mathbf{H}^*$ *there exists a uniquely* $g \in \mathbf{H}$ *such that* $\|G\|_{\mathbf{X}^*} = \left\|g\right\|_{\mathbf{X}}$ *and* $G(h) =< g, h > \forall h \in \mathbf{H}$

Definition 2.2.9. *Let* $1 \leq p < \infty$ *and suppose* Ω *is a Lesbesgue measurable subset of* $\mathbb{R}$ *then the following function spaces are defined,*

$$\mathbf{L}^p(\Omega) = \left\{ f : \Omega \to \mathbb{R}, \sqrt[p]{\left(\int_\Omega \left|f\right|^p dx \right)} < \infty \right\} \tag{2.4}$$

and

$$\mathbf{L}^\infty(\Omega) = \left\{ f : \Omega \to \mathbb{R}^n, ess \sup_{x \in \Omega} \left|f(x)\right| < \infty \right\}. \tag{2.5}$$

Also,

$$\mathbf{L}^p_{loc}(\Omega) = \{ f : \Omega \to \mathbb{R}, \forall \text{ compact } \omega \subset \Omega \; f \in \mathbf{L}^p(\omega) \} \tag{2.6}$$

Theorem 2. *(Fischer-Riesz) The function spaces* $\mathbf{L}^p(\Omega)$ *are Banach spaces. The function space* $\mathbf{L}^2(\Omega)$ *is a Hilbert space with the inner product structure defined as:*

$$< f, g >_{\mathbf{L}^2(\Omega)} := \int_\Omega f g dx \tag{2.7}$$

2.2.2 Partial Differential Equations

Definition 2.2.10. *Let* $\Omega \subset \mathbb{R}^n$ *be open and let* $f \in \mathbf{L}^p_{loc}(\Omega)$*. If there exists a function* $w \in \mathbf{L}^p_{loc}(\Omega)$ *such that,*

$$\int_\Omega w\phi = (-1)^\alpha \int_\Omega f D^\alpha \phi dx, \ \ \forall \phi \in C_0^\infty(\Omega) \tag{2.8}$$

then $D^\alpha f := w$ *is called* α*-th weak partial derivative of* f*.* $C_0^\infty(\Omega)$ *is the set of infinitely differentiable test functions on* Ω *which vanish on the boundary of* Ω*.*

Definition 2.2.11. *Let* $\Omega \subset \mathbb{R}^n$*. The Sobolev space* $\mathbf{W}^{k,p}$ *is defined as,*

$$\mathbf{W}^{k,p} = \{f \in \mathbf{L}^p(\Omega) \mid D^\alpha f \in \mathbf{L}^p(\Omega) \ \forall \, |\alpha| \leq k\} \tag{2.9}$$

For $k \in \mathbb{N}_0$, $p \in [1, \infty)$*. The space is endowed with the norm*

$$\left\| f \right\|_{\mathbf{W}^{k,p}}(\Omega) = \sqrt[p]{\left(\sum_{|\alpha| \leq k} \int_\Omega \left| D^\alpha f(x)^p \right| dx \right)} \tag{2.10}$$

A space of particular interest is $H^k(\Omega) := \mathbf{W}^{k,2}(\Omega)$, which is a Hilbert space. Using definition of $\mathbf{W}^{k,p}$,

$$H^1(\Omega) = \left\{ f \in \mathbf{L}^2(\Omega) \mid D^1 f \in \mathbf{L}^2(\Omega) \right\} \tag{2.11}$$

The space $H^1(\Omega)$ is endowed with the norm

$$\left\| f \right\|_{H^1(\Omega)} = \sqrt{\int_\Omega \left(f^2 + \left| \nabla f \right|^2 \right) dx} \tag{2.12}$$

being a Hilbert space $H^1(\Omega)$ has the following inner product structure,

$$< f, g >_{H^1(\Omega)} = \int_\Omega fg dx + \int_\Omega \nabla f . \nabla g dx \tag{2.13}$$

Consider a separable Banach space $\mathbf{X}$.

Definition 2.2.12. *$s : [0,T] \to \mathbf{X}$ is a simple function if it possess the form*

$$s(t) = \sum_{i=1}^{m} 1_{E_i}(t) y_i \tag{2.14}$$

where $E_i \subset [0,T]$ are Lesbesgue measurable sets and $y_i \in \mathbf{X}$

$f : [0,T] \to \mathbf{X}$ is a strongly measurable function if there exist a sequence of simple functions $s_k : [0,T] \to \mathbf{X}$ such that,

Definition 2.2.13. *For a separable Banach space $\mathbf{X}$, the function space $\mathbf{L}^p([0,T];\mathbf{X})$ is defined for $1 \leq p < \infty$ as*

$$\mathbf{L}^p([0,T];\mathbf{X}) := \left\{ y : [0,T] \to \mathbf{X} \mid y \text{ is strongly measurable}, \int_0^T \left\| y(t) \right\|_{\mathbf{X}}^p dt < \infty \right\} \tag{2.15}$$

also,

$$\mathbf{L}^\infty([0,T];\mathbf{X}) := \left\{ y : [0,T] \to \mathbf{X} \mid y \text{ is strongly measurable}, ess \sup_{t \in [0,T]} \left\| y(t) \right\|_{\mathbf{X}} < \infty \right\} \tag{2.16}$$

Definition 2.2.14. *Weak time derivative. $v \in \mathbf{L}^1([0,T];\mathbf{X})$ is the weak time derivative of $y \in \mathbf{L}^1([0,T];\mathbf{X})(y_t = v)$ if*

$$\int_0^T \phi'(t) y(t) dt = -\int_0^T \phi(t) v(t) dt \quad \forall \phi \in C_0^\infty((0,T)) \tag{2.17}$$

Theorem 3. *The dual space of $\mathbf{L}^p([0,T];\mathbf{X})$ can be identified with $\mathbf{L}^q([0,T];\mathbf{X}^*)$, for $1 \leq p < \infty$ and $\frac{1}{p} + \frac{1}{q} = 1$ using the pairing,*

$$< v, y >_{\mathbf{L}^q([0,T];\mathbf{X}^*),\mathbf{L}^p([0,T];\mathbf{X})} = \int_0^T \langle v(t), y(t) \rangle_{\mathbf{X}^*,\mathbf{X}} dt \tag{2.18}$$

If the space $\mathbf{X}$ *is a separable Hilbert space then* $\mathbf{L}^2([0,T];\mathbf{X})$ *is a Hilbert space with inner product defined as*

$$< v, y >_{\mathbf{L}^q([0,T];\mathbf{X})} := \int_0^T \langle v(t), y(t) \rangle_{\mathbf{X}} \, dt \tag{2.19}$$

Definition 2.2.15. *$W([0,T];H,V)$ is the linear space of all $y \in \mathbf{L}^2([0,T];V)$ having a distributional time derivative $y' \in \mathbf{L}^2([0,T];V^*)$, where H, V are separable Hilbert spaces with continuous and dense embedding $V \hookrightarrow H$. If H is identified with its dual H^*, then the following continuous and dense embedding is obtained,*

$$V \hookrightarrow H \cong H^* \hookrightarrow V^* \tag{2.20}$$

This relation is called Gelfand Triple.

Theorem 4. *For a Gelfand Triple $V \hookrightarrow H \hookrightarrow V^*$ there exists a continuous embedding $W([0,T];H,V) \hookrightarrow C([0,T];H)$. Moreover, for all $y, p \in W([0,T];H,V)$ the following formula can be obtained*

$$\langle y(T), p(T) \rangle_H - \langle y(0), p(0) \rangle = \int_0^T \langle y'(t), p(t) \rangle_{V^*,V} \, dt + \int_0^T \langle p'(t), y(t) \rangle_{V^*,V} \, dt \tag{2.21}$$

2.2.3 Optimization Theory

Definition 2.2.16. *Let $G : U \subset \mathbf{X} \to \mathbf{Y}$ be an operator with a non empty subset U as the domain. If the limit*

$$dG(u,h) := \lim_{t \to 0} \frac{G(u+th) - G(u)}{t} \tag{2.22}$$

exists in V, then it is called the directional derivative of G at u along the direction h. If the limit exists for all directions $h \in U$, then the mapping $h \to dG(u,h)$ is termed as the first variation of G at u.

Definition 2.2.17. *If the first variation of G at $u \in U$ exists and if there exists a continuous linear operator $\mathcal{A} : \mathbf{X} \rightarrow \mathbf{Y}$ in the form,*

$$dG(u,h) = \mathcal{A}h \ \ \forall h \in \mathbf{X} \tag{2.23}$$

Then G is Gateaux differentiable at u and the operator $\mathcal{A}$ becomes the Gateaux derivative of the function at u. Alternately, $\mathcal{A} = G'(u)$.

Theorem 5. *Let $\mathbf{X}$ be a Banach space and $C \subset \mathbf{X}$ be a non empty and convex subset. Also, let $\mathbf{J} : \mathbf{X} \rightarrow \mathbb{R}$ be defined on an open neighborhood of C. If $x*$ is a local solution to the problem $\min_{x \in \mathbf{X}} \mathbf{J}(x)$ s.t $x \in C$ at which $\mathbf{J}$ is Gateaux-differentiable, then the following optimality condition holds,*

$$\langle \mathbf{J}'(x^*), x^* - x \rangle_{\mathbf{X}^*, \mathbf{X}} \geq 0 \ \ \forall x \in C \tag{2.24}$$

2.3 Information Theory

Information theory studies the quantification, storage, and communication of information [28]. It was originally proposed by Claude E. Shannon in 1948 to find fundamental limits on signal processing and communication operations such as data compression, in a landmark paper entitled "A Mathematical Theory of Communication". A key measure in information theory is *entropy*. Entropy quantifies the amount of uncertainty involved in the value of a random variable. In Chapter 7 an information theoretic gain metric called *mutual information* is used to direct a swarm of robots towards unexplored regions in an unknown environment. The information presented in this section are adopted from [28, 151]

2.3.1 Probability theory

Given a sample space Ω, a $\sigma-$algebra $\mathcal{F} \subset 2^{\Omega}$, and a probability measure $\mathbb{P}$, let the triple $(\Omega, \mathcal{F}, \mathbb{P})$ denote a probability space. Given a random variable X

that maps Ω to an alphabet set χ such that $X : \Omega \to \chi$, let $\mathbb{P}(X = x)$ denote the probability density when X takes the value $x \in \chi$. $\mathbb{P}(X)$ is the shorthand notation for $\mathbb{P}(X = x)$. Let $Y : \Omega \to \gamma$ be a second random variable, then $\mathbb{P}(X, Y)$ denote the joint probability, i.e., $P(\{X = x\} \cup \{Y = y\})$ with $x \in \chi$ and $y \in \gamma$. From Kolmogorov's definition of conditional probabilities of X given Y : $\mathbb{P} \coloneqq \mathbb{P}(X, Y)/\mathbb{P}(Y)$. The expected value of a real random variable $A : \Omega \to \mathbb{R}$ denoted as $\mathbb{E}(A)$ is given by $\mathbb{E}(A) \coloneqq \int_\Omega A \, d\mathbb{P}$.

Given two independent theoretical experiments $\mathbf{E}^{(1)}$ and $\mathbf{E}^{(2)}$ that are statistically identical, and let $\left(\Omega^{(1)}, \mathcal{F}^{(1)}, \mathbb{P}^{(1)}\right)$ denote the probability space associated with the experiment $\mathbf{E}^{(1)}$. Likewise, $\left(\Omega^{(2)}, \mathcal{F}^{(2)}, \mathbb{P}^{(2)}\right)$ is defined for experiment $\mathbf{E}^{(2)}$. As the two experiments are statistically identical, the two probability spaces are identical. Let X_1 and X_2 be random variables defined on $\left(\Omega^{(1)}, \mathcal{F}^{(1)}, \mathbb{P}^{(1)}\right)$ and $\left(\Omega^{(2)}, \mathcal{F}^{(2)}, \mathbb{P}^{(2)}\right)$ respectively. The two random variables are statistically identical, meaning, they are identical as a function from the $\sigma-$algebra to the appropriate range space. Let the product probability space [129] of $\left(\Omega^{(1)}, \mathcal{F}^{(1)}, \mathbb{P}^{(1)}\right)$ and $\left(\Omega^{(2)}, \mathcal{F}^{(2)}, \mathbb{P}^{(2)}\right)$ be denoted by $\left(\Omega^{(3)}, \mathcal{F}^{(3)}, \mathbb{P}^{(3)}\right)$. Formally speaking, the product sample space is given by $\Omega^{(3)} = \Omega^{(1)} \times \Omega^{(2)}$, the resultant sigma algebra $\mathcal{F}^{(3)} \subset 2^{\Omega^{(3)}}$ is the smallest $\sigma-$algebra such that $\mathcal{F}^{(3)}$ contains $\mathcal{F}^{(1)} \times \mathcal{F}^{(2)}$, and the probability measure $\mathbb{P}^{(3)}$ is such that, for any event $\mathcal{E}^{(3)} = \mathcal{E}^{(1)} \times \mathcal{E}^{(2)} \subset \mathcal{F}^{(3)}$ with $\mathcal{E}^{(1)} \subset \mathcal{F}^{(1)}$ and $\mathcal{E}^{(2)} \subset \mathcal{F}^{(2)}$, it satisfies $\mathbb{P}^{(3)}(\mathcal{E}^{(3)}) = \mathbb{P}^{(1)}(\mathcal{E}^{(1)}) \times \mathbb{P}^{(2)}(\mathcal{E}^{(2)})$.

2.3.2 Mutual Information

Given a random variable X, let $H(X)$ denote the entropy defined as

$$H(X) \coloneqq -\mathbb{E}[\log(\mathbb{P}(X))] \tag{2.25}$$

Using the above definition entropy of a discrete random X_d that can take values in the set $\chi = \{x_1, \cdots, x_N\}$ becomes,

$$H(X_d) = -\sum_{x \in \chi} \mathbb{P}(X_d = x) \log(\mathbb{P}(X_d = x)) \tag{2.26}$$

In general, $H(X_d)$ is non-negative and zero if and only if X_d is deterministic. The idea entropy can be extended to the case of continuous random variables. Let X_c be a continuous random variable, then its *differential entropy*(entropy of a continuous random variable is termed as differential entropy) is given by:

$$H(X_c) = -\int \mathbb{P}(X_c = x) \log(\mathbb{P}(X_c = x))\, dx \tag{2.27}$$

Contrary to the entropy of discrete random variables differential entropy can be negative. Also, differential entropy is not the limiting case of discrete entropy [28]. Despite these differences, differential entropy is conceptually similar to discrete entropy and the two concepts are used interchangeably.

Let $X^{(1)}, \cdots, X^{(n)}$ be a finite collection of random variables, then the joint entropy is defined as,

$$H(X^{(1)}, \cdots, X^{(n)}) = -\mathbb{E}[\log(\mathbb{P}(X^{(1)}, \cdots, X^{(n)}))] \tag{2.28}$$

It is shown in [28] that,

$$H(X^{(1)}, \cdots, X^{(n)}) \leq \sum_{i=1}^{n} H(X^{(i)}) \tag{2.29}$$

and equality in the inequality above holds if and only if the random variables are independent.

An interesting question in information theory is " how is the uncertainty of a random variable is affected by a different random variable if the outcome of the

latter is given?". Formally speaking, if X and Y are two random variables then how to compute the quantity $H(X|Y)$. It is important to note that, this quantity is different from $H(X|Y = y)$; which is simply the entropy of the distribution $\mathbb{P}(X|Y = y)$. The difference is that, for $H(X|Y)$ Y is also a random variable. $H(X|Y)$ is known as the conditional entropy of X over Y defined as

$$H(X|Y) = \mathbb{E}(H(X|Y = y)) \tag{2.30}$$

Similarly, conditional differential entropy can also be defined for continuous random variables where appropriate sums are replaced by integrals.

Conditional entropy and entropy can be used to define a new quantity known as *mutual information*. Mutual information measures the likely decrease in uncertainty of a random variable. The mutual information between random variable X and Y is defined as:

$$\begin{aligned} I_{MI}[X;Y] &= H(X) - H(X|Y) && (2.31)\\ &= H(Y) - H(Y|X) && (2.32) \end{aligned}$$

Mutual information is always non-negative and is zero if and only if the random variables are independent. Some interesting properties of mutual information are:

1. $I_{MI}[X;Y] = I_{MI}[Y;X]$
2. $I_{MI}[X;Y] = H(X) + H(Y) - H(X,Y)$
3. $I_{MI}[X^{(1)}, \cdots, X^{(n)};Y] = \sum_{i=1}^{n} I_{MI}[X^{(i)};Y|X^{(1)}, \cdots X^{(i-1)}]$ where
   $I_{MI}[X^{(i)};Y|X^{(1)}, \cdots X^{(i-1)}] = H(X^{(i)}|X^{(1)}, \cdots X^{(i-1)}) - H(X^{(i)};Y|X^{(1)}, \cdots X^{(i-1)})$

From the last property it is easy to show that,

$$I_{MI}[X^{(1)}, \cdots, X^{(n)}; Y] \leq \sum_{i=1}^{n} I_{MI}[X^{(i)}; Y] \tag{2.33}$$

As before equality holds if and only if the random variable are independent.

Mutual information can also expressed in terms of the Kullback-Leibler divergence(KL divergence or KLD)[24]. KLD is a pseudo-metric to measure the "distance" between two probability density functions. KL divergence between two probability density functions f and g is defined as:

$$D_{KL}[f||g] = \int f(x) \log\left(\frac{f(x)}{g(x)}\right) dx \tag{2.34}$$

The mutual information between two random variables X and Y can also be written as:

$$I_{MI}[X; Y] = D_{KL}\left[\mathbb{P}(X, Y)||\mathbb{P}(X)\mathbb{P}(Y)\right] \tag{2.35}$$

Chapter 3

AN ADVECTION-DIFFUSION-REACTION BASED APPROACH TO MAPPING AN ENVIRONMENTAL FEATURE

Source: Ragesh K. Ramachandran et al. [132]

Funding: NSF Awards CMMI-1363499 and CMMI-1436960

ABSTRACT

This chapter presents an approach to mapping a region of interest using observations from a robotic swarm without localization. The robots have local sensing capabilities and no communication, and they exhibit stochasticity in their motion. The swarm population dynamics is modeled with a set of advection-diffusion-reaction partial differential equations (PDEs). The map of the environment is incorporated into this model using a spatially-dependent indicator function that marks the presence or absence of the region of interest throughout the domain. To estimate this indicator function, the solution of an optimization problem is defined in which we minimize an objective functional that is based on temporal robot data. The optimization is performed numerically offline using a standard gradient descent algorithm. Simulations show that the approach can produce fairly accurate estimates of the positions and geometries of different types of regions in an unknown environment.

This chapter presents a method for *mapping a feature of interest in an unknown environment* using a swarm of robots with local sensing capabilities, no localization, and no inter-robot communication. The chapter considers scenarios where the robots exhibit significant randomness in their motion due to sensor and actuator noise or, at the nanoscale, the effects of Brownian motion and chemical interactions. The mapping approach proposed in this chapter is scalable with the number of robots, so that arbitrary swarm populations can be used.

The proposed method relies on developing a continuous abstraction of the swarm population dynamics in the form of an advection-diffusion-reaction PDE model, which is referred to as the *macroscopic model*. This model describes the spatial and temporal evolution of the population densities of robots in different states throughout the domain. To represent individual robots, a *microscopic model* is defined that describes how each robot moves and responds upon encountering a feature of interest. The state transition of a robot is modeled as an irreversible chemical reaction with a high reaction rate. The macroscopic model becomes a more accurate model of the microscopic model as the number of robots increases. In other words, the macroscopic model is a limiting case of the microscopic model as the robot population size tends to infinity.

The mapping problem is posed as the computation of a spatially varying function that represents the map of the feature of interest. To estimate this function, the temporal data that is recorded by the robots during their exploration of the environment is used. This data yields the time evolution of the number of robots that are still exploring the domain; i.e., robots that have not encountered the feature. In practice, this data could be collected from the robots after their deployment by retrieving their recorded times of encounter with the feature. In biomedical imaging applications with nanoscale swarms, this data could be obtained from a

measurable signal that corresponds to the density of the population that is still in the exploring state.

Once this data is obtained, techniques from optimal control are used to compute the function that represents the feature map. In general, optimal control entails the minimization or maximization of an objective functional that is defined in a finite-dimensional space and is subject to a set of ordinary or partial differential *constraint equations*, which govern the system of interest. From a computational perspective, optimal control methods are more effective than black box techniques, such as genetic algorithms and particle swarm optimization, in terms of the number of objective functional evaluations per cycle. This computational advantage mainly arises from their use of the problem structure to calculate the gradient of the control-to-state maps using the adjoint equation. The feature map is defined as the solution of an optimization problem that minimizes an objective functional which is based on the robot data.

This optimization problem is solved numerically offline using standard techniques such as gradient descent algorithms. The approach is validated in simulation for features of varying shape, size, orientation, and location.

Division of Work Karthik Elamvazhuthi helped with formulation of the objective function for the optimal control problem. The author coded up the simulations in Matlab and validated the method.

3.1 Related Work

In the literature, there have been exhaustive studies on mapping and exploring an environment using robots. SLAM (simultaneous localization and mapping) [111, 82], probabilistic mapping [136, 15], and topological and metric map building

[137, 80] are some of the techniques that have been developed for environmental mapping by robots. These techniques have been used for path planning and mapping in small multi-robot groups. However, the problem of scaling these approaches to larger groups becomes intractable for swarms of hundreds or thousands of robots, due to their limitations on communication bandwidth and their spatially distributed nature. In addition, these techniques require the robots to have sophisticated sensing and processing capabilities, which are not feasible in swarm robotic platforms.

Mapping an environment using a robotic swarm is a relatively new area of research in the robotics community. An approach to this problem is given in [147, 33], in which a robotic swarm is used to identify the topological features of an environment from information about the times at which robots encounter other robots and environmental features. This work borrows tools from algebraic geometry and topological data analysis to compute a metric that can be used to classify the topological structure of the environment. The approach requires some minimal inter-robot communication, unlike our strategy which is communication-free.

Our mapping approach uses methods from [42], a stochastic task allocation approach that achieves target spatial distributions of robot activity without using communication or localization. Also, our approach is inspired by, a method for reconstructing environmental features from minimal robot data using compressed sensing techniques. In contrast to the scenarios that we consider, the robots in [42] can move over the features to be mapped, which allows the mapping problem to be formulated as the inversion of a linear operator. Approaches with a similar mathematical framework for parameter estimation have been used extensively in the area of biomedical imaging, especially with MRI and CT scan images. In

these approaches, the system is excited with a stimulus such as a magnetic field, X-rays, or ultrasound, and the system response is used to identify and estimate a spatially-dependent parameter that corresponds to the image [148].

3.2 Problem Statement

We consider a scenario in which N robots are deployed into an unknown, bounded environment to map a single feature of interest. We exclude cases in which the feature is located very close to the domain boundary, since robot collisions with this boundary and the high diffusion of swarms that start far from the feature will degrade the estimation. If a robot encounters the feature, it stops moving and records the time at which it stopped. Using data on the number of robots that are still moving at each instant, we aim to estimate the position and geometry of the encountered feature. We can improve the accuracy of this estimate by deploying the swarm in different directions from various locations, which will ensure greater coverage of the domain and result in robot collisions with a larger portion of the feature boundary. This approach may be used to map multiple sparsely distributed features by reconstructing each individual feature from its corresponding data set and computing the entire map as a linear combination of single-feature maps.

Robot capabilities: The robots are assumed have sufficient power to complete the mapping operation. The power requirement for the robots is low, since they are not equipped with communication devices or GPS. The robots have local sensing capabilities and can identify the feature at distances within their sensing range. We may also assume that the robots can detect other robots within their sensing range and perform collision avoidance maneuvers, although we do not simulate collision avoidance in this work. Each robot is equipped with a compass and thus can move in a specified heading. Additionally, the robots

have sufficient memory to store the time of their encounter with the feature.

Robot controller: The robots begin at a specified location in the domain. During a swarm deployment, the robots move with a predetermined time-dependent velocity, $\mathbf{v}(t) \in \mathbb{R}^2$. This velocity is designed to guide the center of mass of the swarm along a desired trajectory through the environment. The velocity field may be initially transmitted to the robots by a computer at their starting location, or the robots may be directed according to the field using external stimuli such as magnetic fields or radiation. The robots' motion is affected appreciably by sensor and actuator noise, due to lack of feedback. If a robot detects a feature within its sensing range, it stops moving and records the time. At a predefined time t_f, the stationary robots around the feature boundary return to the starting point of the deployment and upload their encounter times to a computer. The computer then applies the optimal control method described in Section 3.6 to estimate the map of the feature using this robot data.

3.3 Microscopic Model

This model is used to simulate a robot's motion and its response to an encounter with a feature in its path. The change in a robot's state that is triggered by an encounter is modeled as an irreversible chemical reaction,

$$A \xrightarrow{k} P, \tag{3.1}$$

where the species A represents an *active* (moving) robot, P represents a *passive* (stationary) robot, and k is the reaction rate constant, which in this case is a fixed

probability per unit time. This constant is assigned a high value to enforce a high probability of transitioning from active to passive.

We model the robots as point masses with negligible size compared to the area of the domain. A particular robot i has position $\mathbf{X}_i(t) = [x_i(t)\ y_i(t)]^T$ at time t. The deterministic motion of the robot is directed by the time-dependent velocity field $\mathbf{v}(t) = [v_x(t)\ v_y(t)]^T$. The noise in the robot movement is modeled as a Brownian motion that drives diffusion with an associated diffusion coefficient D. We assume that the robots' navigation error can be modeled as diffusive noise and that the value of D can be estimated. The displacement of robot i over a time step Δt is given by the standard-form Langevin equation [50]:

$$\mathbf{X}_i(t + \Delta t) = \mathbf{X}_i(t) + (\sqrt{2D\Delta t})\mathbf{Z}(t) + \mathbf{v}(t)\Delta t, \tag{3.2}$$

where $\mathbf{Z}(t) \in \mathbb{R}^2$ is a vector of independent standard normal random variables that are generated at time t. The robots avoid collisions with the domain boundary by performing a specular reflection when they encounter this boundary.

3.4 Macroscopic Model

The macroscopic model governs the time evolution of the expected spatial distribution of the robotic swarm. For a swarm whose members move according to Equation 3.2, the macroscopic model is given by an advection-diffusion PDE, as described in [27]. Since our microscopic model includes robot state changes that can be represented as chemical reactions, our macroscopic model takes the form of an advection-diffusion-reaction (ADR) PDE. The model is defined over a domain $\Omega \subset \mathbb{R}^2$ with Lipschitz continuous boundary $\partial\Omega$ and over a time interval $[0, T]$. We define $L = \Omega \times [0, T]$ and $\Gamma = \partial\Omega \times [0, T]$. The state of the macroscopic model is the

population density field $u(\mathbf{x}, t)$ of active robots in the domain at points $\mathbf{x} \in \Omega$ and times $t \in [0, T]$. We specify a spatially varying *indicator function*, $K(\mathbf{x}) : \Omega \rightarrow \{0, 1\}$, that equals 0 at points $\mathbf{x}$ where the feature of interest is absent and equals 1 at points where it is present. The reaction term of the macroscopic model is determined by the rate constant k in Equation 3.1, which is switched on or off by the indicator function $K(\mathbf{x})$ depending on whether the feature of interest occupies point $\mathbf{x}$. This term models the switching of individual robots from the active state to the passive state when they are in the vicinity of the feature. The advection term of the macroscopic model is governed by the velocity field $\mathbf{v}(t)$ that is defined in the microscopic model.

From the above definition, the macroscopic model is given by:

$$\frac{\partial u}{\partial t} = \nabla \cdot (D\nabla u - \mathbf{v}(t)u) - kK(\mathbf{x})u \quad in \;\; L \tag{3.3}$$

with the no-flux boundary condition

$$\vec{n} \cdot (D\nabla u - \mathbf{v}(t)u) = 0 \quad on \;\; \Gamma, \tag{3.4}$$

where $\vec{n} \in \mathbb{R}^2$ is the outward normal of the boundary $\partial\Omega$. We specify that all robots start in the active state and set the initial condition,

$$u(\vec{x}, 0) = u_0, \tag{3.5}$$

to a Gaussian density centered at a point $\mathbf{x}_0$, which we assume is far from the feature. The macroscopic model is numerically solved using the explicit finite-volume method that is described in [42].

Our approach relies on the close correspondence of the macroscopic model solution to the average swarm density over an ensemble of microscopic model simulations. Therefore, the approach is robust to robot malfunctions and external

disturbances as long as these factors do not significantly affect the model correspondence. This implies that the number of failed robots should be small compared to the total swarm size, and that the robots' trajectory drift due to wind, currents, and other environmental influences should be small relative to their modeled motion. In scenarios that violate these conditions, it would be necessary to improve the accuracy of the macroscopic model by estimating the components of $\mathbf{v}$, D, and k that are affected by unmodeled dynamics and disturbances. This is a topic of future work.

3.5 Mathematical Preliminaries

We study the solution to PDEs in the weak sense, which can be found in the Sobolev space $H^1(\Omega) = \left\{y \in L^2(\Omega) : \frac{\partial y}{\partial x_1} \in L^2(\Omega), \ \frac{\partial y}{\partial x_2} \in L^2(\Omega)\right\}$. Here, the spatial derivative is to be understood as a weak derivative defined in the distributional sense. The space is equipped with the common Sobolev space norm, $\left\|y\right\|_{H^1(\Omega)} = \sqrt{\left(\left\|y\right\|^2_{L^2(\Omega)} + \sum_{i=1}^{2} \left\|\frac{\partial y}{\partial x_i}\right\|^2_{L^2(\Omega)}\right)}$. We also define $V = H^1(\Omega)$, which has the dual space $V^* = H^1(\Omega)^*$.

We consider the general system for Equation 3.3-Equation 3.5:

$$\begin{aligned} \frac{\partial u}{\partial t} = Au + \sum_{i=1}^{2} v_i B_i u - K(\vec{x})u + f \quad & in \ \ L, \\ \vec{n} \cdot (D\nabla u - \mathbf{v}u) = g \quad & on \ \ \Gamma, \\ u(\vec{x}, 0) = u_0, & \end{aligned} \tag{3.6}$$

where A is a formal operator and B_i is an operator defined as $B_i : L^2(0,T;V) \to L^2(0,T;L^2(\Omega))$, $K(\vec{x}) \in L^2(\Omega)$, $f \in F = L^2(0,T;L^2(\Omega))$ is the forcing function in the system, $g \in G = L^2(0,T;L^2(\partial\Omega))$, and $u_0 \in L^2(\Omega)$. The variational form of the operator A, called A_g, is defined as $A_g : L^2(0,T;V) \to L^2(0,T;V^*)$. The solution of the system in the weak sense is given by $u \in U = L^2(0,T;V)$ with $u_t \in U^* = L^2(0,T;V^*)$

if it satisfies the equation:

$$\left\langle \frac{\partial u}{\partial t}, \phi \right\rangle_{U^*,U} = \left\langle A_g, \phi \right\rangle_{U^*,U} + \sum_{i=1}^{2} \left\langle v_i B_i u, \phi \right\rangle_F - \left\langle K(\vec{x})u, \phi \right\rangle_F + \left\langle f, \phi \right\rangle_F \tag{3.7}$$

for all $\phi \in L^2(0,T;V)$. The boundary conditions are equipped with A_g in the variational formulation using Green's theorem. This is essentially the variational form of the Laplacian,

$$\left\langle A_g u, \phi \right\rangle_{U^*,U} = -\left\langle D\nabla u, \nabla\phi \right\rangle_{L^2(\Omega)} + \int_{\partial\Omega} (g + \vec{n} \cdot \mathbf{v}u)\, \phi dx. \tag{3.8}$$

In the macroscopic model Equation 3.3-Equation 3.5, we define $A = \nabla^2$, $B_i = \frac{\partial}{\partial x_i}$, $f = 0$, and $g = 0$.

3.6 Optimal Control Approach to Mapping Features

The feature reconstruction problem is framed as an optimal control problem. A gradient descent algorithm is used to compute the optimal control for the problem. An adjoint state equation approach is used to compute the gradient required for the algorithm [16]. The key advantage of this approach is that it derives an explicit formula for the gradient of the objective functional with respect to the control, subject to the constraints. The Hamiltonian and Pontryagin maximum principle can be to used to derive the adjoint equation for finite-dimensional systems. However, in the case of infinite-dimensional systems, the existence of the Hamiltonian has been proven only for a limited class of systems [47]. This motivated us to derive the directional directive of the control-to-state mapping and use the generalized chain rule of differentiation of composite mappings in Banach spaces, as is found in the literature [11, 146]. In order to make the derivatives of certain maps well-defined, an appropriate choice of spaces is made for the parameters and the solutions satisfying the system of differential equations. We present a Lagrangian-based

analysis of these derivatives in the Appendices A and B. The proof for the existence of optimal control for the problem is the same as the one shown in [43].

The optimization procedure uses data on the ratio of the number of active robots at each instant of time to the initial number of active robots at the start of the swarm deployment. To ensure sufficient coverage of the domain, the swarm can be deployed from multiple starting positions and directed along different trajectories. Once this data is obtained, the optimization procedure is performed to find the feature map that would produce data that is similar to the data obtained from the deployments. The computational cost increases greatly with the number of data sets (one from each deployment) that are used for optimization, since the number of PDEs to be solved per iteration varies linearly with the data sets. However, we can obtain a better estimate of the feature map with more data. Hence, there is a tradeoff between the computational cost of the optimization and the accuracy of the estimate. In order to resolve this issue, we discard data sets from deployments in which few robots undergo a state transition compared to the other deployments. A paucity of state transitions indicates that the swarm trajectories infrequently intersect the feature. In addition, our procedure can be easily parallelized since the most computationally intensive part is the solution of the PDEs.

The optimal control problem is formulated as follows. Each of the i swarm deployments yields a sequence of times at which active robots encounter the feature and switch to the passive state. From this data, we can determine the fraction $g_i(t) \in L^2([0,T])$ of active robots in the swarm at each time t during deployment i. The solution $u_i(\vec{x},t)$ of the corresponding macroscopic model Equation 3.3-Equation 3.5 can be used to compute the integral $\int_\Omega u_i(\vec{x},t)d\vec{x}$, the expected fraction of active robots in the domain at time t. We assume that the swarm size is sufficiently large for $g_i(t)$ to closely match this integral if the feature map, represented by the function

$K(\vec{x})$ in Equation 3.3, is known. Therefore, we can frame our optimization objective as the computation of the input $K(\vec{x})$ that minimizes the function

$$J_i(u_i) = \frac{1}{2} \left\| \int_\Omega u_i(\vec{x}, t) d\vec{x} - g_i(t) \right\|^2_{L^2([0,T])}. \tag{3.9}$$

Suppose that the data from N deployments are selected to compute the optimal controls. The swarm velocity and initial distribution for deployment i are given by $\mathbf{v}_i(t)$ and u_0^i, respectively. The macroscopic model with these parameters is considered to be the i^{th} set of constraints, which we denote by $\Psi_i(u_i, K)$ as in [146]. The solution to this model is given by u_i, and the set of solutions for all N deployments is $\mathbf{u} := \{u_1, u_2, ..., u_i, ..., u_N\}$. We define the space of macroscopic model solutions as $U = C([0,T]; L^2(\Omega))$ and the space of admissible input functions as $\Theta_{ad} = \{K(\vec{x}) \in L^2(\Omega);\ K_{min} \leq K(\vec{x}) \leq K_{max}\}$. Furthermore, W_i is a weight that quantifies the significance of the data from deployment i relative to data from the other deployments, and λ is the Tikhonov regularization parameter [77]. Using these definitions, we can frame the optimal control problem as:

$$\min_{(\vec{u}, K(\vec{x})) \in U^N \times \Theta_{ad}} \mathbf{J}(\vec{u}, K) = \sum_{i=1}^{N} W_i J_i(u_i) + \frac{\lambda}{2} \|K(\vec{x})\|^2_{L^2(\Omega)}, \tag{3.10}$$

subject to the constraints $\Psi_i(u_i, K)$, $i = 1, ..., N$.

We must compute the gradient of the objective functional $\mathbf{J}(\vec{u}, K)$ with respect to the control inputs in order to perform the gradient descent algorithm for minimizing this functional. We introduce the Lagrangian functional $\mathcal{L}$ and Lagrangian multipliers p_i, with $\mathbf{p} := \{p_1, p_2, ..., p_i, ..., p_N\}$:

$$\mathcal{L}(\mathbf{u}, \mathbf{p}, K) = \mathbf{J}(\mathbf{u}, K) + \sum_{i=1}^{N} \langle p_i, \Psi_i(u_i, K) \rangle. \tag{3.11}$$

The functions p_i, also known as the adjoint variables, express the sensitivity of the objective functional to variations in the input control variable $K(\mathbf{x})$. The necessary condition for optimality is $\nabla\mathcal{L} = 0$, which implies the following three conditions: (1) $\nabla_{\mathbf{u}}\mathcal{L} = 0$, the adjoint equation; (2) $\nabla_{\mathbf{p}}\mathcal{L} = 0$, the state equation in weak form; and (3) $\nabla_K\mathcal{L} = 0$, the optimal control constraint. These three equations are used to compute the gradient of $\mathbf{J}(\vec{u}, K)$. The derivation of the adjoint and gradient equations is described in appendix A and appendix B respectively.

The solution to an optimization problem that is obtained by a gradient descent algorithm is sensitive to the choice of the initial guess and may be a local minimum of the objective functional rather than the global minimum. To increase the likelihood of obtaining the global minimum, we choose an initial guess for the feature map, represented by $K(\mathbf{x})$, that is guaranteed to include the actual map. The simulation results discussed in Section 3.7 verified that choosing the initial guess in this manner helps in effectively recovering the map of the domain. This initial guess is that the feature covers the entire area traversed by the swarm during each of its i deployments (in actuality, the feature will occupy a subset of this area). Formally, we define $\gamma_i := [0, 1] \rightarrow \mathbb{R}^2$ as the trajectory of the swarm center during the i^{th} deployment and $B_2(\gamma_i(\tau), \delta)$ as a ball with radius δ centered at the point $\gamma_i(\tau)$, and we initially set $K(\vec{x}) = 1$ for all $\vec{x} \in \left(\cup_{i=1}^{N} B_2(\gamma_i(\tau), \delta)\right) \cap \Omega$, $\tau \in [0, 1]$. We choose δ to be 3 times the standard deviation of the initial Gaussian swarm distribution.

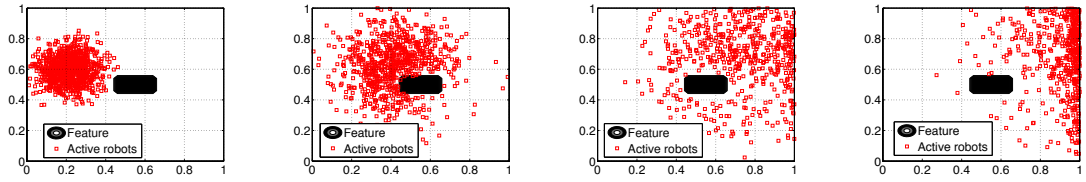
3.7 Simulated Mapping Scenarios

We developed microscopic and macroscopic models of a robotic swarm for six mapping scenarios, each with a single feature in the domain. The six features varied in position, size, shape, and orientation. We applied the method described in Section 3.6 to reconstruct each feature from the simulated robot data on feature

encounter times. For each simulation, we used a swarm of 1000 robots in a normalized domain of size 1 $m \times 1$ m. The value of k was chosen to be $1/dt$, where dt is the time step of the microscopic model, in order to ensure that robots always switched to the passive state when they encountered the feature boundary. For simplicity, the designated velocity fields $\mathbf{v}_i(t)$ of the robots were each assigned a constant heading. The robots moved at a speed of 0.012 m/s with a diffusion coefficient of $D = 5 \times 10^{-4}$ m^2/s, and each simulation ran for 80 s. The microscopic model was simulated in a 26×26 grid, while the macroscopic model was solved in a finer grid of 51×51 grid cells to account for numerical diffusion. In the optimization procedure, $K(\vec{x})$ was bounded between $K_{min} = 0$ and $K_{max} = 1$.

Figure 3.1 shows snapshots of the active robots in a swarm at various times t during a sample deployment. The robots behave according to the microscopic model and move through a domain that contains a rectangular feature. Robots that have switched to the passive state are not shown. The population of active robots decreases as the robots move eastward and encounter the feature in their path.

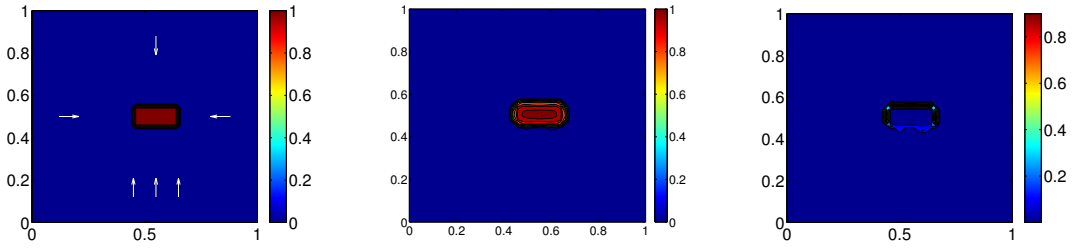
Figure 3.2 through figure 3.10 illustrate the results of our mapping procedure for the six scenarios that we investigated. Each figure shows the actual feature, the map of the feature given by the estimated $K(\vec{x})$, and the error between these two plots. In the plots of the actual features, the white arrows indicate the starting points and directions of the swarm center of mass during deployments, each of which yields one data set. Figure 3.2, figure 3.3 , and figure 3.4 show that we can obtain a fairly accurate map of a rectangle at two different orientations and a triangle using 6 data sets for each scenario. We consider smaller features in the next four figures. From figure 3.6 and figure 3.7, we see that the map of a feature increases in accuracy when more non-redundant data sets are used in the optimization procedure. Figure 3.5 shows that the technique is able to detect the position of the tiny triangle even



(a) t = 8 s (b) t = 27 s (c) t = 53 s (d) t = 75 s

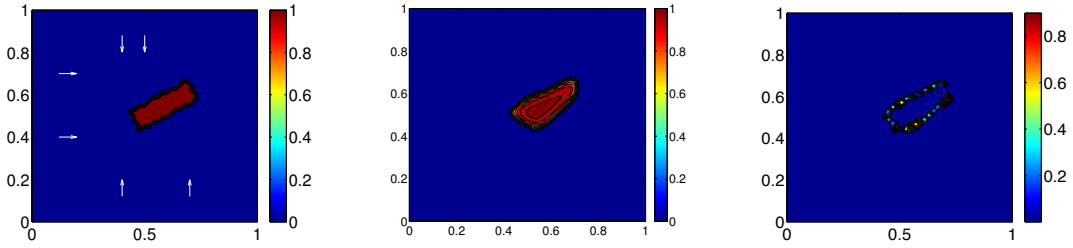
Figure 3.1: Snapshots of the simulated swarm moving through a domain with a rectangular feature.

though it is unable to accurately map its shape. Figure 3.8 represents a worst-case scenario, in which the map is estimated using data from swarms that start at locations far from the feature, which is in one corner of the domain. The swarms are highly diffused by the time they reach the vicinity of the square; however, 8 data sets yield a relatively accurate map. Lastly, figure 3.9 shows that 6 data sets yield a fairly poor estimate of a non-convex L-shaped feature; thus, further work is required on extending the technique to mapping non-convex shapes. Figure 3.10 shows that for each scenario considered, the optimal control approach effectively minimizes the objective function by driving it close to zero from its initial value. Figure 3.11 compares data on the fraction of active robots $g_i(t)$ and the integral $\int_\Omega u_i(\mathbf{x}, t)d\mathbf{x}$ during a particular swarm deployment i corresponding to the $K(\vec{x})$ shown in figure 3.4a.



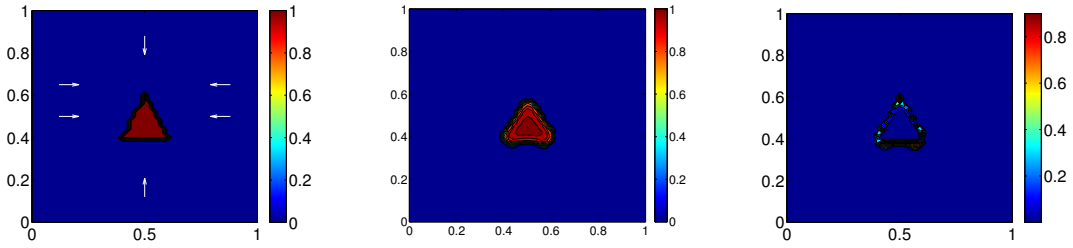
(a) Actual $K(\vec{x})$ (b) Estimated $K(\vec{x})$ (c) Absolute error of estimation

Figure 3.2: $K(\vec{x})$ estimated from 6 data sets for a domain that contains a rectangle. The white arrows show the starting locations and directions of the swarm deployments.



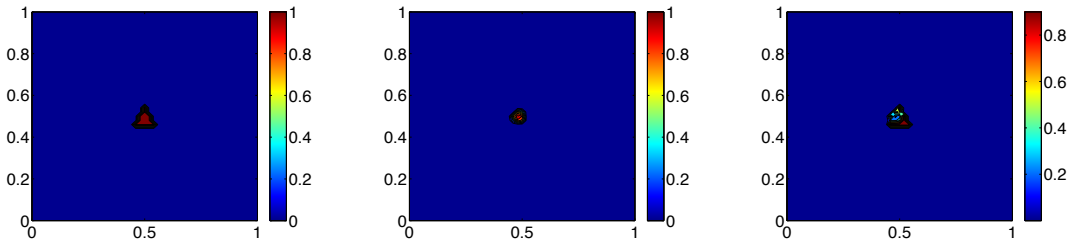
(a) Actual $K(\vec{x})$ (b) Estimated $K(\vec{x})$ (c) Absolute error of estimation

Figure 3.3: $K(\vec{x})$ estimated from 6 data sets for a domain that contains an inclined rectangle



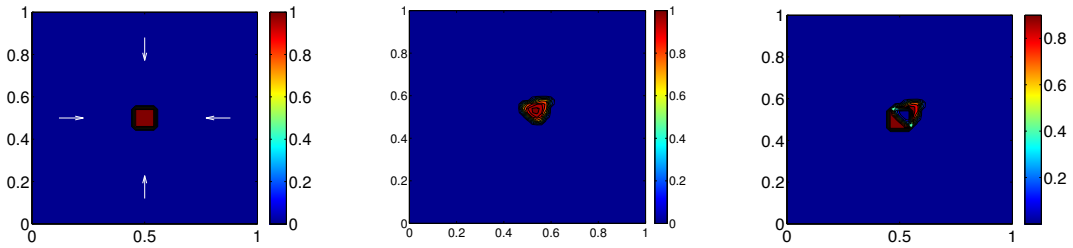
(a) Actual $K(\vec{x})$ (b) Estimated $K(\vec{x})$ (c) Absolute error of estimation

Figure 3.4: $K(\vec{x})$ estimated from 6 data sets for a domain that contains a triangle



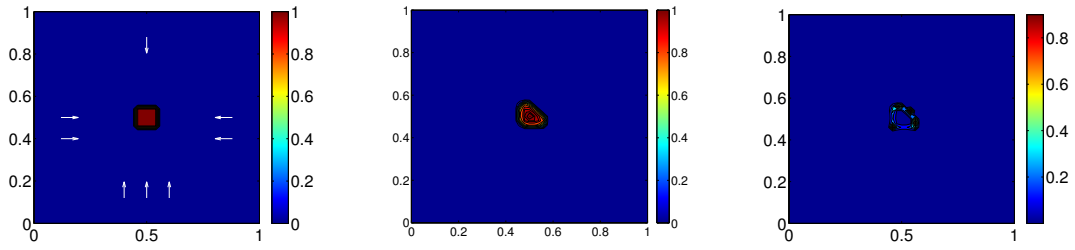
(a) Actual $K(\vec{x})$ (b) Estimated $K(\vec{x})$ (c) Absolute error of estimation

Figure 3.5: $K(\vec{x})$ estimated from 4 data sets for a domain that contains a small triangle



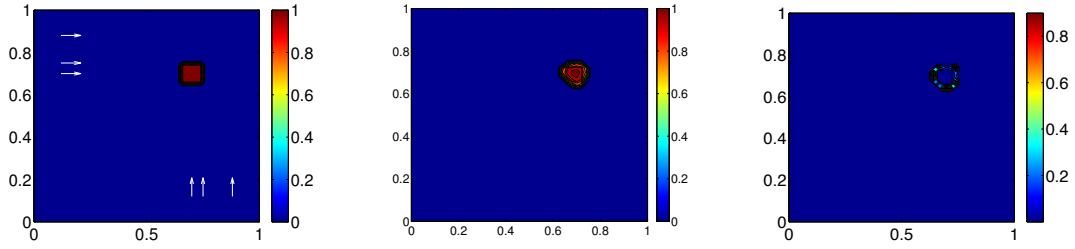
(a) Actual $K(\vec{x})$ (b) Estimated $K(\vec{x})$ (c) Absolute error of estimation

Figure 3.6: $K(\vec{x})$ estimated from 4 data sets for a domain that contains a square at the center



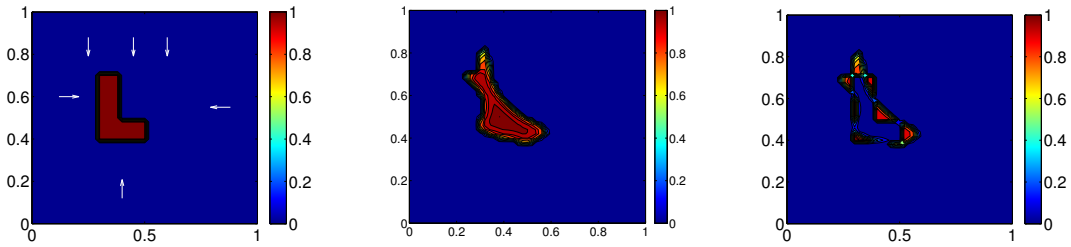
(a) Actual $K(\vec{x})$ (b) Estimated $K(\vec{x})$ (c) Absolute error of estimation

Figure 3.7: $K(\vec{x})$ estimated from 8 data sets for a domain that contains a square at the center



(a) Actual $K(\vec{x})$ (b) Estimated $K(\vec{x})$ (c) Absolute error of estimation

Figure 3.8: $K(\vec{x})$ estimated from 8 data sets for a domain that contains a square in the corner



(a) Actual $K(\vec{x})$ (b) Estimated $K(\vec{x})$ (c) Absolute error of estimation

Figure 3.9: $K(\vec{x})$ estimated from 6 data sets for a domain that contains a non-convex L shaped object

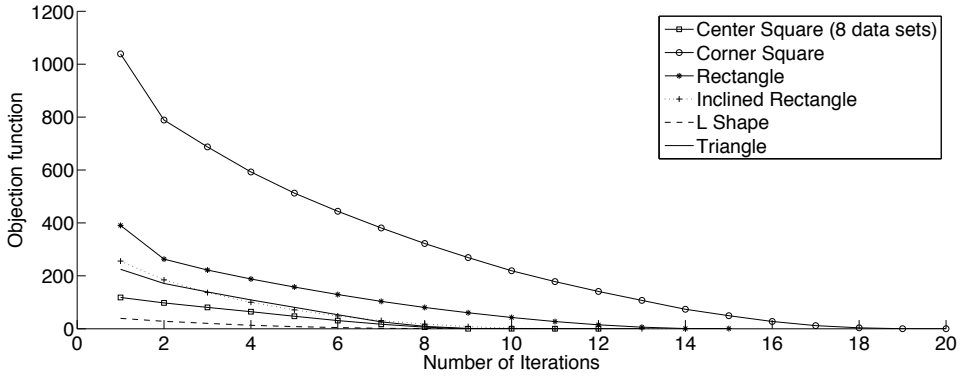


Figure 3.10: Objective function value vs. number of iterations for the different scenarios examined

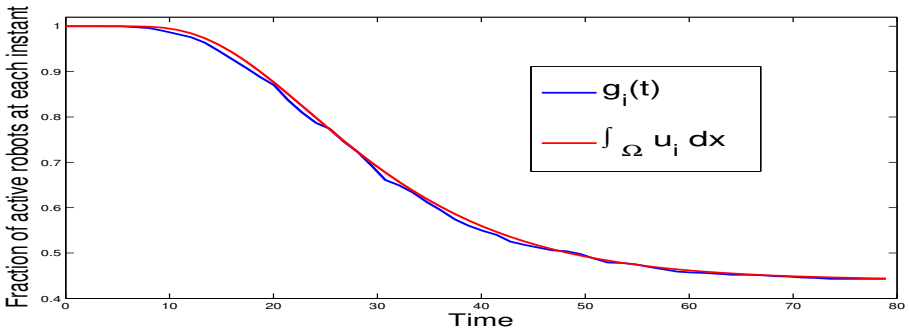


Figure 3.11: Comparison between the fraction of active robots $g_i(t)$ and $\int_\Omega u_i(\mathbf{x}, t)d\mathbf{x}$ during a particular swarm deployment i for the $K(\vec{x})$ shown in Figure 3.4a

Chapter 4

A PROBABILISTIC TOPOLOGICAL APPROACH TO QUANTIFYING ENVIRONMENTAL FEATURES

Source: Ragesh K. Ramachandran et al. [133]

Funding: NSF Award CMMI-1363499 and DARPA Young Faculty Award D14AP00054.

ABSTRACT

This chapter presents a novel automated approach to quantifying the topological features of an unknown environment using a swarm of robots with local sensing and limited or no access to global position information. The robots randomly explore the environment and record a time series of their estimated position and the covariance matrix associated with this estimate. After the robots' deployment, a point cloud indicating the free space of the environment is extracted from their aggregated data. Tools from topological data analysis, in particular the concept of persistent homology, are applied to a subset of the point cloud to construct barcode diagrams, which are used to determine the numbers of different types of features in the domain. It is demonstrate that the approach can correctly identify the number of topological features in simulations with zero to four features and in multi-robot experiments with one to three features.

The work described in this chapter, originally presented in [133], is an initial step toward constructing a map with metric information [133]. In this chapter, we present an automated method for computing the number of topological features in an unknown domain from data obtained by a swarm of inexpensive robots with local sensing, no inter-robot communication, and limited or no access to global position information. The features represent obstacles or other regions of interest that robots do not pass through. The data consist of robots' position estimates and the covariance matrices of these estimates, recorded by the robots during random exploration of the domain. The robots collect this data autonomously and independently during their deployment, without relying on input from a supervisory agent. We assume that after a set period of time, the robots navigate to an easily identifiable landmark (e.g., a beacon), where they transfer this data to a central computer. The computer then processes the data from the entire swarm to extract a point cloud that covers the domain's free space and applies tools from *Topological Data Analysis* (TDA), namely *persistent homology*, to identify the numbers of different types of topological features. Our approach scales with the number of robots and is robust to the failure of a small portion of the swarm.

Division of Work Dr. Sean Wilson conducted the experiments for Section 4.7. The author devised the method to quantify features in the environments. Simulation were coded in python by the author to simulated robots' interaction with the environment. Data obtained from these simulations were analyzed by the author in Matlab to validate the technique.

4.1 Related Work

Existing techniques for environmental mapping such as occupancy grid mapping [136], simultaneous localization and mapping (SLAM) [111, 136], and Proba-

bility Hypothesis Density (PHD) filtering [160] are not feasible to implement for our problem due to the limited sensing and computational capabilities of the robots. Although topological mapping has been extensively studied, TDA has only recently been applied in robotics for environmental characterization. For a scenario with a single robot, [25] presents a method for topological SLAM that encodes the topology of the environment in a generalized Voronoi graph. Few works address the problem of mapping an environment using a robotic swarm with limited sensing, no inter-robot communication, and no global localization. In [132], we presented an optimal control approach to mapping a GPS-denied environment with a robotic swarm using a partial differential equation model of the swarm population dynamics. This strategy works best when the domain contains only a few sparsely distributed features, whereas the approach presented here can be applied to domains that are more densely populated with features.

In [107], the authors propose an algorithm that covers the free space of the environment with robots and then constructs an approximate generalized Voronoi graph of the covered region. This algorithm requires the robots to communicate with a central server that commands their actions. In contrast, our approach does not require a centralized decision maker during the robots' operation. Alternatively, [53] obtains a simplicial approximation of a region of interest as a topological map using dual pairs of nerves that are constructed using relevant visibility and observation covers. Contrary to our strategy, [53] requires the robots to have the ability to detect and maintain a record of landmarks in the domain, such as obstacle corners and edges. The mapping approach in [34] is similar to ours in that it generates a point cloud of the domain's free region and uses persistent homology to compute topological features in the environment. However, unlike our strategy, this approach requires each robot to have an identification label that can

be recognized by other robots.

4.2 Background

Topological Data Analysis (TDA) [18] is an emerging field that aims to provide algorithmic and mathematical tools for studying topological and geometric attributes of data. The fundamental idea underlying TDA is that data has an inherent *shape* that encodes important information regarding the connectivity of the data and yields insight into its global structure. TDA exploits the mathematical framework of *algebraic topology* [60], especially the concept of persistent homology [39], to characterize the topological structure of data. In many applications, data is obtained as a *point cloud* consisting of noisy samples of an intensity map in a Euclidean space. Prominent topological features of a point cloud can be computed using TDA and presented in the form of compact representations such as *persistence diagrams* [38] and *barcode diagrams* [54]. TDA has been extensively applied to problems in computer vision and image processing [124], sensor networks [23, 55], robotics [12, 105], localization [113], and map comparison [5].

We provide a brief introduction to persistent homology, which is central to our mapping methodology. More detailed treatments of the associated theory and computations are given in [38, 73, 162]. Persistent homology is a method of analyzing homological information gathered across different scales. This technique enables the identification of topological features that are present over a large range of scales, as opposed to those which are only temporarily present (short-scale features). Homology is a robust tool that facilitates the study of *global* attributes of spaces and functions from *local* computations on noisy data. Let $\mathbf{T}$ be a topological space which admits a simplicial decomposition, then one can affiliate a sequences of vector spaces with it called *homology groups*, denoted by $H_k(\mathbf{T})$, $k = 0, 1, 2, ...$, each

of which encodes a particular topological feature of $\mathbf{T}$. In persistent homology, these features are characterized using *Betti numbers*, which are the ranks of the homology groups. These numbers are topological invariants. Intuitively speaking, the k^{th} Betti number of $\mathbf{T}$, denoted by β_k, is the rank of $H_k(\mathbf{T})$ and represents the number of independent k-dimensional cycles in $\mathbf{T}$. For example, if $\mathbf{T} \subset \mathbb{R}^2$, then β_0 is the number of connected components in $\mathbf{T}$ and β_1 is the number of holes in $\mathbf{T}$. If $\mathbf{T} \subset \mathbb{R}^3$, then β_0, β_1, and β_2 are the numbers of connected components, tunnels, and voids in $\mathbf{T}$, respectively.

In a typical TDA application, a finite set of samples from a metric space $\mathbf{M}$ is available. These samples, along with the metric associated with $\mathbf{M}$, comprise the point cloud $\mathbf{C}$ of the space. In TDA, the metric is used to map $\mathbf{C}$ onto a collection of simplices called a simplicial complex. Simplices are combinatorial objects constructed from the subsets of $\mathbf{C}$. A k-simplex $\sigma = [v_0, v_1, ..., v_k]$ is an ordered list of $k+1$ elements $\{v_0, v_1, ..., v_k\} \in \mathbf{C}$, called vertices. The simplicial complex provides a discrete representation of the underlying topological space using a combinatorial structure that can be represented algebraically using linear operators (matrices). It is this combinatorial structure that permits us to develop algorithms for homological computation. There are various ways to build a simplicial complex from a point cloud. The simplest way is to choose a parameter $\delta > 0$ and add a k-simplex to the simplicial complex if every vertex in the simplex is within a distance δ from every other. The simplicial complex constructed in this manner is called the Vietoris-Rips complex [52] or Rips complex for short, often denoted as $Rips(\mathbf{C}, \delta)$.

For large datasets, the number of simplices in the simplicial complex can be enormous, making the computations highly inefficient. We reduce the computational requirements by choosing a subset of the point cloud consisting of *landmark points*, denoted by $\mathbf{L} \subset \mathbf{C}$, as vertices for the Rips complex. These landmark points were

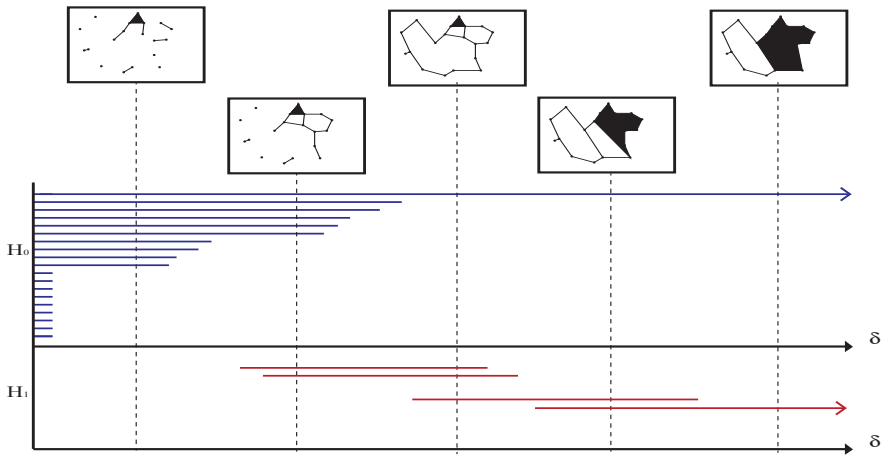


Figure 4.1: An example barcode diagram of a filtration formed from a Rips complex. $\beta_k(\delta_i)$ is the number of horizontal segments in the barcode for $H_k(\mathbf{T})$ that intersect the dashed line at $\delta = \delta_i$. The arrows in H_0 and H_1 indicate the persistent topological features. The shaded regions contain the 2-simplices (triangles).

selected using a greedy inductive selection process called a sequential max-min algorithm [2]. In order to compute persistent Betti numbers, we require a *filtration*, defined as a family of $Rips(\mathbf{C}, \delta)$ parametrized by δ such that $Rips(\mathbf{C}, \delta_1) \subseteq Rips(\mathbf{C}, \delta_2)$ for all $\delta_1 > 0$, $\delta_2 > 0$ where $\delta_1 \leq \delta_2$.

The persistent topological features of $\mathbf{T}$ over multiple values of δ can be identified using a *barcode diagram*, which is a graphical representation of $H_k(\mathbf{T})$ in terms of the homology generators. The Betti number β_k gives the number of generators of $H_k(\mathbf{T})$. A barcode plots a set of horizontal line segments on a graph whose x-axis spans a range of δ values and whose y-axis depicts an arbitrary ordering of homology generators. The numbers of arrows in the barcode for dimension 0 and dimension 1 indicate the numbers of connected components and features in the domain, respectively. A barcode diagram can be computed automatically using algorithms that find the homology generators of the homology that is constructed

on a point cloud. Figure 4.1 illustrates a barcode diagram that is obtained from an example point cloud.

4.3 Problem Statement

We consider a scenario in which N robots are deployed into a bounded, unknown, GPS-denied 2D environment in order to collect data that can be used to determine the number of topological features in the domain. The robots have local sensing capabilities and can identify features and other robots at distances within their sensing range to perform collision avoidance maneuvers. Each robot is equipped with a compass and wheel encoders, which enable it to estimate its position and orientation with uncertainty.

The robots perform correlated random walks in the domain, avoiding features and other robots. During its motion, each robot estimates its position in a global reference frame using its onboard odometry and a Kalman filter. Methods for estimating position using noisy odometry motion models and various types of sensor information have been studied extensively in the literature [142]. At fixed time intervals, the robot records its estimated position and the covariance matrix corresponding to the uncertainty of the estimate. After a time span T, all robots travel to a common location where their stored data is retrieved and processed. We assume that T is sufficiently large for the robots to thoroughly cover the domain and that the robots have sufficient memory to store the data that they obtain during their deployment.

The robots follow the motion model described in [142]. Each robot has a constant translational speed v and an orientation $\theta(t)$ at time t with respect to a global frame. We define a robot's velocity vector at time t as $\mathbf{V}(t) = [v_x(t),\ v_y(t)]^T = [v\ \cos(\theta(t)),\ v\ \sin(\theta(t))]^T$ and its position vector as $\mathbf{X}(t) = [x(t),\ y(t)]^T$. The displace-

ment of a robot over a time step Δt is given by

$$\mathbf{X}(t + \Delta t) = \mathbf{X}(t) + \mathbf{V}(t)\Delta t + \mathbf{W}(t), \tag{4.1}$$

where $\mathbf{W}(t) \in \mathbb{R}^2$ is a vector of independent, zero-mean normal random variables that are generated at time t to model the randomness in the robot's motion due to sensor and actuator noise. At the beginning of a time step, each robot generates a random number between 0 and 1. If this number is below a predefined threshold p_{th}, the robot randomly chooses a new $\theta(t) \in [-\pi, \pi]$. At time $t = 0$, the start of a deployment, each robot is assigned the parameters v and p_{th} and obtains accurate measurements of its position $\mathbf{X}(0)$ and orientation $\theta(0)$.

We consider two types of scenarios. In **Type I** scenarios, robots receive accurate estimates of their global positions when they are close to the boundary of the domain. For example, robots on the exterior of a building will have access to GPS measurements that are unavailable to robots inside. In **Type II** scenarios, robots do not receive global position updates anywhere in the domain, which may for instance be located underground or underwater.

4.4 Feature Extraction Methodology

During a deployment, the data that robot $j \in \{1, ..., N\}$ obtains at time $t_k \in [0, T]$, $k \in \{1, ..., K\}$, consists of the element $d_k^j = \{\mu_k^j, \Sigma_k^j\}$, where $\mu_k^j \in \mathbb{R}^2$ is the mean of the robot's estimate of its (x, y) position at time t_k, and $\Sigma_k^j \in \mathbb{R}^{2\times 2}$ is the covariance matrix of its position estimate at this time. In this section, we present a three-step methodology for extracting the topological features of the domain from this data.

In the first step, we discretize the domain into a high-resolution uniform grid, as in the occupancy grid mapping algorithms described in [142]. Let m_i denote the grid cell with index $i \in \{1, ..., M\}$ and $\mathbf{M} = \{m_i\}$ denote the set of all grid cells. The

goal of this step is to use the robots' data to assign each grid cell m_i a probability p_i^f of being *free*, or unoccupied by a topological feature. Toward this end, we compute p_{ijk}, the probability that robot j occupied grid cell m_i at time t_k, for all robots, cells, and measurement times. This probability is obtained by numerically integrating the Gaussian distribution with mean μ_k^j and covariance matrix Σ_k^j over the region $[x_i^l, x_i^u] \times [y_i^l, y_i^u]$ occupied by the cell:

$$p_{ijk} = \int_{y_i^l}^{y_i^u} \int_{x_i^l}^{x_i^u} \mathcal{N}(\mu_k^j, \Sigma_k^j)\, dx\, dy \tag{4.2}$$

Next, we assign a score $s_i \in [0, \infty)$ to each grid cell m_i according to the formula

$$s_i \;=\; \sum_{j=1}^{N} \sum_{k=1}^{K} \log\left(\frac{1}{1 - p_{ijk}}\right) \tag{4.3}$$

We rescale each score s_i to a value $s_i^C \in [0, C]$, where C is chosen such that the value of $1 - \exp(C)^{-1}$ is close to one. This rescaling improves numerical stability when converting the scores to probabilities, especially values near zero and one. Finally, the probability of each grid cell being free is computed as $p_i^f \;=\; 1 - \exp(s_i^C)^{-1}$.

4.5 Point Cloud Extraction and Landmark Selection

In the second step, we extract a point cloud $\mathbf{C}$ and select a subset $\mathbf{L}$ of these points as landmark points. The procedure for this step is summarized in algorithm 1.

The point cloud is constructed by sampling a dense, uniformly random set of points from the domain and rejecting those points that are inside a grid cell m_i for which p_i^f is below a given threshold (i.e., there is a high probability of cell m_i being occupied by a feature). In this work, we set the threshold heuristically. Landmark points are selected from the point cloud using the sequential max-min algorithm [2]. This algorithm initially chooses a random point in $\mathbf{C}$ as the first landmark. Given a set of $i - 1$ landmarks denoted by $\mathbf{L}_{i-1}$, the algorithm selects the i^{th} landmark as

Algorithm 1 Point cloud computation

Input: $\{p(m_i)\}$, NO_OF_SAMPLES - no of samples, THRESHOLD - threshold for rejection.

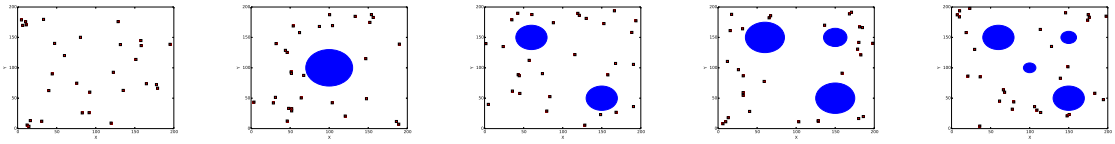
Output: point_cloud - point cloud.

Global Variables: domain = {x0, y0, length, width} - Structure containing domain origin and dimensions

1: **function PCLOUD**($\{p(m_i)\}$, NO_OF_SAMPLES, THRESHOLD)
2: point_cloud = [domain.x0 + domain.length * rand(NO_OF_SAMPLES,1), domain.y0 + domain.width * rand(NO_OF_SAMPLES,1)] ▷ Random sampling of the domain
3: **for** i = 0 to NO_OF_SAMPLES - 1 **do**
4: prob = getProb($\{p(m_i)\}$, point_cloud[i,:])
5: **if** prob < THRESHOLD **then**
6: delete point_cloud[i,:] ▷ deleting the point which is probably not free
7: **end if**
8: **end for**
return point_cloud
9: **end function**

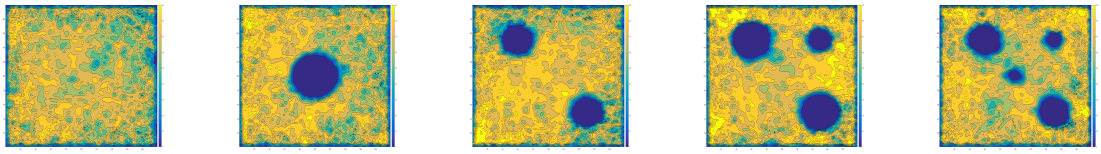
the point $c \in \mathbf{C}$ that maximizes the function $d(c, \mathbf{L}_{i-1}) = \min\{\|c - l\| : l \in \mathbf{L}_{i-1}\}$. The landmarks chosen in this manner tend to cover the point cloud.

Finally, we use the landmark points to construct a filtration using the tools discussed in Section 4.2, and we extract barcode diagrams from this filtration. We chose the Rips complex as a basis for constructing the filtration [54] and used the MATLAB-based JavaPlex package [3] to perform all persistent homology computations and generate the barcodes. We computed persistent homology only for



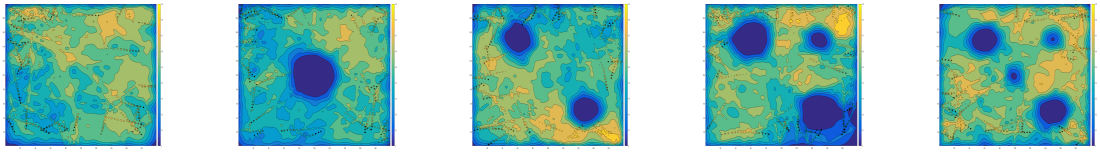
(a) No feature (b) One feature (c) Two features (d) Three features (e) Four features

Figure 4.2: Snapshots of a simulated swarm moving through different domains.



(a) No feature (b) One feature (c) Two features (d) Three features (e) Four features

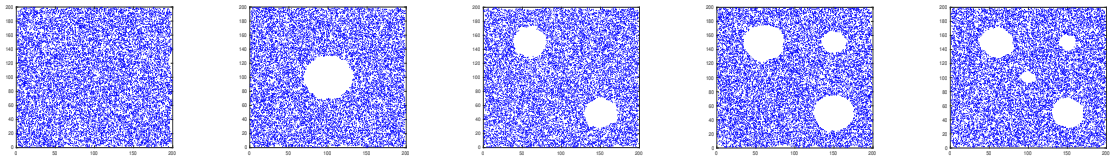
Figure 4.3: Contour plots of p_i^f, the probability that grid cell m_i is free, over all grid cells of discretized domains in **Type I** scenarios. Colorbar values range from 0 to 0.9.



(a) No feature (b) One feature (c) Two features (d) Three features (e) Four features

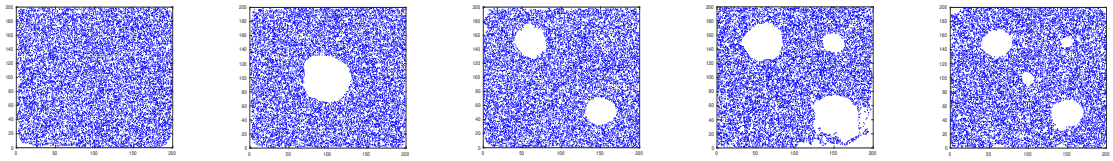
Figure 4.4: Contour plots of p_i^f, the probability that grid cell m_i is free, over all grid cells of discretized domains in **Type II** scenarios. Colorbar values range from 0 to 0.9.

dimensions zero and one, since higher dimensions are not relevant for our application.



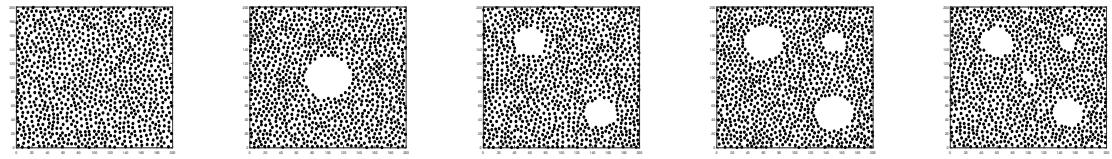
(a) No feature (b) One feature (c) Two features (d) Three features (e) Four features

Figure 4.5: Point clouds computed over domains in **Type I** scenarios.



(a) No feature (b) One feature (c) Two features (d) Three features (e) Four features

Figure 4.6: Point clouds computed over domains in **Type II** scenarios.

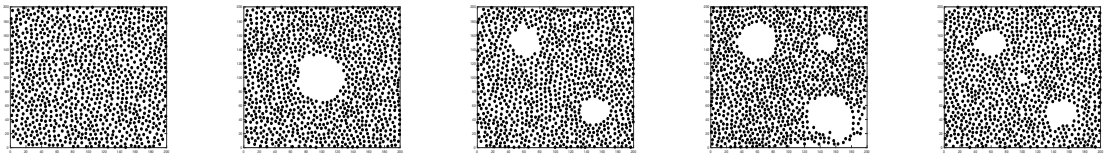


(a) No feature (b) One feature (c) Two features (d) Three features (e) Four features

Figure 4.7: Landmark points selected over domains in **Type I** scenarios.

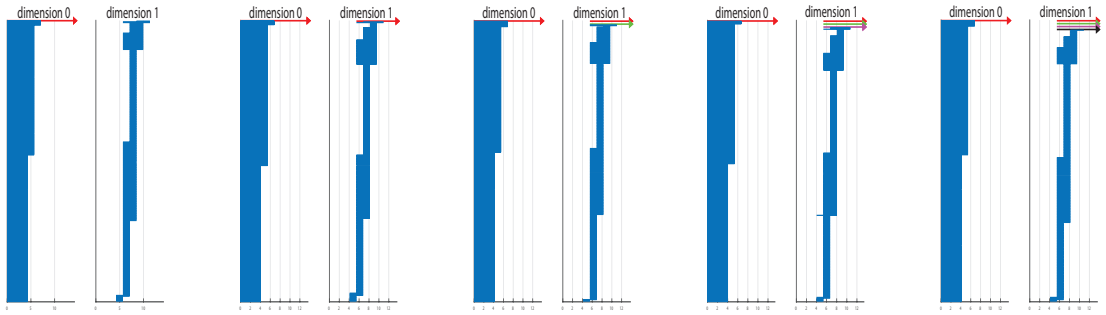
4.6 Simulations

We applied the methodology described in Section 4.4 to estimate the number of topological features in the simulated environments shown in Figure 4.2 for both **Type I** and **Type II** scenarios. The simulations were coded in Python, and all other computations were performed in MATLAB. The simulated swarm consisted of 30



(a) No feature (b) One feature (c) Two features (d) Three features (e) Four features

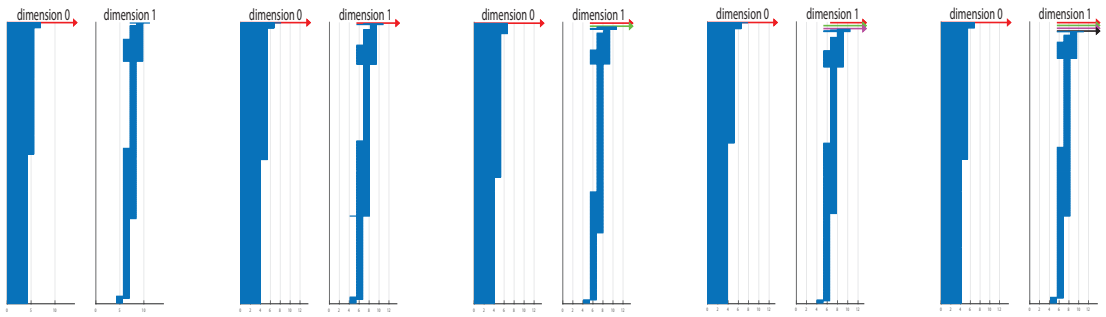
Figure 4.8: Landmark points selected over domains in **Type II** scenarios.



(a) No feature (b) One feature (c) Two features (d) Three features (e) Four features

Figure 4.9: Barcodes computed for domains in **Type I** scenarios.

point robots, each with a sensing radius of 5 cm, an average speed of $v = 20$ cm/sec, and $p_{th} = 0.2$. The robots explored a 200 cm $\times$ 200 cm domain over a time period $T = 200$ sec. At the start of each simulation, the robots were placed at random locations near the domain boundary. The robots dispersed throughout the domain according to the model Equation 4.1, where the covariance matrix of the random variables in $\mathbf{W}(t)$ was set to be a diagonal matrix with 0.1 on the diagonal. Upon encountering a feature, the domain boundary, or another robot within a distance of 5 cm, a robot would randomly choose a different direction to avoid a collision. After each simulated swarm deployment, we randomly sampled 16,000 points from the



(a) No feature (b) One feature (c) Two features (d) Three features (e) Four features

Figure 4.10: Barcodes computed for domains in **Type II** scenarios.

domain, extracted a point cloud **C** by using a threshold of 0.2 for p_i^f, and selected a set **L** of 1,000 landmark points from **C**. The maximum filtration value (maximum value of δ) used for the barcode computation was heuristically chosen to be 3Δ, where $\Delta = \max\{d(c, \mathbf{L}) : c \in \mathbf{C}\}$.

Figure 4.3 to Figure 4.10 plot the outputs of the different steps of our methodology for both **Type I** and **Type II** scenarios: contour plots of p_i^f (Figure 4.3 and Figure 4.4), point clouds (Figure 4.5 and Figure 4.6), landmark points (Figure 4.7 and Figure 4.8), and barcode diagrams (Figure 4.9 to Figure 4.10). The contour plots of p_i^f in the **Type I** scenarios are more accurate than the plots in the **Type II** scenarios, in the sense that they display higher probabilities of free space in areas that are actually unoccupied by features. However, the plots in the **Type II** scenarios do correctly estimate very low probabilities of free space in areas that are occupied by features.

The barcode arrows for both **Type I** and **Type II** scenarios give the correct numbers of connected components and features for each simulated environment. These results show that our methodology can accurately extract topological features

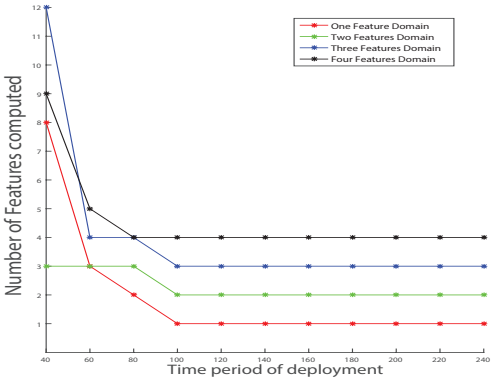


Figure 4.11: Computed number of features versus swarm deployment time period T (in seconds) for simulations with 30 robots on the domains shown in figure 4.2 for **Type II** scenarios. The effect on the approach for **Type II** scenarios when varying the time period of deployment of the swarm for various domains featured in figure 4.2. The numbers of features are computed for the data obtained from each deployment for all the four domains. In this figure the X-axis represents the time period of deployment in seconds and Y-axis represents the computed number of features in the domain.

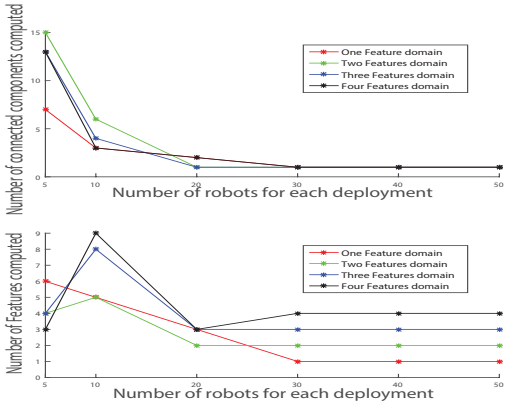


Figure 4.12: Computed numbers of connected components (*top*) and features (*bottom*) versus number of robots N for simulations with $T = 200$ sec on the domains shown in figure 4.2 for **Type II** scenarios.The effect on the approach for **Type II** scenarios when varying the number of robots in swarm for various domains featured in figure 4.2. The numbers of features are computed for the data obtained from each deployment for all the four domains. In this figure the X-axis represents the number of robots in swarm for each deployment; the Y-axis of the top and bottom figures represent the computed number of connected components and the computed number of features in the domain respectively.

even when the robots do not receive accurate estimates of their global positions.

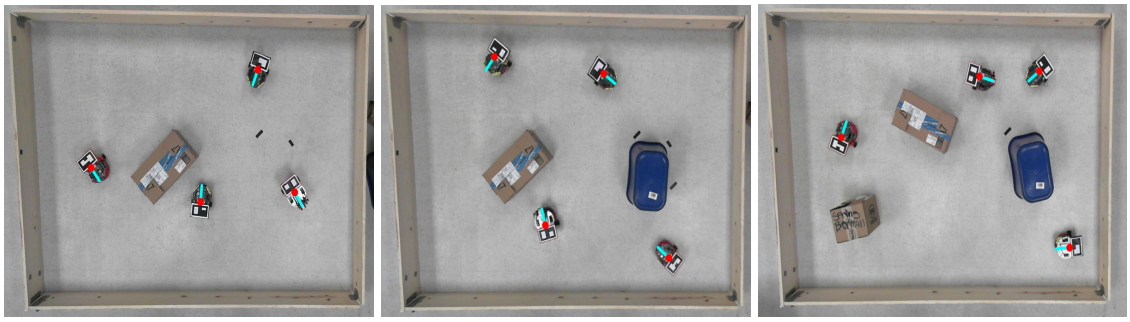
We also examined the effect of the quantity of robot position data on the accuracy of our approach for **Type II** scenarios. Larger quantities of robot data can be obtained by extending the time period T of the swarm deployment or by deploying a larger number of robots, N. We ran simulations with 30 robots over the four domains shown in Figure 4.2 with deployment times T that varied from 40 sec to 240 sec, at intervals of 20 sec. At the end of each deployment, the number of topological features was computed from the resulting barcode diagram. Figure 4.11 plots the computed number of features in each domain for every value of T. The Figure shows that the correct number of features is identified in each domain when $T \geq 100$ sec. For shorter deployments, the robots do not always cover a sufficiently large area of the domain for their recorded position data to yield an accurate count of the number of features. Hence, Figure 4.11 shows that the shortest possible time period over which the swarm should be deployed in the simulated scenarios is $T \in (80 \text{ sec}, \ 100 \text{ sec}]$. We also ran simulations over the four domains in Figure 4.2 with robot population sizes $N \in \{5, 10, 20, 30, 40, 50\}$ and $T = 200$ sec and computed the numbers of connected components and topological features in each domain. Figure 4.12 plots these numbers for every value of N and shows that they are accurate when data is obtained by $N \geq 30$ robots. In practice, such simulations can be performed to determine estimates of the minimum values of T and N that will yield accurate counts of the number of features in an environment.

We also note that the effectiveness of our approach depends on the degree of uncertainty in the robots' position estimates, as quantified by the covariance matrices associated with the position data. A large covariance indicates a highly uncertain position estimate and results in a low probability p_i^f of the corresponding grid cell being free. This low value reduces the likelihood that the central computer

will misidentify the grid cell as being free (i.e., a possible location for a robot) if it is already known to be occupied by an obstacle. In addition, since covariances in robot positions will increase over time, newly acquired position data will not result in significant changes in the value of p_i^f.

4.7 Multi-Robot Experiments

In addition to simulations, we validated our methodology through experiments with four Pheeno mobile robot platforms [154] in a **Type II** scenario with one to three features. The robots were initially placed at random locations in a 1.5×2.1 meter rectangular arena that was bounded by wooden walls, as shown in Figure 4.13. The robots were controlled to move at 10 cm/sec with an avoidance radius of 10 cm. Whenever a robot detected a feature, wall, or another robot, it avoided a collision by moving according to a specular reflection from the detected object and then continued in a straight line. The robots were marked with 2D binary identification tags to enable real-time tracking of their positions and orientations by an overhead camera (Microsoft Life Cam, resolution of 1920×1080 pixels). A control computer broadcast each robot's initial state $\mathbf{x} = [x, y, \phi]^T$ over WiFi, where x and y are the robot's position coordinates in the arena and ϕ is its heading. Each robot used an Extended Kalman Filter (EKF) to estimate its state at intervals of 200 ms. This state was updated according to a kinematic unicycle model and a measurement state vector, $\mathbf{z} = [\Delta d_e,\ \Delta\phi_e,\ \phi_c]^T$, where Δd_e is the encoders' measurement of the linear distance traveled, $\Delta\phi_e$ is the change in heading angle measured by the encoders, and ϕ_c is the orientation of the robot in the global frame measured by the compass. The state error covariance matrix $\mathbf{P}$, process covariance matrix $\mathbf{Q}$, and measurement covariance matrix $\mathbf{R}$ were set to $\mathbf{P} = diag(0.2, 0.2, 0.1)$, $\mathbf{Q} = diag(2, 2, 4)$, and $\mathbf{R} = diag(0.1, 5, 0.4)$. These matrices were chosen to favor the robot's measurements over

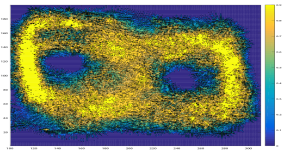


(a) (b) (c)

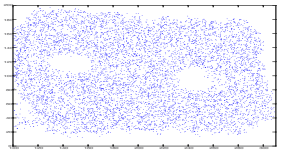
Figure 4.13: The experimental arena with four Pheeno robots and (a) one feature, (b) two features, or (c) three features. At the start of the experiment, the control computer identifies the robots' positions and orientations, indicated by the red dots and cyan lines, from the robots' 2D binary identification tags. This identification is done using the thresholding, boxpoint, and contouring OpenCV libraries on a Windows computer.

the kinematic motion model. The initial state estimate covariance was chosen to reflect errors in tag placement on the robots and camera discretization error. The EKF was implemented on Pheeno's Arduino Pro Mini microcontroller (3.3V 8MHz), while the state data and covariance matrices were stored onboard its Raspberry Pi 2 Model B.

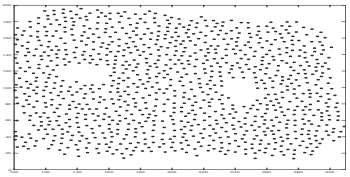
The results in Figure 4.14a - Figure 4.14d confirm that our methodology correctly extracts one connected component and two topological features from the robots' data after being deployed in the environment in Figure 4.13b. The plots in Figure 4.15 and Figure 4.16 show that given a sufficiently long deployment time T and a sufficiently large number of robots N, our approach produces accurate counts of the numbers of connected components and features in environments with one, two, and three features. There is a trade-off between the robots' deployment time and the reliability of their position data, since the EKF state estimates will drift due



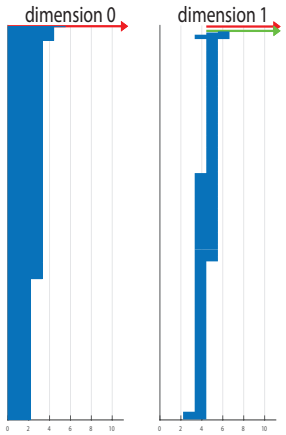
(a) Contour plot of p_i^f; colorbar values range from 0 to 0.9



(b) Point cloud



(c) Landmark points



(d) Barcode Diagram

Figure 4.14: Experimental results from a **Type II** environment containing two objects.

to the robots' wheel slip and sensor noise. These factors cause the covariances of the position estimates to eventually grow larger than the environment and thus yield no useful information for mapping. This uncertainty can be reduced in a **Type 1** scenario by correcting the drift with direct GPS measurements or with estimates of global position using local measurements of known objects in the environment. From our experiments, it is evident that larger numbers of robots yield more accurate mapping results, since there is a higher chance of robots exploring small gaps between features before the covariances of their position data grow too large to provide useful information.

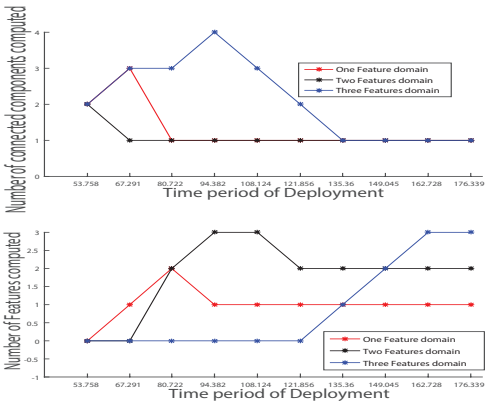


Figure 4.15: Computed numbers of connected components (*top*) and features (*bottom*) versus swarm deployment time period T (in seconds) for experiments with four robots on the domains shown in Figure 4.13 for **Type II** scenarios. Experimental validation of the approach for **Type II** scenarios when varying the time period of deployment of four robots for various domains having one to three features. The numbers of features are computed for the data obtained from each deployment for three domains. In this figure the X-axis represents the time period of deployment; the Y-axis of the top and bottom figures represent the computed number of connected components and the computed number of features in the domain respectively.

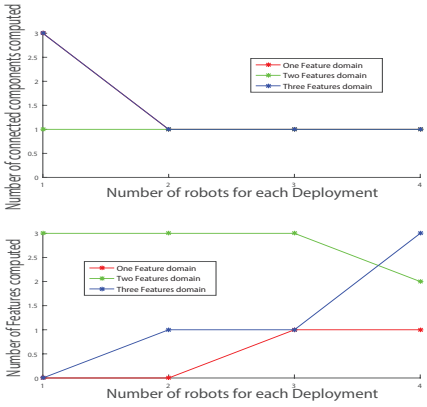


Figure 4.16: Computed numbers of connected components (*top*) and features (*bottom*) versus number of robots N for experiments with T = 180 sec on the domains shown in Figure 4.13 for **Type II** scenarios. Experimental validation of the approach for **Type II** scenarios when varying the number of robots in swarm for three minutes in various domains having one to three features. The numbers of features are computed for the data obtained from each deployment for three domains. In this figure the X-axis represents the number of robots in swarm for each deployment; the Y-axis of the top and bottom figures represent the computed number of connected components and the computed number of features in the domain respectively

Chapter 5

A PROBABILISTIC APPROACH TO CONSTRUCTING TOPOLOGICAL MAPS OF AN ENVIRONMENT

Source: Ragesh K. Ramachandran et al. [134]

Funding: NSF Award CMMI-1363499 and DARPA Young Faculty Award D14AP00054.

ABSTRACT

In this chapter, a novel procedure is presented for constructing a topological map of an unknown environment from data collected by a swarm of robots with limited sensing capabilities and no communication or global localization. Topological maps are sparse roadmap representations of environments that can be used to identify collision-free trajectories for robots to navigate through a domain. This method described in this chapter uses uncertain position data obtained by robots during the course of random exploration to construct a probability function over the explored region that indicates the presence of obstacles. Techniques from topological data analysis, in particular the concept of persistent homology, are applied to the probability map to segment the obstacle regions. Finally, a graph-based wave propagation algorithm is applied to the obstacle-free region to construct the topological map of the domain in the form of an approximate Generalized Voronoi Diagram. The effectiveness of the approach is demonstrated in a variety of simulated domains and in multi-robot experiments on a domain with two obstacles, and we conduct an analysis of its computational time complexity.

This chapter describes work that was first presented in [134]. In this work, we address the specific problem of finding safe robot trajectories through an unknown environment using uncertain localization data acquired by robots with onboard odometry. Toward this end, we use the robots' data to construct a *topological map (or graph)*, which is a sparse representation of an environment that encodes all of its topological features, such as holes that represent obstacles, and provides a collision-free path through the environment in the form of a roadmap [140]. If the domain is embedded in $\mathbb{R}^2$, then the topological map can be mathematically described as a one-dimensional *deformation retract* of the domain [52]. A topological map can be constructed in the form of a *Generalized Voronoi Diagram (GVD)*, also called a *Generalized Voronoi Graph (GVG)* [8]. A GVD provides all possible path homotopies in an environment containing obstacles and indicates the maximum clearances from obstacles and the domain boundary. In addition, since GVDs are graphs, standard graph search algorithms can be used for planning on GVDs. Due to the computational complexity of computing exact GVDs, algorithms have been developed to generate *approximate GVDs (AGVDs)* in practice [65].

In this work, we present a novel automated procedure for generating the topological map of an unknown, GPS-denied environment using data from a swarm of robots with limited sensing capabilities and no inter-robot communication. This procedure is an extension of our work in Chapter 4, which presents an approach to estimating the number of topological holes in a domain by constructing a *Rips complex* filtration [52] on its free space. As in this previous work, our procedure begins with the deployment of a swarm of robots into the environment, which they randomly explore while recording estimates of their positions and the covariance matrices of these estimates. The robots are then retrieved, and their data are used to derive a probabilistic map of the domain. We adopted a similar deployment

strategy to map GPS-denied environments in Chapter 3, which only requires robot data on encounter times with features of interest but, unlike the work presented here, is limited in application to domains with a few sparsely distributed features.

We employ tools from topological data analysis (TDA) to segment the obstacle regions in the domain by constructing a probability-based filtration on the domain's free space, thereby simultaneously computing the optimal filtration parameter and estimating the number of topological features in the domain. TDA has previously been used for super-level set estimation of probability densities [104]. Next, we use a graph-based *wave propagation algorithm* [107] to construct the topological map of the domain in the form of an AGVD. This map can then be used by humans or more expensive robots to navigate safely through the environment. The data-gathering portion of our procedure is decentralized, in that the robots act autonomously and a central supervisor is not required to control their individual operations. After this phase, a central server is needed to construct the AGVD from the collected robot data, since the robots do not have the resources to perform these computations onboard. While we only consider 2D environments in this paper, our techniques can be extended to 3D environments as well.

Division of Work Dr. Sean Wilson conducted the experiments for Section 5.8. The author devised the method to generate the topological map of the environments. Simulations were coded in python by the author to simulated robots' interaction with the environment. Data obtained from these simulations were analyzed by the author in Matlab to validate the technique.

5.1 Related Work

Most of the existing work on using robots to generate topological maps requires the robots to have sophisticated sensing and localization capabilities [25, 75]. Other

work on exploration and topological mapping uses robots with limited sensing capabilities, but does not scale well in a swarm robotic framework [144]. A similar approach to ours is presented in [107], which constructs an AGVD of a domain by combining a graph search algorithm with a coverage algorithm based on concepts from algebraic topology. Unlike our procedure, this approach requires each robot to communicate with a central server that commands its actions. In [53], a simplicial approximation of a region of interest is obtained as a topological map by constructing dual pairs of nerves using relevant visibility and observation covers. This strategy requires the robots to detect, identify, and store landmarks in the domain such as obstacle corners and edges, and therefore requires the robots to have higher sensing and processing capabilities than in our procedure. The approach in [35] generates a point cloud of the domain's free space in a coordinate-free manner and employs persistent homology to compute topological features in the domain. This strategy requires inter-robot communication, and each robot must have a unique identifier that is recognized by other robots.

5.2 Background

Topological Data Analysis (TDA) [18] provides algorithmic and mathematical tools for studying topological and geometric attributes of noisy data in a coordinate-free manner. Detailed treatments of the theoretical and computational aspects of TDA are given in [38, 73, 162, 60]. A basic introduction to TDA and *algebraic topology* is given in Section 4.2. We now revisit the concept of a *simplicial complex* in order to provide background for the subsequent sections.

A key concept in algebraic topology is the *abstract simplicial complex*. Although this complex is in general defined on arbitrary sets, here we restrict its definition to subsets of Euclidean spaces and use notation from [38]. We say that vectors

$v_0, v_1, ..., v_k \in \mathbb{R}^n$ are *affinely independent* if the vectors $v_1 - v_0, ..., v_k - v_0$ are linearly independent. A k-simplex $\sigma \subset \mathbb{R}^n$ can be defined as the convex hull of $k+1$ affinely independent vectors $\{v_0, v_1, ..., v_k\}$, called *vertices*, and is often represented as $\sigma = [v_0, v_1, ..., v_k]$. A *face* τ of the simplex σ is the convex hull of a non-empty subset of $\{v_0, v_1, ..., v_k\}$. This relationship is commonly denoted as $\tau \leq \sigma$. A *simplicial complex* κ is defined as a finite collection of simplices σ such that (1) $\sigma \in \kappa$, (2) $\tau \leq \sigma$ implies that $\tau \in \kappa$, and (3) $\sigma, \sigma' \in \kappa$ implies that $\sigma \cap \sigma'$ is empty or is a face of both σ and σ'.

Simplicial complexes provide discrete representations of an underlying topological space using a combinatorial structure, which can be expressed algebraically with linear operators (matrices). This algebraic structure can be exploited to develop algorithms for homological computations. If $f : \kappa \to \mathbb{R}$ is a function such that $\tau \leq \sigma$ implies that $f(\tau) \leq f(\sigma)$, then $f^{-1}((-\infty, \delta])$ is a simplicial complex denoted by κ_δ and $\delta_1 \leq \delta_2$ implies that $\kappa_{\delta_1} \subseteq \kappa_{\delta_2}$, yielding a *filtration of simplicial complexes* with δ as the *filtration parameter*. The persistent homology is obtained by varying the value of the filtration parameter, computing the generators of homology groups (the basis of the homology group vector spaces) for each simplicial complex obtained for the parameter value, and identifying the persistent homology generators.

5.3 Problem Statement

We consider the problem of generating a topological map of a bounded, unknown, GPS-denied 2D environment using data collected by a swarm of N robots. We assume that each robot can identify features and other robots that fall within its local sensing range, enabling it to avoid collisions, and can estimate its position and orientation with uncertainty using measurements from a compass and wheel encoders. After the swarm is deployed into the domain, each robot performs

a correlated random walk while estimating its global position using its onboard odometry and refining its measurements using a Kalman filter. At fixed time intervals, each robot records its estimated position and the associated covariance matrix corresponding to the uncertainty of the estimate. After a time T, which we assume is sufficiently large for the robots to thoroughly cover the domain, all robots travel to a common location where their stored data is retrieved.

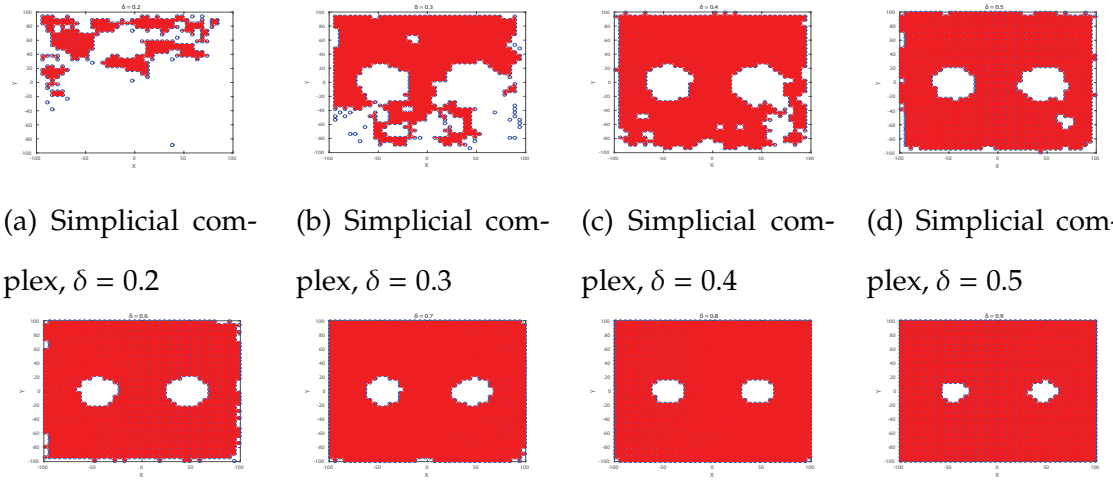
Similar to the strategy in Chapter 4, the robots follow the standard odometry motion model described in [142]. Each robot has a constant speed v and an orientation $\theta(t)$ at time t with respect to a global frame. The velocity and position vectors of a robot at time t are defined as $\mathbf{V}(t) = [v_x(t),\ v_y(t)]^T = [v\ \cos(\theta(t)),\ v\ \sin(\theta(t))]^T$ and $\mathbf{X}(t) = [x(t),\ y(t)]^T$, respectively. At time $t = 0$, the beginning of a deployment, each robot is provided with accurate measurements of its position $\mathbf{X}(0)$ and orientation $\theta(0)$. At the start of every time step Δt during the deployment, each robot generates a uniform random number between 0 and 1 and randomly chooses a new orientation $\theta(t) \in [-\pi, \pi]$ if the number is below a predefined threshold p_{th}. During a time step, the displacement of a robot is described by the equation

$$\mathbf{X}(t + \Delta t) = \mathbf{X}(t) + \mathbf{V}(t)\Delta t + \mathbf{W}(t), \tag{5.1}$$

where $\mathbf{W}(t) \in \mathbb{R}^2$ is a vector of independent, zero-mean normal random variables that are generated at time t. These variables model randomness in the robot's motion due to sensor and actuator noise.

5.4 Topological Map Generation Procedure

In this section, we present a three-step procedure for extracting a topological map of the domain in the form of an AGVD from the data obtained by the robots.



(a) Simplicial complex, $\delta = 0.2$

(b) Simplicial complex, $\delta = 0.3$

(c) Simplicial complex, $\delta = 0.4$

(d) Simplicial complex, $\delta = 0.5$

(e) Simplicial complex, $\delta = 0.6$

(f) Simplicial complex, $\delta = 0.7$

(g) Simplicial complex, $\delta = 0.8$

(h) Simplicial complex, $\delta = 0.9$

Figure 5.1: Filtration used to generate the barcode diagram Figure 5.6b for the domain shown in Figure 5.4b. The red triangles are the 2-simplices that are constructed from the centers of the grid cells in the domain discretization.

5.4.1 Estimation of the Number of Obstacles

We first discretize the domain into a high-resolution uniform grid of M cells, as in occupancy grid mapping algorithms [142], and use the robots' data to assign a probability p_i^f to each grid cell m_i, $i \in \{1, ..., M\}$, of being *free*, or unoccupied by an obstacle. We first presented this computation in Chapter 4 and summarize it here. During a deployment, the data obtained by robot $j \in \{1, ..., N\}$ at time $t_k \in [0, T]$, $k \in \{1, ..., K\}$, consists of the tuple $d_k^j = \{\mu_k^j, \Sigma_k^j\}$, where $\mu_k^j \in \mathbb{R}^2$ is the mean of the robot's estimate of its position in Cartesian coordinates at time t_k, and $\Sigma_k^j \in \mathbb{R}^{2\times 2}$ is the covariance matrix of its position estimate at this time. The probability p_{ijk} that robot j occupied grid cell m_i at time t_k is computed for all robots, cells, and measurement times. This discrete probability is calculated by integrating

the Gaussian distribution with mean μ_k^j and covariance matrix Σ_k^j over the region $[x_i^l, x_i^u] \times [y_i^l, y_i^u]$ occupied by cell m_i. A score $s_i \in [0, \infty)$ is then assigned to cell m_i according to the equation

$$s_i = \sum_{j=1}^{N} \sum_{k=1}^{K} \log\left(\frac{1}{1 - p_{ijk}}\right). \tag{5.2}$$

This score is rescaled to a value $s_i^C \in [0, C]$, where C is chosen such that the value of $1 - \exp(C)^{-1}$ is close to one. The rescaling improves numerical stability when converting the scores to probabilities, especially for values near zero and one. Finally, we compute the probability of cell m_i being free as $p_i^f = 1 - (\exp(s_i^C))^{-1}$.

Next, we identify the persistent topological features in the domain and find the *optimal threshold* α_{opt} for which all grid cells with $p_i^f < \alpha_{opt}$ belong to an obstacle. As discussed in Chapter 5, we generate a filtration of simplicial complexes based on a parameter δ in order to compute the persistent homology. Let α denote a given threshold for identifying grid cells m_i that belong to obstacles (the "obstacle grid cells"), according to $p_i^f < \alpha$. We define the filtration parameter δ as $1 - \alpha$ in order to be consistent with the conditions described in Chapter 5. Thus, the value of δ varies from 0.1 to 0.9 when the threshold α varies from 0.9 to 0.1. We construct the simplicial complex κ_δ by selecting the center points of the grid cells with $p_i^f \geq \alpha = 1 - \delta$ and constructing 1-simplices and 2-simplices from these points (the 0-simplices). The 1-simplices are generated by taking each element of the 0-simplices and pairing it with its immediate vertical, horizontal, and diagonal neighbors (8-connectivity [57]) if the neighbors are elements of the 0-simplices. Thereafter, the 2-simplices are constructed by taking every subset of three elements in the 1-simplices that form a triangle. Figure 5.1 shows the filtration constructed for the domain in Figure 5.4b, which was used in the multi-robot experiments described in Section 5.8.

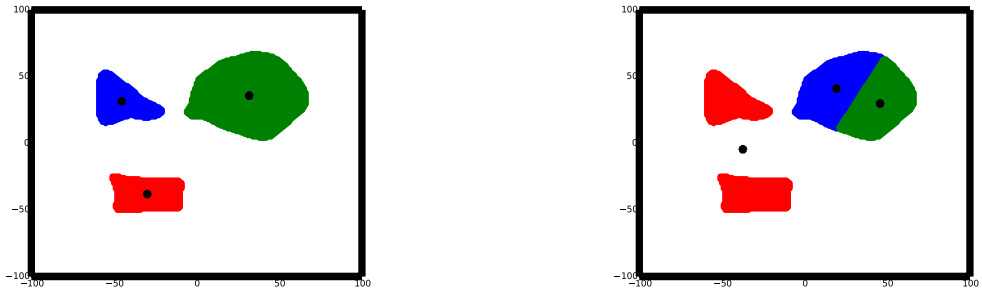
Barcode diagrams are extracted from these filtrations, and the number of barcode arrows in each homology group corresponds to the number of topological features in the domain that are encoded by that particular group. The *optimal filtration parameter* δ_{opt} is defined as the minimum value of δ for which all the topological features are captured by the corresponding simplicial complex. Alternately, it is the value of δ for which there exists no barcode segment other than arrows in any of the homology groups for all values of the filtration parameter greater than this value. Thus, the optimal threshold can be defined as $\alpha_{opt} = 1 - \delta_{opt}$. In practice, we can compute δ_{opt} by taking the maximum value of δ that is spanned by the non-arrow barcode segments in all the homology groups.

We used the MATLAB-based JavaPlex package [3] to perform the persistent homology computations and generate the barcode diagrams. Persistent homology was computed only for dimensions zero and one, since higher dimensions are not relevant for our application.

5.4.2 Obstacle Segmentation

In the second step, we use the information gathered in the previous step to segment the portion of the domain that is occupied by obstacles and identify the obstacle grid cells. By definition, the grid cells with $p_i^f < \alpha_{opt}$ belong to an obstacle. Since we have determined the number of obstacles N_O in the domain (the number of arrows in H_1 in the barcode diagram), a straightforward approach would be to use a K means clustering algorithm on the center points of these grid cells. However, since K means techniques are highly sensitive to the choice of randomly initialized points, it is difficult to guarantee the correct classification of the obstacle grid cells. This is illustrated in figure 5.2.

Instead, to classify the grid cells in each obstacle, we develop an algorithm that



(a) Correct clustering

(b) Incorrect clustering

Figure 5.2: K means clustering algorithm applied to the same point cloud twice. Different colors indicate different clusters and the filled circles are the positions of the cluster centers.

takes as input (1) the number of obstacles in the domain, and (2) an *obstacle graph* $\mathcal{G}_O$ whose vertices are the center points of the obstacle grid cells. Denoting the set of these points by C, the edges of $\mathcal{G}_O$ are defined by pairing every element in C with its immediate vertical and horizontal neighbors (4-connectivity [57]) if the neighbors are elements of C. We initialize an open list L with all the elements in C and loop through the following procedure N_O times. We choose an element randomly from L and perform a breadth-first search on $\mathcal{G}_O$ with this element as starting point. The resulting set of elements, denoted by V, consists of the grid cells contained in a single obstacle. Then L is updated by removing those elements in L which are also in V. After the obstacles are segmented in this way, the boundaries of each obstacle are identified as the elements in C that belong to the same obstacle and have fewer than four neighbors, according to $\mathcal{G}_O$. We denote the set of elements along the boundary of the i^{th} obstacle by ∂O_i. The pseudocode for this step is outlined in algorithm 2.

Algorithm 2 Obstacle Segmentation

Input: Obstacle graph $\mathcal{G}_O$ having vertex set $V(\mathcal{G}_O)$, NO_OF_OBS - number of obstacles in the domain.

Output: obstacle_segmented - list of lists containing the points in each obstacle.

```
1:  function OBSTACLE_SEGMENT(G_O, NO_OF_OBS)
2:      L = V(G_O)                                  ▷ initializing the open list
3:      for i = 0 to NO_OF_OBS - 1 do
4:    //selecting a random element from L
5:          index = (length(L) - 1)*rand()
6:       //performing Breadth first search from L[index]
7:          V = breadth_first_search(G_O, L[index])
8:       //adding V to the output list
9:          obstacle_segmented[i] = V
10:         L = L - V                               ▷ updating the open list
11:     end for
        return [obstacle_segmented, label]          ▷ label = [0,..., NO_OF_OBS - 1]
12: end function
```

5.4.3 Voronoi Diagram Construction

In the third step, we develop a new implementation of the wave propagation algorithm to generate the approximate Generalized Voronoi Diagram (AGVD). Let $\partial\mathcal{D}_j$ denote the set of grid cell centers that are closest to the j^{th} edge of the domain boundary. Since we have assumed that the domain boundary is known, we have prior knowledge of the$\partial\mathcal{D}_j$. We define a *free region graph* $\mathcal{G}_f$ whose vertex set $V(\mathcal{G}_f)$ contains the center points of grid cells that lie in the union of the obstacle-free region

with the obstacle boundary sets, ∂O_i. This graph is constructed in the same manner as the obstacle graph in Section 5.4.2. We also define the sets $\{\mathcal{B}_k\} = \{\partial O_i\} \cup \{\partial \mathcal{D}_j\}$ for each obstacle i and each boundary edge j. The distance between $v, w \in V(\mathcal{G}_f)$ is given by a known function $dist(v, w)$, which is based on the discretization of the domain.

The pseudocode of the algorithm is outlined in algorithm 3[107]. The inputs to the algorithm are the free region graph $\mathcal{G}_f$ and the set $\{\mathcal{B}_k\}$. The algorithm starts by initializing an open list with the vertices of $\mathcal{G}_f$. Then the src-value of each element in $\{\mathcal{B}_k\}$, defined as the distance from the element to the closest obstacle or domain boundary edge, is set to zero and a label is assigned to it based on which $\mathcal{B}_k \in \{\mathcal{B}_k\}$ it belongs to (lines 7-14). The remainder of the algorithm is a modified form of Dijkstra's algorithm [120]. Until the open list is non-empty, at every iteration we choose the element with the minimum src-value from the open list and check whether any of the neighbors of this element can be a part of the AGVD (lines 21-25). We update the src-values of the neighbors of this element and copy its label to its neighbors. The algorithm outputs the set of vertices in $V(\mathcal{G}_f)$, which constitutes the topological map of the domain in the form of a discrete AGVD. Figure 5.3 illustrates the progress of the algorithm when it is applied to the domain in Figure 5.4b.

5.5 Computational Complexity

The computational complexity of an algorithm is a key factor in determining its feasibility for real-time implementation. Here, we analyze the complexity of each of the main computational blocks in our procedure. From our analysis, we conclude that the worst-case complexity of our approach is $O(M^{2.372})$, which is same as the worst-case computational complexity of the method in [35], a state-of-the-

Algorithm 3 Topological Map Generation

Input: Free region graph $\mathcal{G}_f$ with vertex set $V(\mathcal{G}_f)$, $\{\mathcal{B}_k\} = \{\partial\mathcal{O}_i\} \cup \{\partial\mathcal{D}_i\}$

Output: gvd_nodes: subset of $V(\mathcal{G}_f)$ which constitutes the discrete AGVD (topological map)

$L = V(\mathcal{G}_f)$ ▷ Initialize the open list

▷ L is a min heap with ∞ priority $\forall$ elements at start

src[v] = ∞ for all v $\in L$ ▷ Distance to obstacles and domain boundaries

label[v] = -1 for all v $\in L$ ▷ Label variable

gvd_nodes = $\emptyset$

mark = 0

for $\mathcal{B}_k \in \{\mathcal{B}_k\}$ **do**

  for v $\in \mathcal{B}_k$ **do**

    label[v] = mark ▷ Label the obstacle and domain boundaries

    src[v] = 0 ▷ Set distance to 0

    L[v] = 0 ▷ Decrease priority of v

  end for

  mark = mark + 1

end for

while $L \neq \emptyset$ **do**

  n = $\arg\min_{\hat{n} \in L}$ src[$\hat{n}$] ▷ Min element of L

  if src[n] == ∞ **then**

    break

  end if

20: $L = L - \text{n}$ ▷ Remove the min element

21: $\text{q} = \arg\min_{\hat{q} \in \text{nbh(n)}} \{\text{src}[\hat{q}] \mid \text{label}[\hat{q}] \neq -1 \;\&\; \text{label}[\hat{q}] \neq \text{label[n]}\}$

22: ▷ Find element in nbh, the expanded neighbor set, with a different label

23: **if** (src[q] + 1 == src[n]) or (src[q] == src[n]) **then**

24: add q to gvd_nodes ▷ Vertex of AGVD

25: **end if**

26: $\text{U} = \{\text{u} \in \text{nbh(n)} \mid \text{u} \in L,\ \text{src[u]} > \text{src[n]} + \text{dist(n,u)}\}$

27: **for** u ∈ U **do**

28: src[u] = src[n] + dist(n,u) ▷ Update distance

29: label[u] = label[n] ▷ Assign same label to neighbor

30: L[u] = src[n] + dist(n,u) ▷ Change priority

31: **end for**

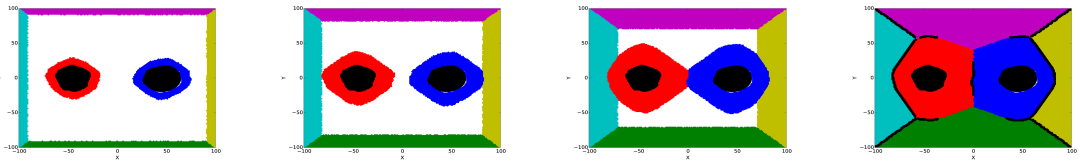
32: **end while**

33: **return** gvd_nodes

art mapping technique for a swarm of resource-constrained robots with stochastic motion and no global localization.

5.5.1 Probability map generation

In order to generate the probability map of the domain, we need to process the data from each robot for every grid cell. Therefore, if N robots each collect K items of data over a domain that is discretized into M grid cells, and we use this data to compute the probability map, then the cost of computation for this block is $O(NKM)$. This cost can be reduced by processing the data from each robot in parallel.



(a) Waves starting to propagate from different edges and obstacles are marked with different colors.

(b) Waves from opposite edges of the domain start to collide with waves from the obstacles

(c) Waves propagate further

(d) AGVD of the domain is constructed from the wave intersection regions (marked in black)

Figure 5.3: Stages of the wave propagation algorithm for constructing an AGVD of the domain shown in Figure 5.4b.

5.5.2 Simplicial complex construction and barcode generation

The simplicial complex generated from the centers of the M grid cells will have a size proportional to M. The worst-case computational complexity of persistent homology is $O(M^{2.372})$, but for most practical applications it is close to $O(M)$ [40].

5.5.3 Obstacle segmentation

The most computationally expensive part in this block is the breadth-first search (BFS) performed on the obstacle graph. The computational complexity of BFS on a graph with V vertices and E edges is $O(V + E)$ [120]. Since the obstacle graph is constructed from a subset of the grid cells of the domain, the number of vertices in the obstacle graph will be a constant factor times M. Thus, the resulting cost becomes $O(M + E)$. Since the obstacle graph is planar, the number of edges will be a constant factor times V, reducing the cost to $O(M)$.

5.5.4 AGVD extraction using the wave propagation algorithm

The cost of this block can be evaluated by analyzing algorithm 3. The most computationally expensive part of the algorithm is the loop from lines 15 to 32. As before, the number of vertices in the free region graph is a constant factor times M. Extracting the minimum element in line 21 will cost $O(\log M)$ due to the heap implementation [120]. During each iteration, a vertex is popped out, and the loop ends when all the vertices are popped. Since the statements inside the loop each have a sub-linear cost, the overall cost of this block is $O(M \log M)$.

5.6 Comparison to Other Mapping Algorithms

Table 5.1 compares key properties of our approach to those of several existing probabilistic sparse map methods. The properties of these methods are described as in [138]. In the table, the *uncertainty* field states how uncertainty is represented in the resulting map. The *convergence* field describes the convergence properties of the algorithms under suitable assumptions. The *incremental* field indicates whether an algorithm can build the map incrementally or not. The *correspondence* field specifies whether the method can accommodate mapping similar features in the environment. Lastly, the *handles raw data* field states whether the method can construct maps from raw sensor data, or whether the data first requires pre-processing and filtering.

5.7 Simulations

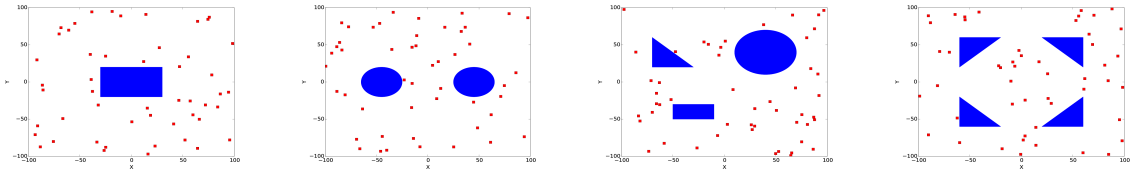
We applied the procedure described in Section 5.4 to generate the topological maps of the simulated environments shown in figure 4.2. All computations and simulations except for persistent homology computations were done in Python.

Table 5.1: Comparison of our approach to several probabilistic sparse map generation methods described in [138]

	Kalman	**Hybrid**	**Occupancy Grid**	**Dogma**	**Our Approach**
Map representation	landmark locations	point obstacles	occupancy grids	occupancy grids	occupancy grids
Robot sensor noise	Gaussian	any	any	any	any
Requires exact robot poses	no	no	yes	yes	no
Uncertainty	posterior poses and map	maximum likelihood map	posterior map	posterior map	posterior map
Convergence	strong	no	strong	weak	weak
Incremental	yes	yes	yes	no	yes
Correspondence	no	yes	yes	yes	yes
Handles raw data	no	yes	yes	yes	yes

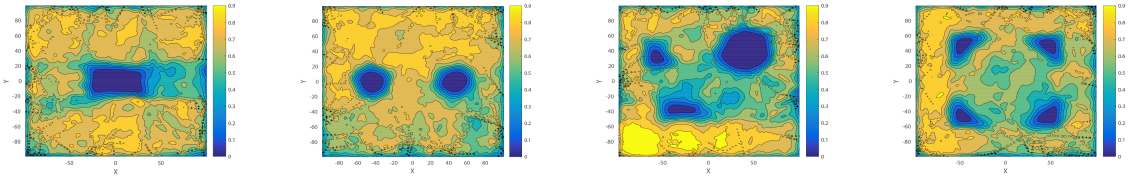
The simulated robotic swarm consisted of 50 point robots, each with a sensing radius of 0.06 m, an average speed of $v = 0.2$ m/s, and $p_{th} = 0.2$. The robots explored a 2 m $\times$ 2 m domain over a time period of 160 s. At the beginning of each simulation, the robots were placed at random positions near the domain boundary. The robots followed the motion model Equation 5.1 while dispersing throughout the domain, with the covariance matrix of the random variables in $\mathbf{W}(t)$ set to a diagonal matrix with 0.1 on the diagonal. Upon encountering an obstacle, the domain boundary, or another robot within its sensing radius, a robot randomly chose a different direction to avoid a collision. The robot data obtained after each simulated swarm deployment was used to construct the AGVD of the domain.

The outputs at different stages of the procedure for each domain are displayed



(a) One obstacle (b) Two obstacles (c) Three obstacles (d) Four obstacles

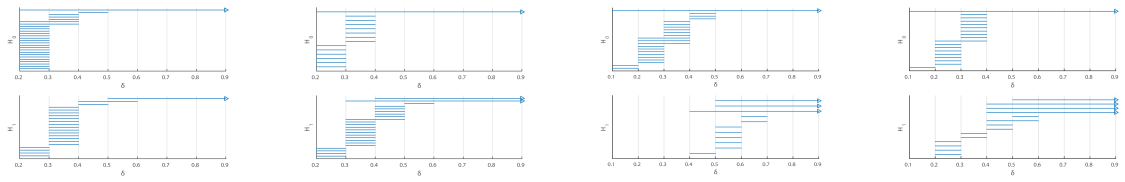
Figure 5.4: Snapshots of a simulated swarm moving through different domains.



(a) One obstacle (b) Two obstacles (c) Three obstacles (d) Four obstacles

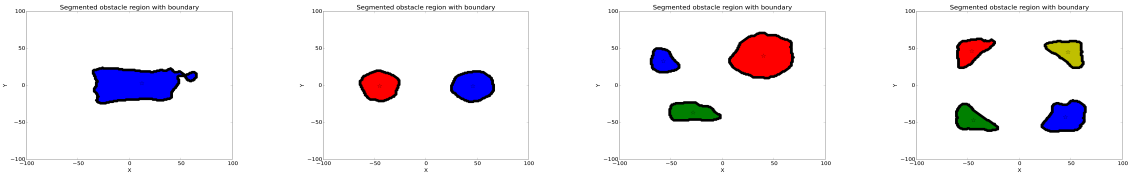
Figure 5.5: Contour plots of p_i^f, the probability that grid cell m_i is free, over all grid cells of discretized domains. Colorbar values range from 0 to 0.9.

in Figure 5.5–Figure 5.8: the contour plots of p_i^f (Figure 5.5), the barcode diagrams (Figure 5.6), the obstacle segmentation (Figure 5.7), and the computed AGVD (Figure 5.8). As mentioned previously, the numbers of arrows in the barcode diagram for dimension 0 (H_0) and dimension 1 (H_1) indicate the numbers of connected components and features (obstacles) in the domain, respectively. The results in Figure 5.6 estimate the correct number of topological features and report the value of δ_{opt} for each case. Figure 5.7 demonstrates that the obstacle segmentation technique described in Section 5.4.2, based on the α_{opt} obtained from Figure 5.6, successfully identifies each distinct obstacle in the domains. Finally, Figure 5.8 displays the AGVD topological maps generated for each domain, which display collision-free trajectories among the obstacles as expected. These results show that our procedure can accurately construct topological maps for different scenarios, even though it



(a) One obstacle; $\delta_{opt} = 0.6$ (b) Two obstacles; $\delta_{opt} = 0.6$ (c) Three obstacles; $\delta_{opt} = 0.7$ (d) Four obstacles; $\delta_{opt} = 0.6$

Figure 5.6: Barcode diagram for each domain in Figure 5.4, generated from the filtration described in Section 5.4.1, with δ_{opt} computed for each case.

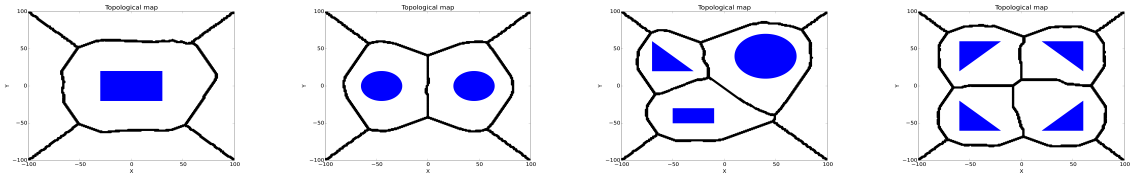


(a) One obstacle (b) Two obstacles (c) Three obstacles (d) Four obstacles

Figure 5.7: Obstacles in each domain, segmented based on the probabilities p_i^f using values of α_{opt} obtained from the barcode diagrams in Figure 5.6. Each colored region represents the interior of an obstacle, with the obstacle boundary marked in black.

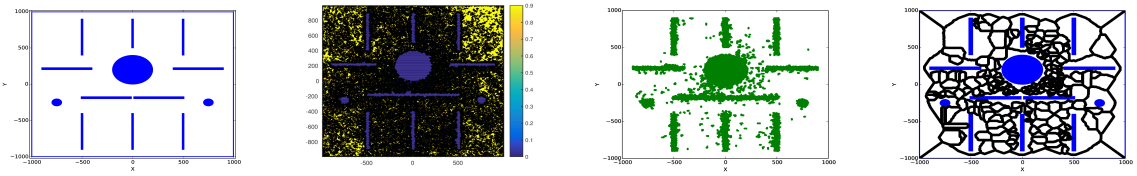
relies on uncertain robot position data.

To study the failure cases of our approach, we also simulated a complex 20 m $\times$ 20 m domain explored by 300 point robots for 200 s. All other parameters were the same as in the previous simulations. The results are presented in Figure 5.9. The topological map generated from the robot data does not reveal the narrow gap between the two rectangular obstacles in the center of the domain, since there is a low probability of robots passing through the gap while recording localization estimates. Also, the large size of the domain prevents the robots from obtaining reliable localization data about certain regions before their odometry noise become too high for the associated robot position data to provide any useful information.



(a) One obstacle (b) Two obstacles (c) Three obstacles (d) Four obstacles

Figure 5.8: Obstacles and AGVD (black lines) constructed from the probabilities p_i^f in each domain.



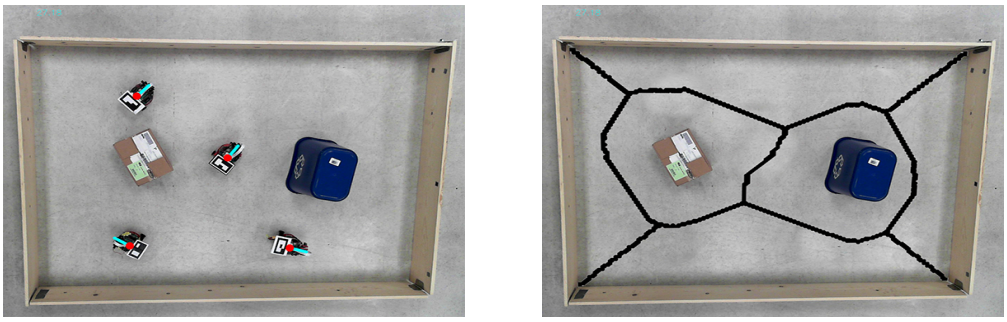
(a) Actual map (b) Contour plot of p_i^f; colorbar values range from 0 to 0.9 (c) Segmented obstacles (d) Topological map

Figure 5.9: Simulation results for a large, complex environment.

As Figure 5.9c shows, the lack of reliable data about these regions causes the procedure to incorrectly identify free space in these regions as obstacles. Although the resulting map in Figure 5.9d does not accurately represent the deformation retract of the domain, it does provide a conservative set of collision-free trajectories for the robots. A possible way to obtain accurate topological maps over large domains is to construct the maps locally and patch them together, similar to the approach in [35].

5.8 Multi-Robot Experiments

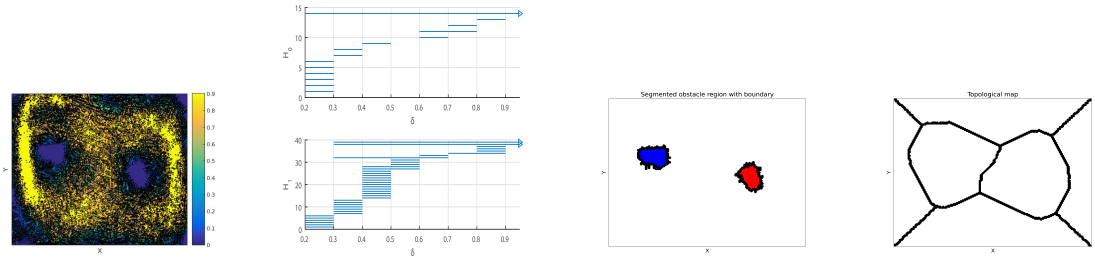
We further validated our procedure through experiments with four Pheeno mobile robot platforms [154] that explored a 1.5×2.1 meter rectangular arena with



(a) (b)

Figure 5.10: (a) (This figure first appeared in [133].) The experimental arena with four Pheeno robots and two obstacles (the box and the recycling bin). The control computer identifies the robots' configurations from their 2D binary identification tags. This identification is done using the thresholding, boxpoint, and contouring OpenCV libraries on a Windows computer. Each robot's position and orientation are displayed with the red dots and cyan lines. These positions and orientations are broadcast to the robots at the start of each experiment so that the robots have a unified global reference frame. (b) Topological map overlaid on the experiment arena.

two features. We first analyzed this experimental data in [133] using the approach to topological feature identification presented in that paper. The robots' initial position states were randomly assigned, as shown in Figure 5.10a. The robots were controlled to move with a linear velocity of 10 cm/s with an avoidance radius of 20 cm. Whenever a robot detected a feature, wall, or another robot, it avoided a collision by moving according to a specular reflection from the detected object and then continued in a straight line. The robots were marked with 2D binary identification tags to enable real-time tracking of their positions and orientations by an overhead camera (Microsoft Life Cam, resolution of 1920×1080 pixels). A



(a) Contour plot of p_i^f; colorbar values range from 0 to 0.9

(b) Barcode from filtration, yielding $\delta_{opt} = 0.9$

(c) Segmented obstacles

(d) Topological map (AGVD)

Figure 5.11: Multi-robot experimental results.

control computer was used to identify each robot's initial configuration in a global reference frame. The state vector, $\mathbf{x} = [x, y, \phi]^T$, was broadcast over WiFi, where x and y are the robot's position coordinates in the arena and ϕ is its heading in the global frame. Each robot used an Extended Kalman Filter (EKF) to update the estimate of its state at intervals of 200 ms.

The EKF was implemented using a kinematic unicycle motion model for each robot and a measurement state vector, $\mathbf{z} = [\Delta d_e,\ \Delta\phi_e,\ \phi_c]^T$, where Δd_e is the encoders' measurement of the linear distance traveled, $\Delta\phi_e$ is the change in heading angle measured by the encoders, and ϕ_c is the orientation of the robot in the global frame measured by the compass. The state error covariance matrix $\mathbf{P}$, process covariance matrix $\mathbf{Q}$, and measurement covariance matrix $\mathbf{R}$ were set to $\mathbf{P} = diag(0.2, 0.2, 0.1)$, $\mathbf{Q} = diag(2, 2, 1)$, and $\mathbf{R} = diag(0.05, 10, 0.4)$.

These matrices were chosen to favor the robot's measurements over the kinematic motion model. The initial state estimate covariance was chosen to reflect errors in tag placement on the robots and camera discretization error. The EKF was implemented on Pheeno's Arduino Pro Mini microcontroller (3.3V 8MHz),

while the state data and covariance matrices were stored onboard its Raspberry Pi 2 Model B. The results in Figure 5.10b and Figure 5.11 show that our procedure is effective at building the topological map (AGVD) of the experimental arena.

Chapter 6

AUTOMATED CONSTRUCTION OF METRIC MAPS USING A STOCHASTIC ROBOTIC SWARM LEVERAGING RECEIVED SIGNAL STRENGTH

Source: Ragesh K. Ramachandran and Berman [131]

Funding: Arizona State University Global Security Initiative

ABSTRACT

In this chapter, a novel automated procedure for constructing a metric map of an unknown domain with obstacles using uncertain position data collected by a swarm of resource-constrained robots is presented. The robots obtain this data during random exploration of the domain by combining onboard odometry information with noisy measurements of signals received from transmitters located outside the domain. This data is processed offline to compute a density function of the free space over a discretization of the domain. Persistent homology techniques from topological data analysis are used to estimate a value for thresholding the density function, thereby segmenting the obstacle-occupied region in the unknown domain. The approach is substantiated with theoretical results to prove its completeness and to analyze its time complexity. The effectiveness of the procedure is illustrated with numerical simulations conducted on six different domains, each with two signal transmitters.

The work in this chapter was first presented in [131]. In this work [131], we develop an automated procedure for constructing a metric map of an unknown, GPS-denied environment with obstacles using uncertain localization data acquired by a swarm of robots with local sensing and no inter-robot communication. The procedure is scalable with the number of robots. Each robot generates the localization data by combining its onboard odometry information with the measured strength of signals that are emitted by transmitters located outside the domain. For example, in a disaster response scenario, the robots may be able to detect radio signals only from the area outside the domain from which they were deployed. Our procedure is also applicable to indoor environments; even though signal propagation through such environments has high unpredictability [48], much research has been devoted to the use of received signal strength intensity (RSSI) for indoor localization of robots [114]. In [106], a technique is presented for multi-robot localization that could be used for mapping environments without global position information. Similar to our approach, this technique uses robot measurements of external signals; however, unlike our approach, it requires robots to distinguish neighboring robots from obstacles and communicate explicitly with them. We prove that our procedure will generate a metric map under specified assumptions on the coverage of the domain by the robots.

Previous chapters Chapter 4 and Chapter 5 presented procedures for estimating the number of obstacles in an unknown domain and extracting a topological map of the domain, respectively. The methodology presented in Chapter 5 generates a topological map in the form of a Voronoi diagram by applying clustering and wave propagation algorithms to a probabilistic map and does not incorporate RSSI measurements. Also Chapter 3 described an optimal control method for mapping GPS-denied environments using a swarm of robots with both advective and dif-

fusive motion. Although this method only requires measurements of encounter times with obstacles, it relies on an accurate partial differential equation model of the swarm dynamics, and it is ineffective on domains with multiple obstacles.

The procedure is fundamentally an occupancy grid mapping method, which represents the unknown domain using a set of evenly-spaced binary random variables that each indicate the presence or absence of an obstacle at that location in the domain. Occupancy grid mapping methods has been studied extensively in robotics literature both in single robot [141, 87] and multi-robot settings[116, 14]. However, our occupancy grid mapping strategy distinguishes itself from other approaches in light of the fact that we prove the probabilistic completeness of our strategy. In words, using Theorem 6 we guarantee that with our approach would result in the map of the explored unknown environment with probability one as long as the assumptions associated with Theorem 6 are satisfied. The first step in our procedure, namely, data collection by a swarm of robots during exploration of the domain, is a decentralized process. In the subsequent step, the collected data is processed offline to compute a probability of occupancy on the grid cells. The computations from this step onward are executed by a central computer that generates the domain map from the computed density function. This is the only centralized component of our mapping procedure, and it is scalable with the number of robots since the map computation can be parallelized. Tools from topological data analysis (TDA) are used to compute a threshold density value in order to identify the obstacle-filled region in the occupancy grid. This computation is performed by constructing a probability-based filtration on the free space in the domain. We direct the reader to Section 5.2 for the necessary background on the topological concepts that are used in this paper. An important contribution of our paper is Theorem 6, a proof of the completeness of our mapping procedure, which was absent in our

earlier work presented in Chapter 5. Our result in Lemma 1, which is needed to prove Theorem 6, cannot be proved for the system considered in Chapter 5 since it is unobservable. This provides insight into why our approach in Chapter 5 cannot be used for metric mapping.

The reason for computing the map offline is twofold. First, the robots localize in the domain with uncertainty that increases over time due to noise in their actuators, sensors, and RSSI measurements. Even though we prove in Lemma 1 that this uncertainty is bounded, the bound could be large for a particular robot depending on the random path that it follows, which would make its localization data unreliable. Hence, each individual robot can only generate an uncertain map of the region that it explores. However, our approach constructs an accurate estimate of the map of a region by fusing data offline from multiple robots that explore the region. Second, in order for each robot to construct the map of the domain online, it should individually cover the entire domain and have sufficient computational capabilities to perform all the map generation calculations onboard. In our strategy, this is infeasible due to the low computational resources of the robots that we consider. As an alternative, the robots could construct local maps, communicate these maps to other robots that they encounter during the course of exploration, and merge the maps that they receive from the other robots. However, this would require the robots to have communication capabilities, which we do not assume in our scenario.

6.1 Problem Statement

We consider the problem of estimating the metric map of a closed, bounded, path connected, GPS-denied domain $D \subset \mathbb{R}^d$ with obstacles using uncertain localization data acquired by a swarm of N robots while exploring the domain. We restrict

our analysis to domains with boundaries having regularity of at least Lipschitz continuity. Although here we only consider the case $d = 2$, it is straightforward to extend our procedure to the case where $d > 2$. We exclude scenarios where an obstacle is located very close to the domain boundary, since it is highly unlikely that the robots will enter the gap between the boundary and obstacle. We assume that such gaps are at least twice a robot's sensing diameter.

Each robot is equipped with a compass, wheel encoders, and a received signal strength indicator (RSSI) device such as Atheros [156], and it can detect obstacles and other robots within its local sensing radius and perform collision avoidance maneuvers. Two radio transmitters are located outside the domain, and the robots' RSSI devices can measure their signals anywhere inside the domain. As the proof of Theorem 6 shows, our strategy requires at least two transmitters to map a two-dimensional domain. We assume that the robots have sufficient memory to store the data that they collect during exploration. We also assume that after a sufficiently large time T, the robots have covered the domain according to the coverage definition given in Section 6.3.1. After time T, the robots move to a common location for extraction of the stored data.

The robots move with a constant speed v and a heading $\theta(t)$ at time t with respect to a fixed global frame. The position and velocity state vectors of a robot in this frame are defined as $\mathbf{X}(t) = [x(t),\ y(t)]^T$ and $\mathbf{V}(t) = [v_x(t),\ v_y(t)]^T = [v\ \cos(\theta(t)),\ v\ \sin(\theta(t))]^T$, respectively. At the initial time $t = 0$, the start of the exploration phase, a precise estimate of $\mathbf{X}(0)$ and $\theta(0)$ is provided to each robot. During the deployment, each robot generates a uniform random number $U \in [0, 1]$ at the start of every time step Δt. If $U \leq p_{th}$, where p_{th} is a specified value, then the robot randomly chooses a new heading $\theta(t) \in [-\pi, \pi]$. We define $\mathbf{W}_x(t) \in \mathbb{R}^2$ and $\mathbf{W}_v(t) \in \mathbb{R}^2$ as vectors of independent, zero-mean normal random variables that are

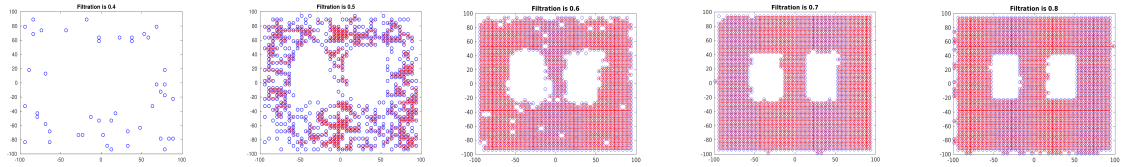
generated at time t. These vectors model randomness in the robots' motion due to wheel actuation noise. We define the vector $\mathbf{W}(t) \in \mathbb{R}^4$ as $\mathbf{W}(t) = [\mathbf{W}_x(t)\ \mathbf{W}_v(t)]$ and note that $\mathbf{W}(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}),\ \mathbf{Q} \in \mathbb{R}^{4\times4}$.

Using this notation, we model each robot as a point mass that follows the standard linear odometry motion model [142, 26], whose state space form can be written as:

$$\begin{bmatrix} \mathbf{X}(t+\Delta t) \\ \mathbf{V}(t+\Delta t) \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \Delta t\mathbf{I} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{X}(t) \\ \mathbf{V}(t) \end{bmatrix} + \begin{bmatrix} \mathbf{W}_x(t) \\ \mathbf{W}_v(t) \end{bmatrix}, \tag{6.1}$$

where $\mathbf{I}$ is the identity matrix. We denote the system matrix of Equation 6.1 by $\mathbf{A}$.

While performing this correlated random walk through the domain, each robot uses an extended Kalman filter [142] to estimate its global position and the associated covariance matrix from its onboard odometry and RSSI measurements of the signals emitted by the two transmitters. The robot records this estimated position and covariance matrix at fixed time intervals. Although exploration through random walking gives only weak guarantees on complete coverage of the domain, it is a simple motion strategy that can be implemented on robots with the limitations that we consider. It should be noted that any exploration strategy that accommodates these limitations can be substituted for random walking. We specify that the line joining the two transmitters lies outside the domain (see Theorem 6). The signal strength attenuation of a radio signal from a transmitter i is a function of distance from the transmitter location $\mathbf{X}_i$ [121]. We adopt the model $S_i(\mathbf{X}(t)) = K_i Pow_i \|\mathbf{X}(t) - \mathbf{X}_i\|^{-\alpha}$ presented in [96], where $\alpha \in [0.1, 2]$, Pow_i is the transmitted signal voltage of transmitter i, and K_i is the corresponding proportionality constant. We set $\alpha = 2$, as is commonly done in the literature [121]. We define $\mathbf{S}(\mathbf{X}(t)) = [S_1(\mathbf{X}(t)),\ ...,\ S_l(\mathbf{X}(t))]^T$, where l is the number of transmitters (here, we set $l = 2$). We also define $\mathbf{N}_S(t) \in \mathbb{R}^l$ and $\mathbf{N}_V(t) \in \mathbb{R}^2$ as vectors of independent,



(a) Simplicial complex, $\delta = 0.4$ (b) Simplicial complex, $\delta = 0.5$ (c) Simplicial complex, $\delta = 0.6$ (d) Simplicial complex, $\delta = 0.7$ (e) Simplicial complex, $\delta = 0.8$

Figure 6.1: Filtration used to generate the barcode diagram Figure 6.5c for the domain shown in Figure 6.2c. The red triangles are the 2-simplices that are constructed from the centers of the grid cells in the domain discretization.

zero-mean normal random variables that are generated at time t. These vectors model noise in the robots' RSSI devices and wheel encoders, respectively.

Let $\mathbf{Z}(t)$ denote the vector of sensor measurements received by a robot at time t. Then the output equation of the system can be written as,

$$\mathbf{Z}(t) = \begin{bmatrix} \mathbf{S}(\mathbf{X}(t)) \\ \mathbf{V}(t) \end{bmatrix} + \begin{bmatrix} \mathbf{N}_S(t) \\ \mathbf{N}_V(t) \end{bmatrix}. \tag{6.2}$$

From the proof of Theorem 6, we can see that if we use two transmitters, it is required that line joining the transmitters should lie outside the domain of interest.

6.2 Map Generation Procedure

This section describes a procedure for extracting a metric map of the domain as an occupancy grid map using the noisy localization data collected by the swarm of robots.

6.2.1 Computation of the Density Function of Free Space on a Discretization of the Domain

As in other occupancy grid mapping algorithms [142], our first step is to discretize the domain into a fine grid of M cells. The objective of this step is to use the robots' recorded data on their estimated positions to compute a density function $p^f : m_i \to [0, 1]$ that encodes the probability of a cell m_i, $i \in 1, ..., M$ being unoccupied by an obstacle, or *free*. We use the notation p_i^f instead of $p^f(m_i)$ for brevity. Here we summarize our approach to computing p_i^f. Although it is similar to the approaches in earlier chapters Chapter 4 and Chapter 5, the probabilistic occupancy grid map computation in this paper uses a different equation for s_i (Equation 6.3), the score assigned to each grid cell i, than the computation in our previous works.

While each robot $j \in \{1, ..., N\}$ moves randomly through the unknown environment, it records data at times $t_k \in [0, T]$, $k \in 1, ..., K$. This data consists of the tuple $d_k^j = \{\mu_k^j, \sigma_k^j\}$, where $\mu_k^j \in \mathbb{R}^2$ and $\sigma_k^j \in \mathbb{R}^{2\times 2}$ are the mean and covariance matrix, respectively, of the robot's estimate of its position in Cartesian coordinates at time t_k. We define p_{ijk} as the discrete probability that the j^{th} robot occupied the cell m_i at time t_k. This probability is calculated for all robots, cells, and times t_k by integrating the Gaussian distribution with mean μ_k^j and covariance matrix σ_k^j over the part of the domain occupied by cell m_i. We then filter out probabilities p_{ijk} that are obtained from Gaussian distributions which are centered far from each grid cell m_i. Toward this end, we define the set $P_i = \{p_{ijk} \mid p_{ijk} > \rho\}$, where $\rho > 0$ is a tolerance. In this paper, we set $\rho = 0.05$ based on the heuristic that, $p_{ijk} < 0.05$ is obtained from a Gaussian distribution which is centered far from the grid cell m_i. We compute p_i^f for each cell m_i using a technique similar to the log odds computation that is commonly employed in the robot SLAM literature [142]. A score $s_i \in [0, \infty)$ is assigned

to each grid cell m_i according to the equation

$$s_i = \frac{1}{|P_i|} \sum_{p_{ijk} \in P_i} \log\left(\frac{1}{1 - p_{ijk}}\right). \tag{6.3}$$

We then compute the probability that cell m_i is free using the formula $p_i^f = 1 - (\exp(s_i))^{-1}$. p_i^f could be interpreted as the probability that a grid cell m_i of being free or unoccupied by an obstacle.

Next, we apply a moving average linear filter, a common technique in image processing, to the probabilities p_i^f. This ensures that the automated thresholding step, described in the next section, is effective even if the robots fail to cover a few free grid cells in the domain. For each grid cell m_i, we replace p_i^f with the mean of p_i^f and the p_j^f of its neighboring grid cells m_j. This eliminates any p_i^f value that is unrepresentative of its neighborhood. The simulations in Section 6.4 use a 3×3 square neighborhood for filtering. Strictly speaking, this step can be skipped if the assumption on the coverage of the domain by the swarm is satisfied.

6.2.2 Thresholding the Density Function to Generate the Map

In this step, we threshold each p_i^f to classify the corresponding grid cell m_i as a free or obstacle-occupied cell. The existence of a threshold for this classification is proven in Theorem 6. We apply persistent homology [39], a topological data analysis (TDA) technique based on algebraic topology [60], to automatically find a threshold based on the p_i^f of each grid cell. An implicit assumption required for this technique is that each obstacle contains at least one grid cell with $p_i^f = 0$. This TDA-based technique provides an adaptive method for thresholding an occupancy grid map of a domain that contains obstacles at various length scales. In fact, it can be used with other occupancy grid mapping methods to implement automated thresholding. We describe The technique is described in full in Section 5.4.1 and

we summarize it here for completeness.

First we select the center points of the grid cells with $p_i^f > \gamma = 1 - \delta$. These points become the 0-simplices subset $\mathcal{S}_0$ of the simplicial complex κ_δ that we are constructing. Let $e_0 \in \mathcal{S}_0$. Then a 1-simplex is constructed by pairing e_0 with its adjacent vertical, horizontal, and diagonal neighbors (8-connectivity) [57] if the neighbors are elements of $\mathcal{S}_0$. A set of 1-simplices $\mathcal{S}_1 \subset \kappa_\delta$ is generated using the elements in $\mathcal{S}_0$. Similarly, a set of 2-simplices $\mathcal{S}_2 \subset \kappa_\delta$ is constructed from the elements in $\mathcal{S}_1$, choosing every subset of three elements in $\mathcal{S}_1$ that form a triangle. Figure 6.1 illustrates a filtration constructed for the domain in Figure 6.2c.

Once a filtration is constructed, Next, a *barcode diagram* is used to identify the number of topological features in the domain, which is given by the number of barcode arrows in each homology group. The δ_{cls} corresponding to the threshold γ_{cls} for classification of the grid cells is defined as the minimum value of δ for which all the topological features are captured by the corresponding simplicial complex. In other words, δ_{cls} is the value of δ for which all the barcode segments except the arrow are annihilated for all values of δ greater than this value. Therefore, the threshold γ_{cls} for classification of the grid cells is $1 - \delta_{cls}$. This computation is done in practice by taking the maximum value of δ that is spanned by the terminating barcode segments in all the homology groups. In this chapter, the persistent homology computation was done using the MATLAB-based JavaPlex package [3]. Since the results presented in this chapter are only for two-dimensional domains, we restricted the persistent homology computations to dimensions zero and one.

6.3 Analysis of the Mapping Procedure

6.3.1 Probabilistic completeness of the Procedure

In this section, we analyze the *completeness of the approach* in a probabilistic sense, meaning that the procedure described in Section 6.2 will result in a probabilistic occupancy map of the unknown domain that distinguishes between occupied and free grids cells, provided that the inputs to the procedure satisfy certain assumptions with probability one. The approach may fail to produce the desired output if the assumptions do not hold. The simulation results in Section 6.4 demonstrate the effectiveness of our procedure even when the required assumptions are not fully satisfied.

The most important assumption required for the completeness of our approach is that the domain is *completely covered* by the swarm of robots. By this, we mean that the recorded localization data includes at least one data tuple per free grid cell whose μ lies inside the grid cell. We assume that even if some of the robots fail to return after exploring the domain, sufficient data is obtained from the recovered robots to achieve complete coverage of the domain.

We begin our analysis by proving the existence of a threshold on the density function, which serves as a decision variable to distinguish between free and occupied grid cells. Toward this end, we first state the following lemma, which gives a result that is required to prove Theorem 6. The result in Lemma 1 follows trivially when both the robot dynamics and measurement models are linear. However, proving this result requires a careful analysis when either the dynamics or measurement models are nonlinear, as in our case.

Lemma 1. *The error in the robots' position estimates is bounded with probability one,*

with a common bound for all robots, if each robot follows the motion model Equation 6.1 and estimates its state vector using an extended Kalman filter based on the outputs in Equation 6.2.

Proof. Let $\mathbf{S}(\mathbf{X})_{\mathbf{X}}$ denote the first-order derivative of $\mathbf{S}(\mathbf{X})$ with respect to $\mathbf{X}$ in Equation 6.2. Note that we have dropped the variable t for conciseness. We also define $\hat{\mathbf{X}}$ as the estimate of $\mathbf{X}$. Assuming that $\mathbf{S}(\mathbf{X})$ is analytic in a neighborhood of an estimate $\hat{\mathbf{X}}$, we can write the first-order Taylor-series expansion of $\mathbf{S}(\mathbf{X})$ about $\hat{\mathbf{X}}$ as,

$$\mathbf{S}(\mathbf{X}) = \mathbf{S}(\hat{\mathbf{X}}) + \mathbf{S}(\mathbf{X})_{\hat{\mathbf{X}}}(\mathbf{X} - \hat{\mathbf{X}}) + o(||\mathbf{X} - \hat{\mathbf{X}}||^2). \tag{6.4}$$

If $\mathbf{h} = (\mathbf{X} - \hat{\mathbf{X}})$, then the higher-order terms in Equation 6.4 are represented by $o(||\mathbf{h}||^2)$, whose norm tends to zero faster than $||\mathbf{h}||^2$ does as $||\mathbf{h}|| \to 0$, by the definition of $o(||\mathbf{h}||^2)$. In other words, $\lim_{||\mathbf{h}|| \to 0} \frac{\left||o(||\mathbf{h}||^2)|\right|}{||\mathbf{h}||^2} \to 0$. This implies that there exists an open ball of radius ϵ around $\mathbf{h}$ such that if $||\mathbf{h}||^2 < \epsilon$, where $\epsilon > 0$, then $\left\|o(||\mathbf{h}||^2)\right\| < ||\mathbf{h}||^2 < \epsilon$. Thus, the inequality

$$\left\|\mathbf{S}(\mathbf{X}) - \mathbf{S}(\hat{\mathbf{X}}) - \mathbf{S}(\mathbf{X})_{\hat{\mathbf{X}}}(\mathbf{X} - \hat{\mathbf{X}})\right\| < \left\|\mathbf{X} - \hat{\mathbf{X}}\right\|^2 \tag{6.5}$$

is satisfied in some neighborhood of $\hat{\mathbf{X}}$ if $\mathbf{S}(\hat{\mathbf{X}})$ is analytic in that neighborhood.

From Theorem 3.1 in [108], we know that the estimation error $\zeta_k = \left\|\mathbf{X}_k - \hat{\mathbf{X}}_k\right\|$ of an extended Kalman filter at the k^{th} time step, where $k \in \{1, 2, ..., K\}$, is bounded with probability one as long as the following conditions hold:

1. $\zeta_0 \leq \epsilon$ for some $\epsilon \geq 0$.

2. Define $\mathbf{X}_k^s = [\mathbf{X}_k; \mathbf{V}_k]$ as the state vector in Equation 6.1, $f(\mathbf{X}_k^s)$ as the state map, and $\mathbf{A}_k = \frac{\partial f}{\partial \mathbf{X}_k^s}(\hat{\mathbf{X}}_k^s)$. The matrix $\mathbf{A}_k$ is nonsingular for all $k \geq 0$.

3. Let $h(\mathbf{X}_k^s)$ be the output map given in Equation 6.2 and $\mathbf{H}_k = \frac{\partial h}{\partial \mathbf{X}_k^s}(\hat{\mathbf{X}}_k^s)$. Define the functions ϕ and χ as:

$$\phi(\mathbf{X}_k^s, \hat{\mathbf{X}}_k^s) = f(\mathbf{X}_k^s) - f(\hat{\mathbf{X}}_k^s) - \mathbf{A}_k(\hat{\mathbf{X}}_k^s)\left(\mathbf{X}_k^s - \hat{\mathbf{X}}_k^s\right), \tag{6.6}$$

$$\chi(\mathbf{X}_k^s, \hat{\mathbf{X}}_k^s) = h(\mathbf{X}_k^s) - h(\hat{\mathbf{X}}_k^s) - \mathbf{H}_k(\hat{\mathbf{X}}_k^s)\left(\mathbf{X}_k^s - \hat{\mathbf{X}}_k^s\right). \tag{6.7}$$

There exist real numbers $\epsilon_\phi, \epsilon_\chi, K_\phi, K_\chi > 0$ such that

$$\left\|\phi(\mathbf{X}_k^s, \hat{\mathbf{X}}_k^s)\right\| \leq K_\phi \left\|\mathbf{X}_k^s - \hat{\mathbf{X}}_k^s\right\|^2 \leq K_\phi \epsilon_\phi^2, \tag{6.8}$$

$$\left\|\chi(\mathbf{X}_k^s, \hat{\mathbf{X}}_k^s)\right\| \leq K_\chi \left\|\mathbf{X}_k^s - \hat{\mathbf{X}}_k^s\right\|^2 \leq K_\chi \epsilon_\chi^2. \tag{6.9}$$

4. There are positive real numbers $\bar{a}, \bar{h}, \bar{p}, \underline{p} > 0$ such that

$$\|\mathbf{A}_k\| \leq \bar{a}, \tag{6.10}$$

$$\|\mathbf{H}_k\| \leq \bar{h}, \tag{6.11}$$

$$\underline{p}I \leq \mathbf{P}_k \leq \bar{p}\mathbf{I}, \tag{6.12}$$

where $\mathbf{P}_k$ is the covariance matrix at the k^{th} time step.

To prove the lemma, we will now show that these four conditions are satisfied. Conditions (1) and (2) are satisfied because $\zeta_0 = 0$ and $\mathbf{A}_k$ is the constant matrix $\mathbf{A}$, which is nonsingular. Equation 6.8 is satisfied trivially, since $\phi(\mathbf{X}_k^s, \hat{\mathbf{X}}_k^s)$ is zero when $\mathbf{A}_k$ is a constant matrix. In condition (3), we need to determine whether the bounds described in Equation 6.9 are fulfilled with $h(\mathbf{X}_k^s) = [\mathbf{S}(\mathbf{X}(t)); \mathbf{V}(t)]$ in Equation 6.7. To verify this, it is enough to show that a condition analogous to Equation 6.7 is satisfied when the output map is restricted to the signal map. In other words, we need to check whether Equation 6.7 is satisfied when $h(\mathbf{X}_k^s) = \mathbf{S}(\mathbf{X})$. Equation 6.5 shows that this condition is true locally at every point as long as $\mathbf{S}(\mathbf{X})$ is analytic, which is true for all points inside the domain in our case, since the transmitters are located outside the domain.

Examining condition (4), we find that computation of $\mathbf{H}_k$ is required for further analysis. Given the definition $h(\mathbf{X}_k^s) = [\mathbf{S}(\mathbf{X}(t)); \mathbf{V}(t)]$ from Equation 6.2, we can compute the Jacobian of $h(\mathbf{X}_k^s)$ as,

$$\mathbf{H}_k(\mathbf{X}_k^s) = \begin{bmatrix} \mathbf{S}_k(\mathbf{X_k})_\mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{6.13}$$

where

$$\mathbf{S}_k(\mathbf{X_k})_\mathbf{X} = \begin{bmatrix} \frac{-\alpha K_1 Pow_1(x_k - x_{t1})}{\left((x_k - x_{t1})^2 + (y_k - y_{t1})^2\right)^{\frac{2+\alpha}{2}}} & \frac{-\alpha K_1 Pow_1(y_k - y_{t1})}{\left((x_k - x_{t1})^2 + (y_k - y_{t1})^2\right)^{\frac{2+\alpha}{2}}} \\ \frac{-\alpha K_2 Pow_2(x_k - x_{t2})}{\left((x_k - x_{t1})^2 + (y_k - y_{t1})^2\right)^{\frac{2+\alpha}{2}}} & \frac{-\alpha K_2 Pow_2(y_k - y_{t2})}{\left((x_k - x_{t1})^2 + (y_k - y_{t1})^2\right)^{\frac{2+\alpha}{2}}} \end{bmatrix}. \tag{6.14}$$

Here, (x_{ti}, y_{ti}) are the Cartesian position coordinates of the i^{th} transmitter.

Equation 6.10 and Equation 6.11 are trivially satisfied. Now it is left to prove that the constraint described using Equation 6.12 is also in agreement. This inequality is related to the observability of the system. Using Theorem 4.1 in [108], we deduce that Equation 6.12 is satisfied if the linearized system is observable for every n; i.e., if the observability matrix of the linearized system $\mathbf{O}_k = [\mathbf{H}_k;\ \mathbf{H}_k\mathbf{A}_k;\ \mathbf{H}_k\mathbf{A}_k^2;\ \mathbf{H}_k\mathbf{A}_k^3]$ has full rank for all k.

$\mathbf{O}_k$ can be computed to be

$$\mathbf{O}_k = \begin{bmatrix} \mathbf{S}_k(\mathbf{X_k})_\mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \\ \mathbf{S}_k(\mathbf{X_k})_\mathbf{X} & \Delta t \mathbf{S}_k(\mathbf{X_k})_\mathbf{X} \\ \mathbf{0} & \mathbf{I} \\ \mathbf{S}_k(\mathbf{X_k})_\mathbf{X} & 2\Delta t \mathbf{S}_k(\mathbf{X_k})_\mathbf{X} \\ \mathbf{0} & \mathbf{I} \\ \mathbf{S}_k(\mathbf{X_k})_\mathbf{X} & 3\Delta t \mathbf{S}_k(\mathbf{X_k})_\mathbf{X} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{6.15}$$

where $\mathbf{0}, \mathbf{I} \in \mathbb{R}^{2\times 2}$.

After row transformations of $\mathbf{O}_k$ using Gaussian elimination, we obtain

$$\mathbf{O}_k = \begin{bmatrix} \mathbf{S}_k(\mathbf{X_k})_{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \Delta t \mathbf{S}_k(\mathbf{X_k})_{\mathbf{X}} \\ & \mathbf{0} \end{bmatrix}, \tag{6.16}$$

where the large $\mathbf{0}$ is a matrix of zeros. Since $\Delta t \neq 0$, it is evident from Equation 6.16 that $\mathbf{O}_k$ is not full rank if and only if $\mathbf{S}_k(\mathbf{X_k})_{\mathbf{X}}$ is not full rank. It can be shown that the points (x_k, y_k) at which $\mathbf{S}_k(\mathbf{X_k})_{\mathbf{X}}$ is not full rank obey the following equation:

$$\frac{(x_k - x_{t1})}{(y_k - y_{t1})} = \frac{(x_k - x_{t2})}{(y_k - y_{t2})} = constant. \tag{6.17}$$

These points comprise the line joining the two transmitters. Therefore, if we ensure that this line does not pass through the domain, then the system is observable. Alternately, we could make the system observable by introducing a third transmitter which is non-collinear to the other two transmitters. Under these constraints, condition (4) is satisfied, implying that the state estimation error for each robot is bounded. Therefore, the estimation error of the robot positions is also bounded, since it is a part of the state. The maximum of all the robots' position estimation errors serves as a uniform bound for these errors. □

Finally, we prove the existence of a threshold value of p_i^f that distinguishes the free grid cells from the occupied cells.

Theorem 6. *Under the assumption of complete coverage of the domain, a grid cell m_i is free if and only if there exists a threshold $\gamma \in [0, 1]$ for which $p_i^f > \gamma$ with probability one.*

Proof. We begin by proving the sufficient part of the statement; i.e., that there exists a threshold $\gamma \in [0, 1]$ such that a grid cell m_i is free if $p_i^f > \gamma$. Lemma 1 shows that there exists an estimation error bound on the position estimate for all robots uniformly. Thus, the uncertainty associated with position is also bounded. Also, for every free grid cell m_i, there is at least one data tuple d_k^j that is centered inside the grid due to the assumption on coverage. The boundedness of the uncertainty ensures the existence of a two-dimensional symmetric Gaussian distribution function with an associated covariance matrix having a finite norm σ_{max}. The integral of this function over the grid cell is less than or equal to the integral of the Gaussian function associated with d_k^j. Without loss of generality, we assume that the grids cells are square with area $[-s, s]\times[-s, s]$ for simplicity. After some algebraic manipulation, we can derive that

$$\frac{1}{|P_i|}\sum_{p_{ijk}\in P_i}\log\left(\frac{1}{1-p_{ijk}}\right) > \frac{1}{|P_i|}\log\left(\frac{1}{1-(p)_i}\right) \tag{6.18}$$

where,

$$(p)_i = \frac{1}{2\pi\sigma_{max}^2}\int_{-s}^{s}\exp(-\frac{1}{2}(\frac{x^2}{\sigma_{max}^2})dx\int_{-s}^{s}\exp(-\frac{1}{2}(\frac{y^2}{\sigma_{max}^2})dy. \tag{6.19}$$

Let $t = \frac{x}{\sqrt{2}\sigma_{max}}$, then $\sqrt{2}\sigma_{max}dt = dy$. Substituting x and y in the above equation with t would result in the following equation:

$$(p)_i = \frac{1}{\sqrt{\pi}}\int_{-\frac{s}{\sqrt{2}\sigma_{max}}}^{\frac{s}{\sqrt{2}\sigma_{max}}}\exp(-t^2)dt\frac{1}{\sqrt{\pi}}\int_{-\frac{s}{\sqrt{2}\sigma_{max}}}^{\frac{s}{\sqrt{2}\sigma_{max}}}\exp(-t^2)dt \tag{6.20}$$

Which can be rewritten as:

$$(p)_i = \left(\frac{1}{\sqrt{\pi}}\int_{-\frac{s}{\sqrt{2}\sigma_{max}}}^{\frac{s}{\sqrt{2}\sigma_{max}}}\exp(-t^2)dt\right)^2 \tag{6.21}$$

From [19], $\frac{1}{\sqrt{\pi}} \int_{-s}^{s} \exp(-t^2)$ is the error function [1] denoted as $erf(s)$. $erf(s)$ can also be defined as $1 - erfc(s)$, where $erfc(s)$ is the complementary error function expressed as $erfc(s) = \frac{2}{\sqrt{\pi}} \int_{s}^{\infty} \exp(-t^2)dt$.

Therefore, Equation 6.21 can be expressing using $erf(s)$ as :

$$(p)_i = \left(erf(\frac{s}{\sqrt{2}\sigma_{max}}) \right)^2 \tag{6.22}$$

Using Corollary 1 in [20] we can estimate $erfc(s)$ as:

$$erfc(s) \leq exp(-s^2) \tag{6.23}$$

$$1 - erfc(s) \geq 1 - exp(-s^2) \tag{6.24}$$

or

$$erf(s) \geq 1 - exp(-s^2) \tag{6.25}$$

Using the above result $(p)_i$ can be estimates as:

$$(p)_i > \left(1 - \exp\left(-\frac{s^2}{2\sigma_{max}^2} \right) \right)^2 . \tag{6.26}$$

We now combine Equation 6.26 and Equation 6.18 and use the formula $p_i^f = 1 - (\exp(s_i))^{-1}$ to compute p_i^f, as mentioned in Section 6.2.1. After some algebraic simplification, we obtain the following inequality:

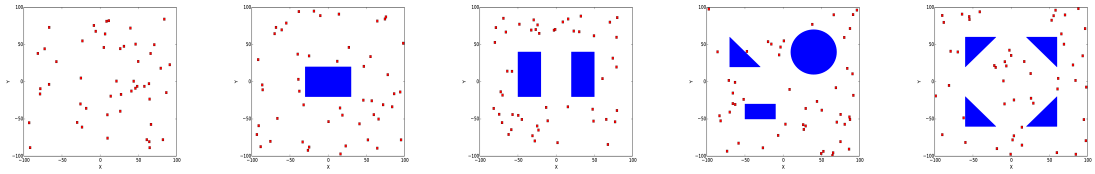
$$p_i^f > 1 - \left(1 - \left(1 - \exp\left(-\frac{s^2}{2\sigma_{max}^2} \right) \right)^2 \right)^{\left(\frac{1}{|P_i|} \right)} . \tag{6.27}$$

Note that the threshold, given by the right side of the above inequality, is bounded between zero and one, and that it increases as σ_{max} decreases and vice versa.

Now we prove the condition of necessity of the statement; i.e., that there exists a threshold γ such that $p_i^f > \gamma$ implies that the grid cell m_i is free, for all grid cells. We use proof by contradiction to establish this. First, suppose that this proposition is false. Then for any chosen γ, $p_i^f > \gamma$ does not imply that the grid cell m_i is free, for all grid cells. In other words, for every γ chosen, there exists at least one occupied grid cell for which $p_i^f > \gamma$. Let us choose γ to be the right-hand side of Inequality 6.27. Now assume that an occupied grid cell $Obsm_i$ satisfies the condition $p^f(Obsm_i) > \gamma$. This can happen in two possible cases. First, there may exist at least one data point d_k^j which is centered inside $Obsm_i$. This cannot occur, since we assume that the robot cannot move over obstacles. Second, σ_{max} may be unbounded, which is also not true according to Lemma 1. Therefore, we have confirmed that there exists a threshold that filters occupied grid cells. In other words, we were wrong to assume that the proposition was false. Thus, the proposition is true. □

Remark: Lemma 1 cannot be proven in the strategy presented in our previous paper [134], since the system there is unobservable (for this case, $\mathbf{S}_k(\mathbf{X_k})_\mathbf{X} = 0$ in Equation 6.16). Thus, the strategy presented in that paper does not guarantee metric map generation of the domain. Instead, it generates only a conservative topological map [25]. In addition, note that the lower bound on p_i^f from Inequality 6.27 increases as $|P_i|$ increases, indicating that as more robots visit a grid cell i, its probability of being free increases.

A TDA-based technique is used to estimate the threshold γ, because the threshold computed using Inequality 6.27 works only with complete coverage. If γ_{true} and γ_{est} are the true threshold and estimated threshold computed using the method described in Section 6.2.2, respectively, then $\gamma_{est} \geq \gamma_{true}$, since the metric and topological information coincide once the filtration parameter exceeds the value $1 - \gamma_{true}$.

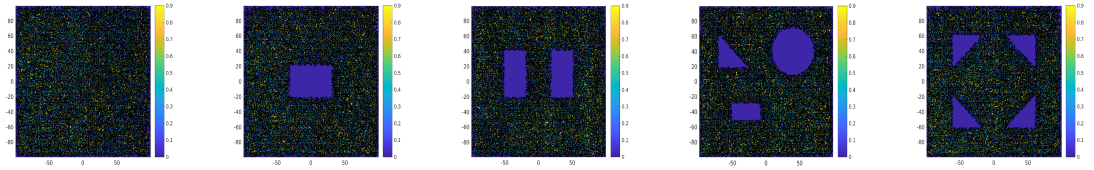


(a) No obstacles (b) One obstacle (c) Two obstacles (d) Three obstacles (e) Four obstacles

Figure 6.2: Snapshots of a simulated swarm of robots (red squares) moving through different domains with obstacles (blue shapes).

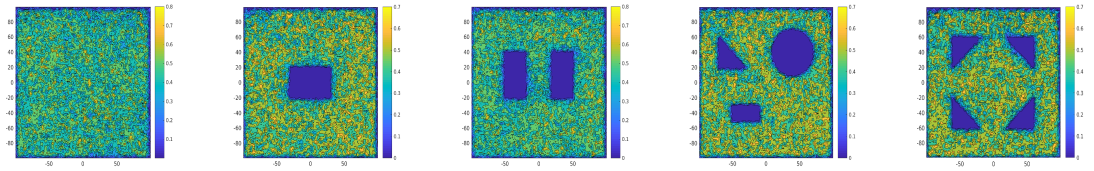
6.3.2 Computational Complexity Analysis

We analyze the computational complexity of the procedure using a similar approach to that in Section 5.5. Based on our analysis, the worst-case complexity of our procedure is $O(M^{2.372})$. The first step of our procedure is the computation of the density function. This computation varies linearly with the amount of data and the number of grid cells. That is, if N robots each collect K elements of data while exploring a discretized domain containing M grid cells, then the cost of computing the density function is of the order $O(NKM)$. This step can be parallelized by processing data from each robot in parallel, resulting in a reduced computational cost of the order $O(KM)$. The thresholding step is the most computationally expensive part of the procedure. This is because it requires the generation of a simplicial complex, whose size is linear in M, and a persistent homology computation that has a worst-case complexity of $O(M^{2.372})$, although for most practical scenarios it approaches $O(M)$ [40].



(a) No obstacles (b) One obstacle (c) Two obstacles (d) Three obstacles (e) Four obstacles

Figure 6.3: Contour plots of p_i^f, the probability that grid cell m_i is free, over all grid cells of the discretized domains generated after the step described in Section 6.2.1. Colorbar values range from 0 to 0.9.

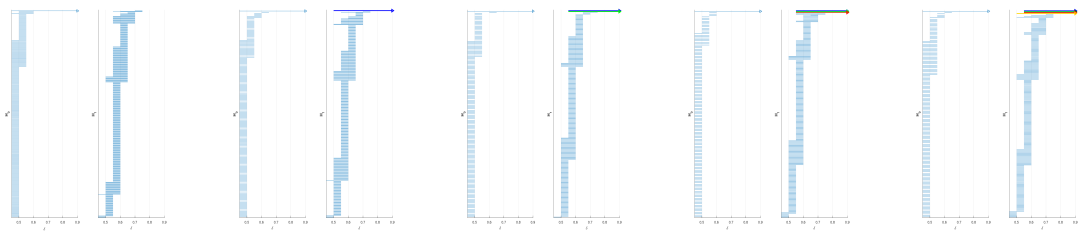


(a) No obstacles (b) One obstacle (c) Two obstacles (d) Three obstacles (e) Four obstacles

Figure 6.4: Contour plots of the filtered p_i^f shown in Figure 6.3, as described in Section 6.2.1.

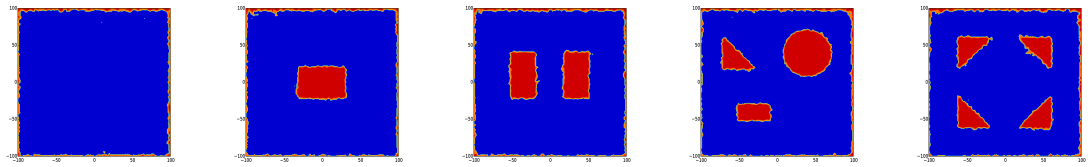
6.4 Simulation Results

In this section, we validate the mapping procedure in Section 6.2 by constructing metric maps of six simulated domains, each with two signal transmitters. Swarms of point robots in each domain were simulated in Python, and all other computations were performed in MATLAB. The robots have a sensing radius of 0.06 m and an average speed of $v = 0.2$ m/s. The simulations were initialized by placing the robots at random points near one of the domain boundaries. The robots follow the motion model Equation 6.1, in which $\mathbf{W}(t)$ is a diagonal matrix with 0.1 on the



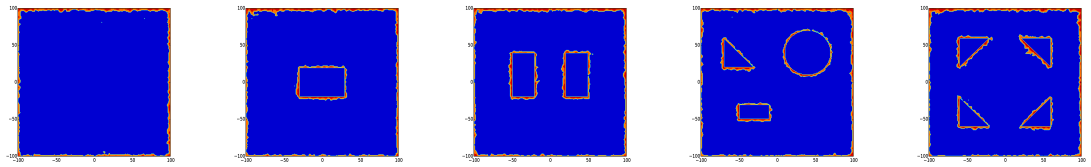
(a) No obstacles (b) One obstacle (c) Two obstacles (d) Three obstacles (e) Four obstacles

Figure 6.5: Barcode diagram for each domain, generated from the filtration described in Section 6.2.2, with δ_{cls} computed for each case as $\delta_{cls} = 0.75$.



(a) No obstacles (b) One obstacle (c) Two obstacles (d) Three obstacles (e) Four obstacles

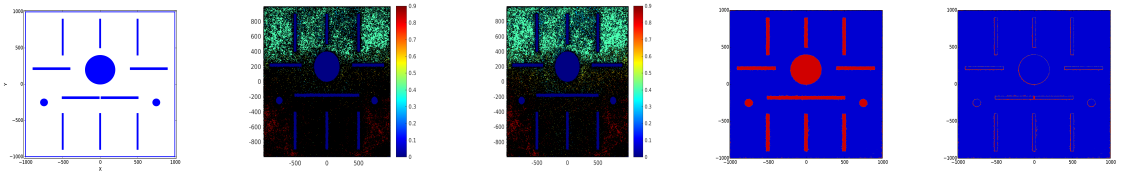
Figure 6.6: Contour plots of the thresholded map based on the thresholds computed using the TDA technique described in Section 6.2.2.



(a) No obstacles (b) One obstacle (c) Two obstacles (d) Three obstacles (e) Four obstacles

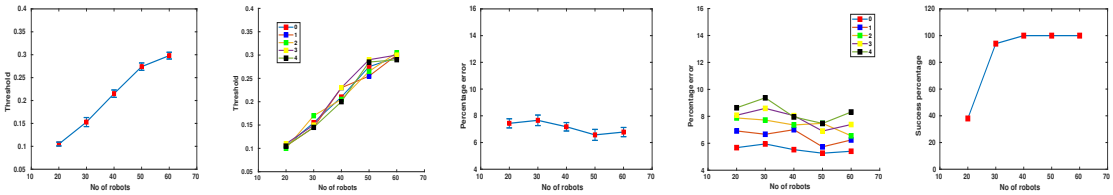
Figure 6.7: Contour plots of the absolute error between Figure 6.2 and Figure 6.6.

diagonal. The robots employ a simple collision avoidance policy, for which $p_{th} = 0.2$, by choosing a new random direction upon encountering the domain boundary,



(a) Complex domain (b) Computed p_i^f (c) Filtered p_i^f (d) Map (e) Absolute error

Figure 6.8: The outputs of the mapping procedure for a complex domain.

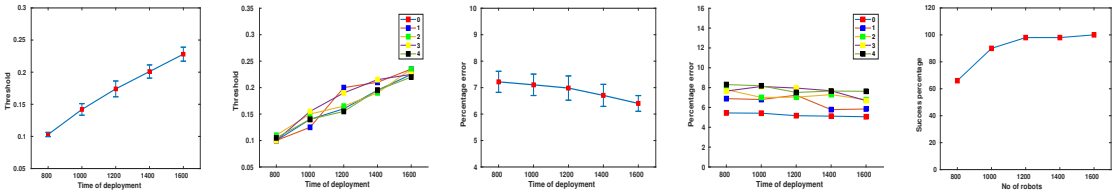


(a) Threshold from all domains combined (b) Mean threshold for each domain (c) Map estimation error from all domains combined (d) Mean map estimation error for each domain (e) % of trials that correctly characterize the domain

Figure 6.9: Plots showing the effect of the number of robots N on the threshold and map estimation error with $T = 300s$.

an obstacle, or another robot.

For the 2 m $\times$ 2 m domains shown in Figure 6.2, swarms of $N = 50$ robots were simulated over a deployment time of $T = 300$ s. The outputs at various stages of the mapping procedure for these domains are displayed in Figure 6.3-Figure 6.7: the contour plots of the computed p_i^f (Figure 6.3), the filtered p_i^f (Figure 6.4), the barcode diagrams (Figure 6.5), the thresholded maps (Figure 6.6), and the absolute error in the maps (Figure 6.7). These results show that the procedure generates an accurate metric map of each domain. To further evaluate the performance



(a) Threshold from all domains combined

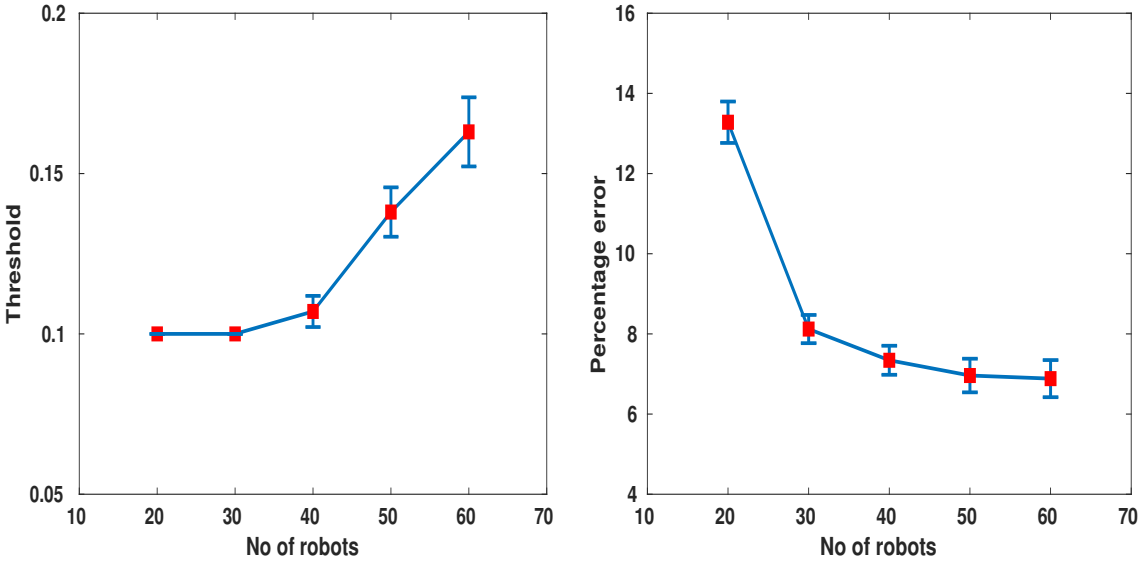
(b) Mean threshold for each domain

(c) Map estimation error from all domains combined

(d) Mean map estimation error for each domain

(e) % of trials that correctly characterize the domain

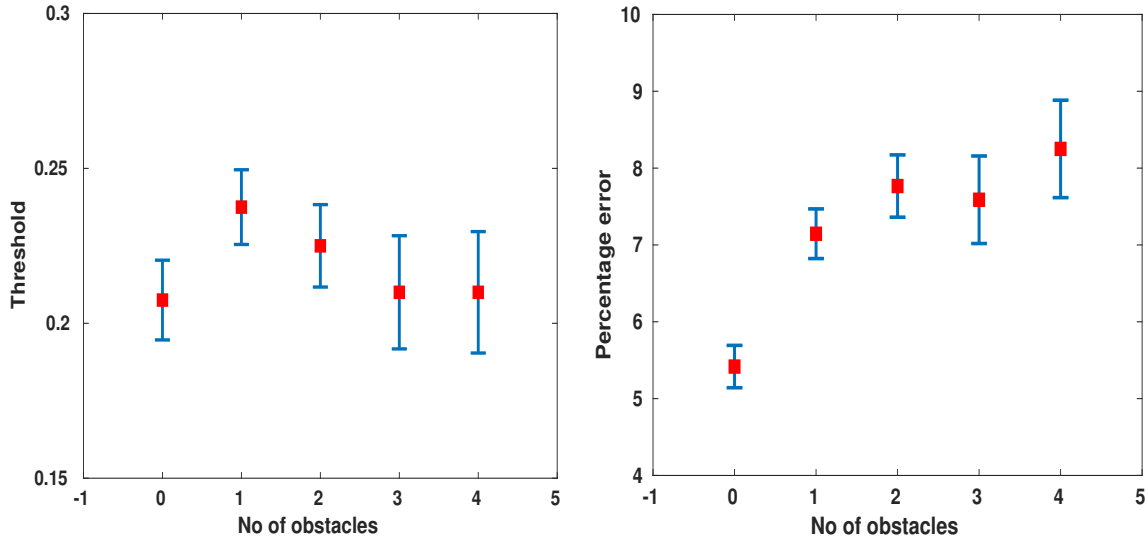
Figure 6.10: Plots showing the effect of the deployment time T on the threshold and map estimation error.



(a) Threshold for various N

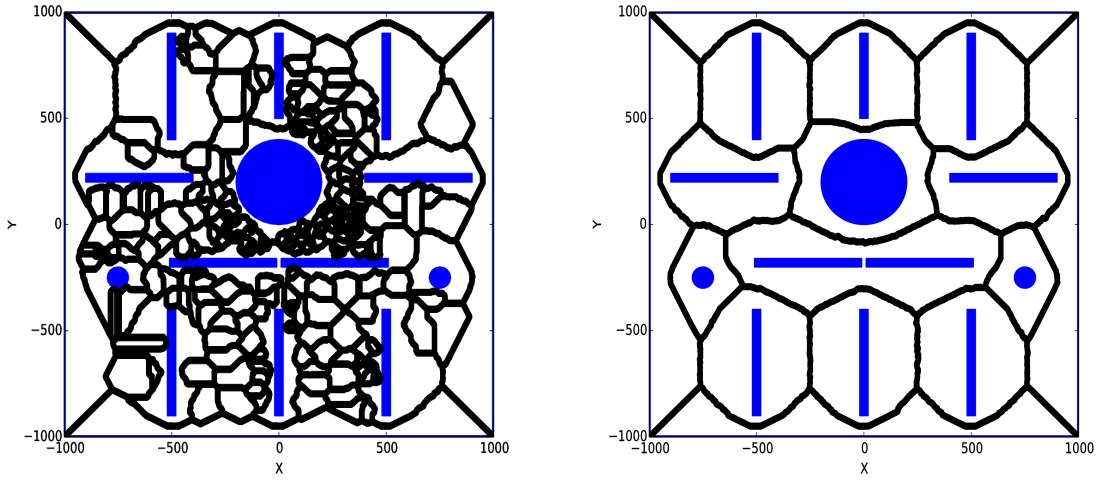
(b) Estimation error for various N

Figure 6.11: Plots showing the effect of the number of robots N on the threshold and map estimation error with $T = 160s$.



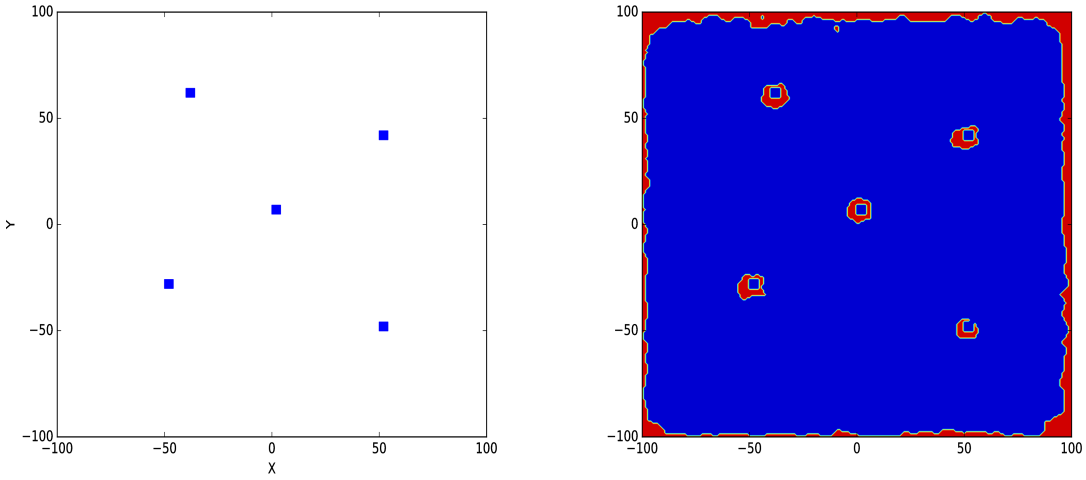
(a) Threshold (b) Map estimation error

Figure 6.12: Results from 20 simulations on each domain in Figure 6.2.



(a) Topological map from Chapter 5 (b) Topological map using current approach

Figure 6.13: Topological maps generated for a complex domain.



(a) Actual domain

(b) Absolute error

Figure 6.14: Simulation on a domain with five small square obstacles.

of our procedure, we ran 20 simulations on each domain in Figure 6.2 with the same parameters. Figure 6.12 shows the average threshold value γ with its 95% confidence interval and the mean absolute error (MAE) of the map estimation error with the corresponding 95% confidence interval for each domain. We used MAE rather than root-mean-squared error, since each error contributes proportionally in MAE. The plots confirm the effectiveness of our approach, since the average MAE lies between 5% to 8% and the error bars are relatively small.

We also conducted simulations in a larger, more complex domain of size 20 m $\times$ 20 m, in which $N = 200$ robots were deployed for $T = 1200$ s. These results are presented in Figure 6.8 and show that the procedure still generates an accurate map. A topological map for this domain was also constructed using the technique described Chapter 5 and compared with the one presented in that paper. From Figure 6.13, we see that our current approach results in an improved topological map.

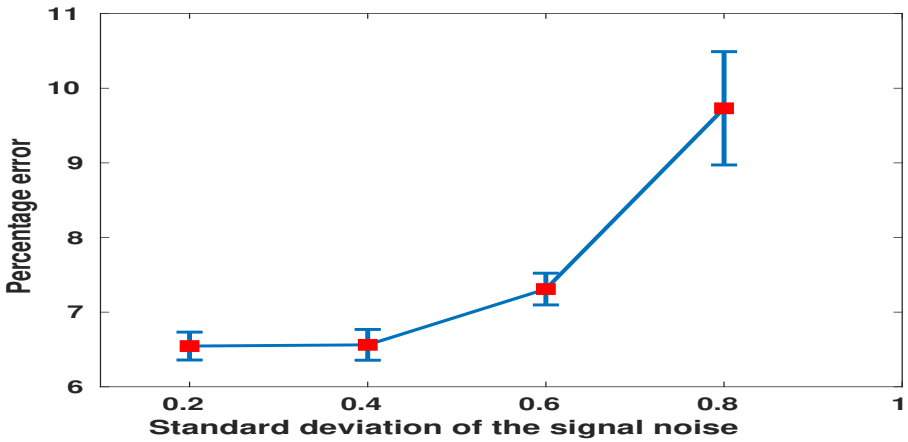


Figure 6.15: Plots showing effect of the noise of the signal on the map estimation error.

Finally, in order to examine the effectiveness of our strategy on domains with small obstacles, we performed a simulation on a domain of size 2 m $\times$ 2 m with five square obstacles, each of size 4 cm $\times$ 4 cm, using 50 robots with a deployment time of 300 s. The maximum standard deviation of the normal distribution associated with the robot's position was approximately 2.8 cm in both the x and y directions. The results of this simulation, plotted in Figure 6.14, show that our technique is able to generate a reasonably accurate map of the domain, even when the uncertainty in the robots' position is comparable to the size of obstacles in the domain.

We also investigated the effect of the number of robots N and the deployment time T on the performance of the procedure. For each of the 5 domains in Figure 6.2, we ran 10 simulations each with $N \in \{20, 30, 40, 50, 60\}$ and $T = 300$ s (Figure 6.9), and 10 simulations each with $T \in \{800, 1000, 1200, 1400, 1600\}$ s and $N = 40$ (Figure 6.10). The legends in subfigures (b) and (d) of Figures 6.9-6.10 show the number of obstacles in the domain corresponding to each plot. In addition, the error bars in Figures 6.9-6.12 represent the 95% confidence interval of the true

value. Figure 6.9a and Figure 6.10a show that the resulting mean threshold γ with its 95% confidence interval, computed from the 50 simulations over all domains for each parameter set, increases with increasing N and T, respectively. The corresponding plots of the mean γ for each domain, Figure 6.9b and 6.10b, exhibit the same trend. Figure 6.9c and Figure 6.10c plot the mean MAE of the map estimation error with its 95% confidence interval versus N and T, respectively, from the 50 simulations for each parameter set. The mean map error does not vary significantly with N, possibly because the deployment time $T = 300$ s is sufficient to thoroughly cover the domains, and it decreases with increasing T, as would be expected since more localization data is gathered during the deployment. To test this hypothesis, we reran the 50 simulations with 5 swarm sizes and $T = 160$ s (Figure 6.11). Figure 6.11b indeed shows that for this low T, the MAE of the map estimation error decreases as N increases. The mean map error for each domain versus N and T are shown in Figure 6.9d and Figure 6.10d, respectively. Finally, Figure 6.9e and Figure 6.10e show the dependence on N and T of the percentage of the 50 simulations for each parameter set in which the topological technique described in Section 6.2.2 successfully identifies the number of obstacles (topological features) in the five domains. As expected, the success rate increases with increasing N and T. In addition, we also investigated the effect on noise of the received signal on the map estimation error and the results are presented in Figure 6.15. The results in the figure were obtained by running 50 simulation trails on the 5 domain for each standard deviation of noise from $\{0.2, 0.4, 0.4, 0.8\}$. It is important to note that $\mathbf{N}_S(t)$ in Equation 6.2 was a diagonal matrix with 0.1 along the diagonal during the trails. The results in Figure 6.15 demonstrate our method's efficacy in estimating the map even when the signal models are inaccurate.

Chapter 7

INFORMATION CORRELATED LÉVY WALK EXPLORATION AND DISTRIBUTED MAPPING USING A SWARM OF ROBOTS

Source: Ragesh K. Ramachandran et al. [135]

Funding: Arizona State University Global Security Initiative.

ABSTRACT

In this chapter, we present a novel distributed method for constructing an occupancy grid map of an unknown environment using a swarm of robots with global localization capabilities and limited inter-robot communication was presented. The robots explore the domain by performing Lévy walks in which their headings are defined by maximizing the mutual information between the robot's estimate of its environment in the form of an occupancy grid map and the distance measurements that it is likely to obtain when it moves in that direction. Each robot is equipped with laser range sensors, and it builds its occupancy grid map by repeatedly combining its own distance measurements with map information that is broadcast by neighboring robots. Using results on average consensus over time-varying graph topologies,we prove that all robots' maps will eventually converge to the actual map of the environment.In addition, we demonstrate that a technique based on topological data analysis, developed in our previous work for generating topological maps, can be readily extended for adaptive thresholding of occupancy grid maps. We validate the effectiveness of our distributed exploration and mapping strategy through a series of 2D simulations and multi-robot experiments.

In this chapter, we propose an approach in which the robots explore the domain while simultaneously building a map online from their own distance measurements and from maps communicated by other robots that they encounter. We present a distributed algorithm for sharing occupancy grid maps among robots in such a way that eventually each robots' occupancy grid map converges to the same global map. As in most occupancy grid mapping strategies, we assume that each robot is either capable of accurately estimating its pose or is equipped with a localization device. It would be an interesting future work to examine how our work translates into the scenario where robots are equipped with only weak localization capabilities, meaning pose information with bounded uncertainty. We also introduce an exploration strategy for the robotic swarm which combines elements from information theory[28] with Lévy walk. We combine random exploration with an information-based approach to obtain the advantages of both methods. We follow the information theoretic approach described in [22, 72] and extend the idea to swarm robotic scenario by combining it with Lévy walks. Finally, we illustrate that a topological data analysis based technique used for generating topological map in Chapter 5 can also be used for adaptive thresholding of occupancy grid maps with a slight modification. The threshold separates occupied grid cells from the unoccupied ones in the map by using tools from *algebraic topology*[60]. A significant difference between our earlier topological approach and the one present in this chapter is the use of cubical complex[73] instead of simplicial complex[38]. Unlike works presented in Chapter 4, Chapter 5 and Chapter 6. Also, unlike our previous works, this mapping strategy computes the map online. The contributions of the paper are enumerated as follows:

- We present a new scalable swarm robotic exploration strategy that combines information-theoretic approaches with Lévy walks.

- We develop a completely decentralized strategy for occupancy grid mapping of unknown environments using a swarm of robots. This strategy does not require robots to have unique identities.

- We demonstrate that a technique based on topological data analysis (TDA), used to generating topological maps in Chapter 5, can be extended for adaptive thresholding of occupancy grid maps.

We validate our mapping approach through 2D simulations using the swarm robot simulator *Stage* [149] on various environments with different sizes and layouts. Furthermore, we conducted experiments to validate our strategy using the commercially available *Turtle 3 Burger* robots.

Division of Work Zahi Kakish conducted the experiments for Section 7.7. The author devised an exploration strategy and a distributed mapping method to map an unknown environment. The simulations were run by the author on the multi-robot simulation platform Stage. Data obtained from these simulations were analyzed by the author in Matlab to validate the technique.

7.1 Related Work

Many existing multi-robot control strategies are extensions of single-robot techniques under centralized schemes or all-to-all communication among robots. An important contribution in this line of work is the generalization of particle filter based approaches to multi-robot systems, with the assumption that robots broadcast their local observations and controls [67]. Another approach is the extension of the Constrained Local Submap Filter technique to multi-robot settings. In this approach, the robots build a local submap and transmit it to a central leader that constructs the global map [152]. Distributed approaches are often required to

design swarm robotic control strategies, due to the limited communication capabilities of individual robots and the requirement to accommodate disconnected and switching robot interaction graph topologies, restricted communication bandwidth, and random link failures. Hence, researchers have pursued numerous efforts in developing distributed techniques for multi-robot mapping. The paper [117] gives an overview of various multi-robot SLAM approaches. A notable work in distributed feature-based multi-robot mapping is Aragues et al. [7]. In this work, the authors present and formally analyze a fully distributed feature-based map merging problem in dynamic robot networks. In brief, the solution to this problem is a distributed Kalman filter expressed in information matrix form. There is also an ample amount of work in the literature on distributed strategies for occupancy grid mapping [58, 115, 70, 19]. In contrast to our work, these works focus on finding approximate relative transformation matrices among the occupancy grid maps from different robots and fusing the maps together through various image processing techniques. In our work, the robots update their occupancy grid maps based on their laser range sensor measurements and from the occupancy grid maps obtained from their neighbors. We also prove that the robots' occupancy grid maps eventually converge to a common map. Since every robot comes to a consensus regarding the map, this map can be obtained by extracting stored occupancy map information from a few robots, and theoretically only one. In this regard, our approach is robust to robot failures. An important aspect of our distributed mapping strategy is that it is a robot label-free approach, meaning that the robots need not identify their neighbors based on any identification labels during communication.

7.2 Problem Statement

We address the problem of estimating the map of an unknown domain $D \subset \mathbb{R}^d$ using distance measurements acquired by a swarm of N_R robots while exploring the domain. We consider bounded, closed, path-connected domains that contain static obstacles. In this chapter, we describe a strategy in which the robots explore an unknown domain using an information correlated Lévy walk, and build the map of the environment online based on its laser range sensor data and by communicating with other robots present locally. Also the robots are unlabeled and have limited communication capabilities. Although in this chapter we only consider the case $d = 2$, our procedure can be easily extended to the case where $d > 2$.

7.2.1 Robot capabilities

We assume that the robots have the following capabilities. Each robot acquires noisy distance measurements using a laser range sensor such as a SICK LMS200 laser rangefinder [123]. Using this data, a robot can detect its distance to obstacles and other robots within its local sensing radius and perform collision avoidance maneuvers if needed. Each robot broadcasts its stored map information, and other robots that are within a distance b_r of the robot can use this information to update their own maps. We assume that each robot can estimate its own pose with no uncertainty. This assumption can be relaxed to the case where the robots are only capable of weak localization, which we define as pose estimation with bounded uncertainty, using a received signal strength indicator (RSSI) device [156] or ultra-wideband (UWB) sensors [106]. The extension of our mapping approach to the case of weak localization is a direction for future work. It is important to note that the robots are not equipped with any sensors that can distinguish between obstacles

and other robots. The robots also do not have unique identifiers.

7.2.2 Representation of the domain as an occupancy grid map

Every robot models the unknown environment as an occupancy grid map, which does not require any *a priori* information about the size of the domain and can be expanded as the robot acquires new distance measurements [142]. Each grid cell of an occupancy grid map is associated with a value that encodes the probability of the cell being occupied by an obstacle. Let M_t^i denote the occupancy grid map stored by robot R^i at time t, where $i \in \{1, ..., N_R\}$. We specify that each robot discretizes the domain with the same resolution. At this resolution, a map of the entire domain $D \subset \mathbb{R}^2$ is discretized uniformly into $|D|$ grid cells, labeled $m_1^i, ..., m_{|D|}^i$. During the mapping procedure, each robot augments its map based on its own distance measurements and map information from nearby robots, effectively adding grid cells to its current map. The occupancy grid map of robot i at time t is represented by the grid cells $m_1^i, ..., m_{|M_t^i|}^i$, where $|M_t^i|$ denotes the number of grid cells in the robot's map at time t. Henceforth, we will usually drop the subscript t from M_t^i to simplify the notation, with the understanding that the map M^i depends on time.

Let $\mathbf{m}_j^i$, $j \in \{1, ..., |M^i|\}$, be a Bernoulli random variable that takes the value 1 if the region enclosed by grid cell m_j^i is occupied by an obstacle, and 0 if it is not. Thus, $\mathbb{P}(\mathbf{m}_j^i = 1)$ is the probability that grid cell m_j^i is occupied, called its *occupancy probability*. A standard assumption for occupancy grid maps is the independence of the random variables $\mathbf{m}_j^i$. As a result, the probability that map M^i belongs to a domain which is completely occupied is given by $\mathbb{P}(M^i) = \prod_{j=1}^{|M^i|} \mathbb{P}(\mathbf{m}_j^i = 1)$. For the sake of brevity, we will use the notation $\mathbb{P}_{m_j^i} \equiv \mathbb{P}(\mathbf{m}_j^i = 1)$ and $\mathbb{P}_{M^i} \equiv \mathbb{P}(M^i)$ throughout the paper. We also define the set $\bar{\mathbf{P}}_{M^i} = \left\{ \mathbb{P}_{m_j^i} \right\}_{j=1}^{|M^i|}$, which is the collection of the occupancy probabilities of all grid cells in map M^i. Finally, the *entropy* $H(M^i)$

of the map M^i, which quantifies the uncertainty in the map, is defined as [142]:

$$H(M^i) = \sum_{j=1}^{|M^i|} \sum_{k=\{0,1\}} \mathbb{P}(\mathbf{m}_j^i = k) \log_2 \left(\mathbb{P}(\mathbf{m}_j^i = k) \right) \tag{7.1}$$

7.2.3 Mapping approach and evaluation

Our mapping approach consists of the following steps. All robots explore the domain simultaneously using the random walk strategy that is defined in Section 7.3. While exploring, each robot updates its occupancy grid map with its own distance measurements, broadcasts this map to neighboring robots, and then modifies its map with the maps transmitted by these neighboring robots using a predefined discrete-time, consensus-based protocol, which is discussed in Section 7.4. We prove that the proposed protocol guarantees that every robot's map will eventually converge to a common map. A technique for post-processing the occupancy grid map based on topological data analysis (TDA) is presented in Section 7.5. We evaluate the performance of our mapping approach according to two metrics: (1) the percentage of the entire domain that is mapped after a specified amount of time, and (2) the entropy of the final occupancy grid map, as defined in Equation 7.1.

7.3 Exploration Based On Information Correlated Lévy Walk

In this section, we describe the motion strategy used by robots to explore the unknown domain. Exploration strategies for robotic swarms generally use *random*, *guided*, or *information-based* approaches [142, 58]. Random exploration approaches are often based on Brownian motion (e.g., [51, 32, 155]) or Lévy walks (e.g., [34, 128, 49]), which facilitate uniform dispersion of the swarm throughout a domain from any initial distribution. Moreover, these approaches do not rely on centralized motion planning or extensive inter-robot communication, which can scale poorly

with the number of robots in the swarm. Information-based approaches, such as [22, 72], guide robots in the direction of maximum information gain based on a specified metric, which can increase the efficiency of exploration compared to random approaches. *Mutual information* (or *information gain*), a measure of the amount of information that one random variable contains about another [28], is a common metric used to assess the information gain that results from a particular action by a robot. This metric can be used to predict the increase in certainty about a state of the robot's environment that is associated with a new sensor measurement by the robot.

We specify that each robot in the swarm performs a combination of random and information-based exploration approaches, in order to benefit from the advantages of both types of strategies. We refer to this exploration strategy as an *information correlated Lévy walk* and describe its implementation in this section.

To execute a Lévy walk, a robot repeatedly chooses a new heading and moves at a constant speed [159] in that direction over a random distance that is drawn from a heavy-tailed probability distribution function $p(l)$, of the form

$$p(l) \propto l^{-\alpha}, \tag{7.2}$$

where α is the Lévy exponent. The case $\alpha \geq 1$ signifies a scale-free *superdiffusive* regime, in which the expected displacement of a robot performing the Lévy walk over a given time is much larger than that predicted by random walk models of uniform diffusion. This superdiffusive property disperses the robots quickly toward unexplored regions.

We define the heading chosen by the robot before each step in the Lévy walk as the direction that maximizes the robot's information gain about the environment. This is computed as the direction that maximizes the mutual information between

the robot's current occupancy grid map and the distance measurements that it is likely to obtain when it moves in that direction, based on the forward measurement model of a laser range sensor [142] over a finite time horizon. These measurements are expected to decrease the entropy of the robot's occupancy grid map, defined in Equation 7.1. Therefore, the computed robot heading is more likely to direct the robot to unexplored regions than a uniformly random heading. The calculation of this heading is described in the following subsections.

7.3.1 Laser range sensor forward measurement model

We assume that the laser range sensor of each robot R^i has N_l laser beams that all lie in a plane parallel to the base of the robot. The distance measurement obtained by the a^{th} laser beam of robot R^i at time τ is a random variable that will be denoted by $z_\tau^{i,a}$. The random vector of all distance measurements obtained by robot R^i at time τ is represented as $\mathbf{z}_\tau^i = [z_\tau^{i,1} \ \cdots \ z_\tau^{i,a} \ \cdots \ z_\tau^{i,N_l}]^T$.

Define s_{min} and s_{max} as the minimum and maximum possible distances, respectively, that can be measured by the laser range sensor. In addition, let δ denote the actual distance of an obstacle that is intersected by the a^{th} laser beam of robot R^i. The Gaussian distribution function with mean μ and variance σ^2 will be written as $\mathcal{N}(\mu, \sigma^2)$. We define the probability density function of the distance measurement $z_\tau^{i,a}$, given the actual distance δ, as the forward measurement model presented in [22, 125],

$$\mathbb{P}(z_\tau^{i,a} \mid \delta) = \begin{cases} \mathcal{N}(0, \sigma^2), & \delta \leq s_{min} \\ \mathcal{N}(s_{max}, \sigma^2), & \delta \geq s_{max} \\ \mathcal{N}(\delta, \sigma^2), & \text{otherwise,} \end{cases} \tag{7.3}$$

where σ^2 is the variance of the range sensor noise in the radial direction of the laser

beam. Although this model does not incorporate range sensor noise in the direction perpendicular to the laser beam, the experimental results in [22] and our results in Section 7.7 demonstrate that the model captures sufficient noise characteristics for generating accurate maps from the sensor data.

7.3.2 Robot headings based on mutual information

The mutual information between two random variables **A** and **B** is defined as the *Kullback-Leibler distance* [28] between their joint probability distribution, $\mathbb{P}(\mathbf{A}, \mathbf{B})$, and the product of their marginal probability distributions, $\mathbb{P}(\mathbf{A})\mathbb{P}(\mathbf{B})$:

$$\mathbf{I}[\mathbf{A}; \mathbf{B}] = \mathbb{KLD}\left(\mathbb{P}(\mathbf{A}, \mathbf{B}) || \mathbb{P}(\mathbf{A})\mathbb{P}(\mathbf{B})\right) \tag{7.4}$$

This quantity measures how far **A** and **B** are from being independent. In other words, $\mathbf{I}[\mathbf{A}; \mathbf{B}]$ quantifies the amount of information that **B** contains about **A**, and vice versa. For example, if **A** and **B** are independent random variables, then no information about **A** can be extracted from the outcomes of **B**, and consequently, $\mathbf{I}[\mathbf{A}; \mathbf{B}] = 0$. On the other hand, if **A** is a deterministic function of **B**, then the entropies of both random variables are equal to the expected value of $-\log_2(\mathbb{P}(\mathbf{A}))$, and $\mathbf{I}[\mathbf{A}; \mathbf{B}]$ is equal to this quantity, which is its maximum value.

During each step in its random walk, every robot performs the following computations and movements. A new step may be initiated either when the robot completes its previous step, or when the robot encounters an obstacle (or other robot) during its current step. Suppose that the next step by robot R^i starts at time τ. At this time, the robot computes the duration T of the step by generating a random distance based on the Lévy distribution (Equation 7.2) and dividing this distance by its speed s^i, which is constant. Also at time τ, the robot computes the velocity $\mathbf{v}^i_{\Delta t}$ that it will follow during the time interval $\Delta t := [\tau \;\; \tau + T]$. This com-

putation involves several variables, which we introduce here. The pose of robot R^i at time τ is denoted by $\mathbf{x}^i_\tau$. We define a sequence of this robot's poses during the time interval Δt as $\mathbf{X}^i_{\Delta t} := [\mathbf{x}^i_\tau \ \cdots \ \mathbf{x}^i_{\tau'} \ \cdots \ \mathbf{x}^i_{\tau+T}]$, where $\tau' \in \Delta t$. We also define $\mathbf{Z}^i_{\Delta t} := [\mathbf{z}^i_\tau, \cdots, \mathbf{z}^i_{\tau'}, \cdots, \mathbf{z}^i_{\tau+T}]$ as a set of random vectors modeling laser range sensor measurements that the robot is expected to receive as it moves during this time interval. At time τ, robot R^i calculates its velocity $\mathbf{v}^i_{\Delta t}$ as the solution ${}^*\mathbf{v}^i_{\Delta t}$ to the following optimization problem, with the objective function defined as in [22, 71] :

$$ {}^*\mathbf{v}^i_{\Delta t} = \underset{\|\mathbf{v}^i_{\Delta t}\|=s^i, \ \angle\mathbf{v}^i_{\Delta t}\in[-\pi,\pi]}{\arg\max} \frac{\mathbf{I}[M^i; \ \mathbf{Z}^i_{\Delta t} \mid \mathbf{X}^i_{\Delta t}]}{C(\mathbf{v}^i_{\Delta t})}, \tag{7.5} $$

where $\mathbf{I}[M^i; \ \mathbf{Z}^i_{\Delta t} \mid \mathbf{X}^i_{\Delta t}]$ represents the mutual information between the robot's occupancy grid map and its distance measurements given a sequence of the robot's poses. The term $C(\mathbf{v}^i_{\Delta t})$ in Equation 7.5 penalizes the robot for large deviations from its current heading when multiple velocities generate different paths with the same mutual information. We define $C(\mathbf{v}^i_{\Delta t})$ as the Euclidean norm of the difference between $\mathbf{v}^i_{\Delta t}$ and the robot's current velocity. Based on the current occupancy grid map M^i of robot R^i and its set of expected poses $\mathbf{X}^i_{\Delta t}$ under its velocity command $\mathbf{v}^i_{\Delta t}$, R^i can compute the probability distribution of its laser range sensor measurements using the forward measurement model Equation 7.3.

7.3.3 Computing mutual information

In this section, we describe the computation of the objective function in Equation 7.5 and discuss techniques for solving the associated optimization problem. We first focus on computing $\mathbf{I}[M^i; z^{i,a}_\tau]$, the mutual information between the measurement $z^{i,a}_\tau$ obtained by the a^{th} laser beam of robot R^i at time τ and the robot's current occupancy grid map M^i. Grid cells in the map that do not intersect the beam do not to contribute to the mutual information. Hence, the task of comput-

ing $\mathbf{I}[M^i; z_\tau^{i,a}]$ reduces to computing $\mathbf{I}[\mathbf{c}_\tau^{i,a}; z_\tau^{i,a}]$, where $\mathbf{c}_\tau^{i,a}$ is the collection of Bernoulli random variables $\mathbf{m}_j^i$ modeling the occupancy of grid cells in the map of robot R^i that are intersected by the a^{th} beam at time τ. This quantity is defined as:

$$\mathbf{I}[\mathbf{c}_\tau^{i,a}; z_\tau^{i,a}] = \int_{z \in z_\tau^{i,a}} \sum_{c \in \mathbf{c}_\tau^{i,a}} \mathbb{P}(c, z) \log_2 \left(\frac{\mathbb{P}(c, z)}{\mathbb{P}(c)\mathbb{P}(z)} \right) dz, \tag{7.6}$$

where $\mathbb{P}(c, z)$ is the joint probability distribution of c and z, and $\mathbb{P}(c)$ and $\mathbb{P}(z)$ are the probability distributions of the occupancy probabilities of the intersected grid cells and the range sensor distance measurements, respectively. We show in Appendix C that $\mathbf{I}[\mathbf{c}_\tau^{i,a}; z_\tau^{i,a}]$ can be expressed as:

$$\mathbf{I}[\mathbf{c}_\tau^{i,a}; z_\tau^{i,a}] = -\int_{z_\tau^{i,a}} \mathbb{P}(z) \log_2(\mathbb{P}(z)) dz + K, \tag{7.7}$$

where $K = -\log(\sqrt{2\pi}\sigma) - 0.5$. Since K is not a function of the map or the distance measurements, it does not affect the solution to the optimization problem in Equation 7.5 and therefore does not need to be included in this problem. The effect of $\mathbf{c}_\tau^{i,a}$ on $\mathbf{I}[\mathbf{c}_\tau^{i,a}; z_\tau^{i,a}]$ is through the probability distribution $\mathbb{P}(z)$ in Equation 7.7. We now compute this distribution. From the forward measurement model Equation 7.3, $\mathbb{P}(z)$ is completely determined by the distance δ from the laser range sensor to the closest occupied cell in $\mathbf{c}_\tau^{i,a}$. Let e_p denote a binary sequence of length $|\mathbf{c}_\tau^{i,a}|$ in which each of the first $p-1$ elements is 0 and the p^{th} element is 1. The remainder of the elements in the sequence can be either 0 or 1. This sequence is a possible realization of $\mathbf{c}_\tau^{i,a}$, in which the first $p-1$ intersected grid cells are unoccupied, the p^{th} cell is occupied, and the remaining cells may or may not be occupied. For compactness of notation, we define e_0 as the sequence in which all elements are 0; that is, no intersected grid cells are occupied.

Then, we have that

$$\mathbb{P}(z) = \sum_{p=0}^{|\mathbf{c}_\tau^{i,a}|} \mathbb{P}(z \mid \mathbf{c}_\tau^{i,a} = e_p) \mathbb{P}(\mathbf{c}_\tau^{i,a} = e_p) \tag{7.8}$$

We direct the reader to [22, 72] for a detailed description of such sensor models.

We can now extend our computation of the mutual information for a single distance measurement at a given time to $\mathbf{I}[M^i; \mathbf{Z}^i_{\Delta t} \mid \mathbf{X}^i_{\Delta t}]$, the mutual information for all distance measurements taken by robot R^i over a sequence of times. Since the exact computation of this quantity is intractable, we adopt a common technique used in the robotics literature: we select several laser beams on the robot and assume that the measurements from these beams are independent of one another [78, 72]. We define $\mathcal{Z}^i_{\Delta t}$ as the set of distance measurements obtained at times $\tau' \in \Delta t$ from the selected laser beams on robot R^i, indexed by $a' \in \{1, ..., N_l\}$. Then, we can approximate $\mathbf{I}[M^i; \mathbf{Z}^i_{\Delta t} \mid \mathbf{X}^i_{\Delta t}]$ as the following sum over $\mathcal{Z}^i_{\Delta t}$:

$$\mathbf{I}[M^i; \mathbf{Z}^i_{\Delta t} \mid \mathbf{X}^i_{\Delta t}] \approx \sum_{z^{i,a'}_{\tau'} \in \mathcal{Z}^i_{\Delta t}} \mathbf{I}[M^i; z^{i,a'}_{\tau'}] \tag{7.9}$$

In general, finding $\mathcal{Z}^i_{\Delta t} \subseteq \mathbf{Z}^i_{\Delta t}$ that best approximates the formula in Equation 7.9 is an NP-hard problem [22]. Therefore, no approximation algorithm can be designed to find this $\mathcal{Z}^i_{\Delta t}$ in polynomial time. In spite of this, generating $\mathcal{Z}^i_{\Delta t}$ using greedy algorithms has shown promising results [78, 72, 22], which prompted us to use a similar technique of selecting the laser beams having an information gain above a predefined threshold [78]. Now that we have obtained an expression for $\mathbf{I}[M^i; \mathbf{Z}^i_{\Delta t} \mid \mathbf{X}^i_{\Delta t}]$, we can solve the optimization problem in Equation 7.5 and thereby find the robot heading which maximizes the information gain.

An alternate approach to solving the optimization problem in Equation 7.5 is to compute the gradient of the objective function and define the robot's heading as the direction of gradient ascent. However, since the computations are performed on a discrete occupancy grid map, it is not clear that the objective function has a well-defined gradient. Although prior attempts have been made to compute the gradients of information-based objective functions under particular assumptions

[71, 21], the gradient computation relies on numerical techniques such as finite difference methods.

7.4 Occupancy grid map updates by each robot

While exploring the environment, each robot updates its occupancy grid map based on its laser range sensor measurements and the occupancy grid map information broadcast by robots that are within a distance b_r. In this section, we describe how robot R^i updates $\bar{\mathbf{P}}_{M^i}$, the collection of occupancy probabilities of all cells in its map, using both its distance measurements and the sets $\bar{\mathbf{P}}_{M^{\hat{n}}}, \hat{n} \in \mathbb{N}_\tau^i$, where $\mathbb{N}_\tau^i$ denotes the set of robots that are within distance b_r of robot R^i at time τ. We present a discrete-time, consensus-based protocol for modifying the occupancy map of each robot and prove that this protocol guarantees that all robots eventually arrive at a consensus on the map of the environment. As explained in Section 7.4.4, our method for updating the occupancy map is resilient to false positives, meaning that even if a robot incorrectly assigns a high occupancy probability $\mathbb{P}_{m_j^i}$ to a free grid cell j due to noise in its distance measurement, the impact of this noisy measurement on $\mathbb{P}_{m_j^i}$ is eventually mitigated due to the averaging effect of our map modification protocol. Since occupancy grid mapping algorithms require the robots' pose information, we assume that each robot can estimate its own pose using an accurate localization technique.

7.4.1 Updating occupancy map based on laser range sensor measurements

The forward sensor measurement model Equation 7.3 represents the probability that a robot obtains a particular distance measurement given the robot's map of the environment and the robot's pose. The parameter δ in the model can be computed from the robot's map and pose. Commonly used occupancy grid mapping algo-

rithms [142, 44] use an inverse sensor measurement model to update the occupancy probabilities of the grid cells. This type of model gives the probability that a grid cell is occupied, given the laser range sensor measurements and the pose of the robot. Although forward sensor measurement models can be easily derived for any type of range sensor, inverse sensor measurement models are more useful for occupancy grid algorithms [142]. Methods such as supervised learning algorithms and neural networks have been used to derive inverse sensor models based on a range sensor's forward model [139]. Pathak et al. [103] describe a rigorous approach to deriving an analytical inverse sensor model for a given forward sensor model. Although inverse sensor models derived from forward sensor models can be used to efficiently estimate an occupancy grid map, it is difficult to develop a distributed version of such models, since either their computation is performed offline [139] or the mapping between the forward and inverse sensor models is nonlinear [103]. These difficulties preclude us from exploiting these techniques in our mapping approach.

Instead, we propose a heuristic inverse range sensor model for which a distributed version can be easily derived. We specify that each robot estimates its pose and obtains distance measurements at discrete time steps, to reflect the fact that sensor measurements are recorded at finite sampling rates. Let $\mathbf{x}_k^i$ denote the pose of robot R^i at time step k, and let $\mathbf{z}_k^i$ be the vector of its distance measurements at this time step. Our inverse sensor model, which we refer as an *update rule*, is a function $u : (m_j^i, \mathbf{x}_k^i, \mathbf{z}_k^i) \rightarrow [0, 1]$. This function assigns an occupancy probability to grid cell m_j^i based on the robot's pose and all of its distance measurements at time step k. Robot R^i uses this function to modify $\bar{\mathbf{P}}_{M^i}$ based on its distance measurements. We define the update rule in terms of a function $l : (m_j^i, \mathbf{x}_k^i, z_k^{i,a}) \rightarrow [0, 1]$, which assigns an occupancy probability to grid cell m_j^i based on the robot's pose

and its a^{th} laser beam's distance measurement at time step k. The function can be applied only to those grid cells m_j^i that are intersected by the a^{th} beam at time step k. We define l as one of two functions, l_r and l_u, depending on whether the robot estimates that (?) its a^{th} laser beam is reflected (l_r) or not reflected (l_u). These functions depend on $s^a_{m_j^i}$, the distance from the center of cell m_j^i to the a^{th} laser range sensor of robot R^i, and constants p_a, p_f, and p_{hit}. The functions l, l_r, and l_u are defined as follows:

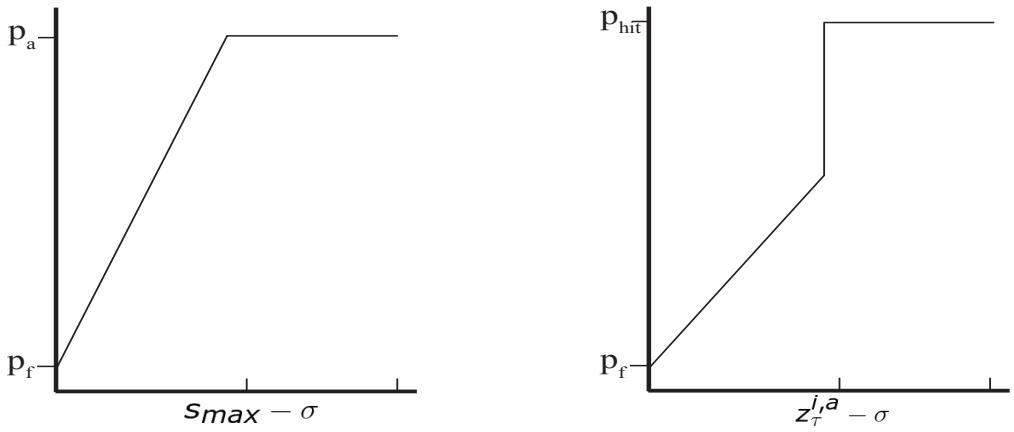
$$l(m_j^i, \mathbf{x}_\tau^i, z_k^{i,a}) = \begin{cases} l_r(s^a_{m_j^i}, z_k^{i,a}) & z_k^{i,a} \leq s_{max} - \sigma \\ l_u(s^a_{m_j^i}) & z_k^{i,a} > s_{max} - \sigma \end{cases} \tag{7.10}$$

$$l_r(s^a_{m_j^i}, z_k^{i,a}) = \begin{cases} \frac{p_a - p_f}{s_{max}} s^a_{m_j^i} + p_f & s^a_{m_j^i} < z_k^{i,a} - \sigma \\ p_{hit} & s^a_{m_j^i} \geq z_k^{i,a} - \sigma \end{cases} \tag{7.11}$$

$$l_u(s^a_{m_j^i}, s_{max}) = \begin{cases} \frac{p_a - p_f}{s_{max}} s^a_{m_j^i} + p_f & s^a_{m_j^i} < s_{max} - \sigma \\ p_a & s^a_{m_j^i} \geq s_{max} - \sigma \end{cases} \tag{7.12}$$

Figure 7.1 illustrates the functions l_r and l_u that are defined in Equation 7.11 and Equation 7.12, respectively.

Now using the function l, we can define our update rule. We describe the mapping from the input to the output of the update rule in the pseudocode in 4. Steps 2 to 4 in the algorithm identifies and store the distance measurements in a set ζ, for which the corresponding laser beams' path's horizontal projection intersected with the input grid cell m_j^i. To make the update rule well defined for any input grid cell, we define $u(m_j^i, \mathbf{x}_k^i, \mathbf{z}_k^i) = 1$ if none of the measurements in $\mathbf{z}_k^i$ provide any information about m_j^i (Step 5). Finally, Step 5 of 4 returns the maximum



(a) unreflected model (b) reflected model

Figure 7.1: Illustrative plots of the functions (a) l_u and (b) l_r. The x-axis in both plots measures the distance between the a^{th} laser range sensor of robot R^i and any grid cell that intersects the beam from this laser, which yields the measurement $z_\tau^{i,a}$.

value of $l(m_j^i, \mathbf{x}_\tau^i, z)$ when evaluated over all elements z in set ζ. Our simulation and experimental results, presented in Section 7.6 and Section 7.7, demonstrate the effectiveness of our proposed update rule.

7.4.2 Occupancy grid map sharing among robots

During exploration of the unknown environment, each robot in addition to updating its occupancy grid map based on its laser range sensor measurements, broadcasts its current occupancy grid map locally and receives occupancy grid maps from robots in its local neighborhood. It is important to note that the maps each robot receives do not contain any information about the robots which broadcasted those maps. As mentioned earlier, an important property of our distributed mapping approach is that it is robust to failure of a subset of robots. This is due to the fact that by following our discrete time consensus based map modification

Algorithm 4 Update rule function: $u_m(m_j^i, \mathbf{x}_k^i, \mathbf{z}_k^i)$

Input: m_j^i(grid cell of interest), $\mathbf{x}_k^i$(pose of i^{th} robot at k^{th} epoch),$\mathbf{z}_k^i$(laser sensor measurements of i^{th} robot at k^{th} epoch)

Output: a value from the interval [0,1].

function $u_m(m_j^i, \mathbf{x}_k^i, \mathbf{z}_k^i)$) ▷ L This function returns the probability of occupancy of grid cell m_j^i based on $\mathbf{x}_k^i$ and $\mathbf{z}_k^i$. By definition the function returns one if the occupancy of m_j^i can be inferred from $\mathbf{x}_k^i$ and $\mathbf{z}_k^i$.

    $\zeta = \emptyset$ ▷ a set to store the laser beam measurements results from laser beams that either passed through m_j^i or was reflected by an obstacle covering m_j^i

    for $z_k^{i,a} \in \mathbf{z}_k^i$ **do** ▷ Iterate through every laser measurement in $\mathbf{z}_k^i$

        if contains_hori_proj(m_j^i, $z_k^{i,a}$) **then**

            $\zeta \leftarrow \zeta \cup z_k^{i,a}$

        end if

    end for

    if $\zeta == \emptyset$ **then** ▷ m_j^i does not contain the horizontal projection path of any laser beam. Therefore, $\mathbf{z}_k^i$ can not used to infer the occupancy of m_j^i.

        return 1

    end if ▷ If ζ is non empty return the maximum value of the single laser beam occupancy probability assignment function evaluated over the set ζ

    return $\max_{z \in \zeta} \{l(m_j^i, \mathbf{x}_\tau^i, z)\}$

end function

protocol Equation 7.13, each robot's map eventually converges to a common map and therefore the structure of the explored environment can be reconstructed using the map information stored in a few surviving robots (theoretically, only one robot needs to survive).

Let $\mathbb{G}(k) = (\mathbb{V}, \mathbb{E}(k))$ be the undirected time varying communication graph. The nodes represent robot indices, $\mathbb{V} = \{1, ..., i, ..., N_R\}$. At a time step k, if two robots $R^i, R^{\hat{n}}$ are in broadcasting range of each other and exchange information, then there is an edge $(i, \hat{n}) \in \mathbb{E}(k)$. We formally define $\mathbb{N}_k^i$, the neighbor robots' indices set of R^i: $\mathbb{N}_k^i = \{\hat{n} | (i, \hat{n}) \in \mathbb{E}(k), i \neq \hat{n}\}$. Before we describe the method by which R^i updates $\bar{\mathbf{P}}_{M^i}$ based on the occupancy grid map from its neighbors, we give an overview about average consensus over time-varying graph topologies. This forms the basis for our discrete-time, consensus-based protocol for map modification.

7.4.3 Average consensus over time-varying graph topologies

Linear consensus protocols over time-varying graphs have been well-studied in the literature [98, 109, 89, 76]. The main results of these works assume the existence of a time interval during which the union of communication graphs contains a spanning tree, which is required in order to reach consensus. In this work, we use results from Kingston et al. [76], which focuses on average consensus in a discrete-time setting. We use notation from [56] to explain the graph-theoretic concepts. The notation is local to this subsection and should not be confused with notation used elsewhere in this paper.

Let $\mathbb{A}[k] = [a_{ij}(k)]$ be the adjacency matrix associated with the time varying undirected communication graph $G(k) = V, E(k)$ at time step k. $V = 1, ..., n$ is the vertex set of G and edge from node i to j exists at time step k if and only if $a_{ij} \neq 0$. The neighbors of node i at time step k can be determined by the elements

in row i of $\mathbb{A}[k]$ which are nonzero. If a node i stores a real scalar information x_i, then V is said to be in average consensus if $x_i = x_j = \frac{1}{n}\sum_i x_i[0]$ for all i, j. This can be achieved if every node updates its information x_i to some weighted linear combination of its neighbors values and x_i. This information dynamics can be written as $x[k+1] = \mathbb{A}[k]x[k]$. It is proved in [76, Theorem 1] that the information dynamics reaches average consensus asymptotically provided that $\mathbb{A}[k]$ is doubly stochastic, meaning that the rows and columns of $\mathbb{A}[k]$ sum to unity, and if there exists a time interval such that the union of communication graphs over this time interval is connected. We use these results to prove an important result in Section 7.4.4.

7.4.4 Consensus based occupancy grid map sharing

In this section, we will describe our discrete time consensus based map modification protocol and prove its asymptotic convergence. As mentioned earlier, each robot explores the unknown domain based on the strategy delineated in Section 7.3. Besides exploring the unknown environment, at every time step each robot updates its internally stored occupancy grid map based on its laser range sensor measurements and occupancy grid maps received from its neighboring robots. The following equation rigorously formulates our discrete-time, consensus-based map modification protocol when applied to the map's grid cell m_j^i:

$$\mathbb{P}_{m_j^i}(k+1) = \prod_{\hat{n}\in\mathbb{N}_k^i \cup i} \left(\mathbb{P}_{m_j^{\hat{n}}}(k)\right)^{a_{i,\hat{n}}(k)} \times u(m_j^i, \mathbf{x}_k^i, \mathbf{z}_k^i) \tag{7.13}$$

If we stack the outputs of $u(m_j^i, \mathbf{x}_k^i, \mathbf{z}_k^i)$ corresponding to every robot at k^{th} time step, we obtain a vector which we denote as $\mathbf{u}_j[k]$. We define $\mathbf{d}$ as a subsequence of the discrete infinite sequence $\{0, 1, ..., \infty\}$, such that $\mathbf{d} = \{d|\ \mathbf{u}_j[d] \neq 1\}$. By using the notation $\mathbf{u}_j[d] \neq 1$ we mean that at least one of the elements in vector $\mathbf{u}_j[d]$ is

not unity. $< \cdot >_{gm}$ is a geometric mean operator which takes a vector as an input and return geometric mean of its elements. If $\mathbf{q} = [q_1, q_2, ..., q_n]$ then $< \mathbf{q} >_{gm} = \sqrt[n]{q_1 \cdot q_2 \cdots q_n}$. $\mathbb{P}_{m_j}[k]$ denotes the vector obtained by stacking $\mathbb{P}_{m_j^i}(k)\ \forall i \in \mathbb{V}$. Now we are in position to state the theorem which is an important result of this paper.

Theorem 7. *If each robot updates its occupancy grid according to the update rule shown in Equation 7.13 then under the assumptions stated below(Assumption 1 - Assumption 3) yields,*

$$\lim_{k\to\infty} \mathbb{P}_{m_j^i}[k] = < \mathbb{P}_{m_j}[0] >_{gm} \cdot \prod_{d\in\mathbf{d}} < \mathbf{u}_j[d] >_{gm} \tag{7.14}$$

Proof. See Appendix D □

Assumption 1. *There exists a time interval such that the union of interaction graph $\mathbb{G}$ over this time interval is connected.*

Assumption 2. *At every instant of time , each robot interacts pairwise with its neighbors and with $a_{i,\hat{n}}(k) = 0.5$ or $a_{i,\hat{n}}(k) = 1$(if $\mathbb{N}_k^i = \emptyset$). In other words, $\mathbb{N}_k^i$ is a singleton or a null set at each time step k.*

Assumption 3. *The set* **d** *is finite.*

Assumption 1 is required to use the results from [76, Theorem 1]. In reality, it is hard to prove that this assumption would hold true for robots exploring arbitrary domains. But one could guess that the assumption is almost surely satisfied as N_R tends to infinity and the results in Section 7.6 and Section 7.7 support our intuition. The pairwise interaction assumption(Assumption 2) is one way to ensure that adjacency matrix corresponding to $\mathbb{G}(k)$ is doubly stochastic, which is required to prove Theorem 7. Suppose at a time step k there are multiple robots around a particular robot within its information transmission range, then in order to make

the Assumption 2 hold, the robot updates its map using the information from one of the neighbors only at that time step. The last assumption(Assumption 3) is used in the proof shown in Appendix D to establish convergence of a sequence summation. In practice Assumption 3 can be realized by programming the robots to update its $\mathbb{P}_{m_j^i}$ using its laser range measurements at the most $\mathbb{U}$, a fixed finite number. The results presented in Section 7.6 and Section 7.7 throws light on the fact that, in practice it is not required to impose Assumption 3 explicitly, as we did not enforce any such condition while conducting simulations and experiments for this chapter.

Theorem 7 delineates that if each robot R^i updates its $\mathbb{P}_{m_j^i}$ according to the update rule Equation 7.13, then $\mathbb{P}_{m_j^i} \forall i \in \{1, ..., N^R\}, j \in \{1, ..., M^i\}$ would eventually reach a value which is proportional to $\prod_{d \in \mathbf{d}} < \mathbf{u}_j[d] >_{gm}$. The proportionality constant is geometric mean of elements of the vector $\mathbb{P}_{m_j}[0]$. The proportionality constant can be made unity if we choose $\mathbb{P}_{m_j^i} = 1$ as initial condition for the update rule Equation 7.13. In this manner, the asymptotic behavior of update rule Equation 7.13 is only dependent on the laser range sensor measurements made by the robots. Asymptotic value of $\mathbb{P}_{m_j^i}$ is proportional to the geometric mean of laser range sensor measurements made by all robots at various time steps. As a result, the asymptotic value of $\mathbb{P}_{m_j^i}$ will converge to a probability value indicating the true occupancy of the grid cell even if a few robots reported highly noise or incorrect measurements.

7.5 Post Processing of occupancy grid maps

In this section, we propose a technique for post processing the occupancy grid map generated by the robots. Post processing of occupancy grid map refers to inferring the most likely occupancy grid map of the explored environment based

on the probability of occupancy of each grid. A common approach to occupancy grid map inference problem is through a procedure called *Maximum A Posterior (MAP)* occupancy grid mapping[142]. MAP procedure computes the occupancy grid map which has maximum probability of occurrence based on probability of occupancy of each grid cell in the map. In general, the MAP problem is posed as an optimization problem and the solution is computed using gradient based hill climbing methods. The approach is computational expensive as one has to perform gradient ascent from different initial conditions to escape from local minima. Also the search space is exponential in the number of grids cell because for a given set of n grid cells there exist a set containing 2^n possible occupancy grid maps[142].

Alternately, our approach is based on concepts from algebraic topology[60] and for most practical purpose has a time complexity which is linear in the number of grid cells($\mathcal{O}(M)$)[134]. Our approach is based on *topological data analysis*(TDA)[38], an applied version of algebraic topology. In the next section, we provide the reader with the basic concepts of TDA and algebraic topology required to understand our work. An in depth treatment of these subjects can be found in [38, 60, 73, 54].

7.5.1 Algebraic topology and TDA

Recently there has been a lot of success in estimating the underlying structure and shape of the data using tools from algebraic topology [18]. Understanding the underlying shape of data a priori would aid in its efficient analysis using statistical techniques like regression [39]. Topological data analysis (TDA) is a collection of algorithms constructed using algebraic topology [60] for performing coordinate free topological and geometric analysis of noisy data. In general, for most applications, data are obtained by noisy sampling of an intensity map supported on a Euclidean domain. This noisy sampled data are referred as a *point cloud*. The dominant

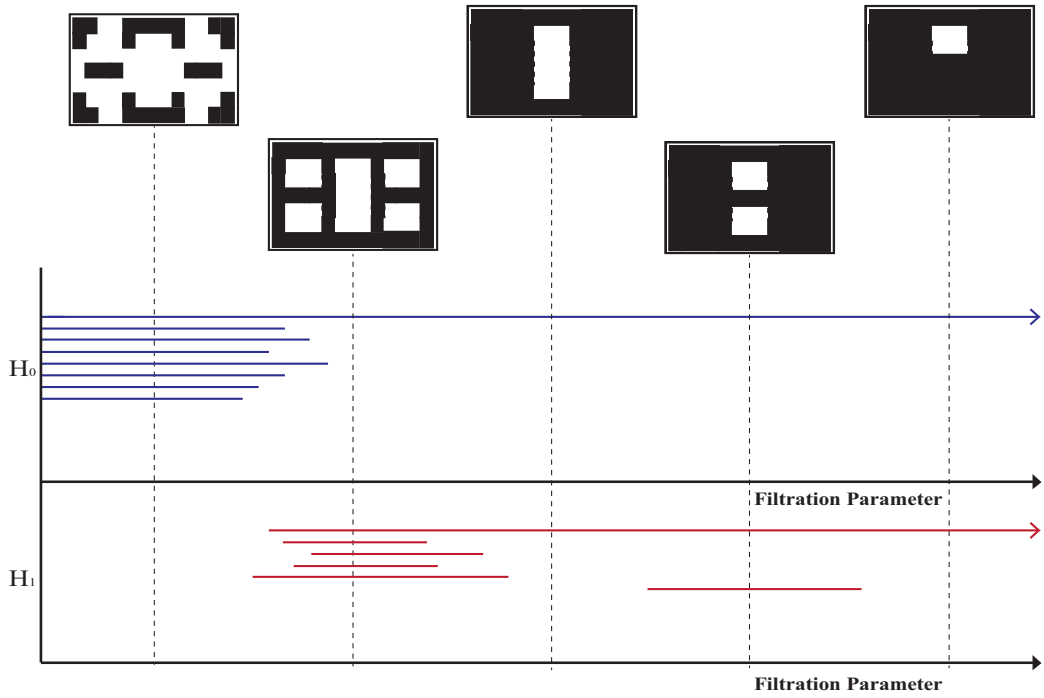


Figure 7.2: An example barcode diagram of a filtration constructed from cubical complex. The arrows in H_0 and H_1 indicate the persistent topological features over variation of the filtration parameter. The shaded regions contain the 2 dimensional *elementary cubes*(squares). In this illustration, one can find that the cubical complex has one persistent topological feature corresponding to each homology H_0 and H_1.

topological features of the domain from which the point cloud is sampled can be computed using TDA. A compact graphical representation of this information is presented using barcode diagrams [54] and persistence diagrams [39].

The vanguard concept of TDA is persistent homology, which basically enables the study of *global* topological facets of a space by performing computations *locally* on the noisy point cloud obtained by sampling the space. Topological features that persist over a longer range of scale can be identified using persistent homology. Unlike our previous works [133, 134], we focus on cubical homology rather

that simplicial homology. Let $\mathbf{T}$ be a topological space which admits a simplicial decomposition, then one can affiliate a sequences of vector spaces with it called *homology groups*, denoted by $H_t(\mathbf{T})$, $t = 0, 1, 2, ...$ Every topological feature information regarding $\mathbf{T}$ is encoded in every one of these vector spaces. The dimension of $H_t(\mathbf{T})$ quantifies the number of independent topological features encoded by $H_t(\mathbf{T})$. These topologically invariant quantities denoted by β_t has the nomenclature *Betti numbers* [54]. Intuitively speaking, β_t gives the number of independent k-dimensional cycles in the topological space. β_0 and β_1 gives the number of connected components in $\mathbf{T}$ and number of holes in $\mathbf{T}$ respectively, if $\mathbf{T}$ is embedded in $\mathbb{R}^2$, denoted by $\mathbf{T} \hookrightarrow \mathbb{R}^2$.

We now define the most basic building block of cubical complex termed as *elementary interval*[73]. An elementary interval is a closed interval $\mathtt{I} \subset \mathbb{R}$ of the form $\mathtt{I} = [l, l+1]$ or $\mathtt{I} = [l, l]$ for some $l \in \mathbb{Z}$. Elementary intervals are called degenerate if it contains only one point, alternately those with unit length are termed nondegenerate. A *cube* or *elementary cube* $\mathbb{Q}$ is constructed by taking finite product of elementary interval: $\mathbb{Q} = \prod_{t=1}^{dim} \mathtt{I}_t \subset \mathbb{R}^{dim}$, where each $\mathtt{I}_t$ is an elementary interval[73]. A set $\mathbb{X} \subset \mathbb{R}$ is a *cubical set* or *cubical complex* if $\mathbb{X}$ can decomposed as a finite union of elementary cubes. If $\mathbb{Q}$ and $\mathbb{O}$ are elementary cubes and $\mathbb{Q} \subset \mathbb{O}$, then $\mathbb{Q}$ is a face of $\mathbb{O}$. For a topological space $\mathbf{T}$, let t-cube ($\square_t$) be a continuous map $\square_t : [0, 1]^t \rightarrow \mathbf{T}$ [81]. As mentioned earlier one can create a cubical complex by connecting t-cubes of different dimensions. A t-cube consists of $2t$ faces which are $t-1$ dimensional cubes. Like in the case of simplicial complex [52], cubes in a cubical complex also must fit together in some nice fashion. In order for $\mathbb{K}$ to be a cubical complex, it must meet the following the requirements: 1) a face $\square_{t-1}$ of cube $\square_t \in \mathbb{K}$ must also be in $\mathbb{K}$, 2) the intersection of any two cubes $\square_t$ and $\square'_t$ is either an empty set or a common face to both $\square_t$ and $\square'_t$.

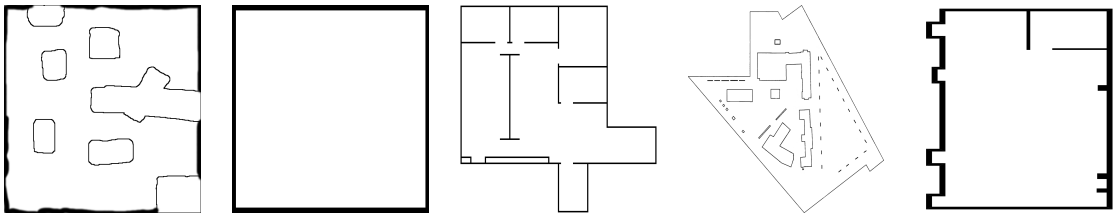
Let $f : \kappa \rightarrow \mathbb{R}$ be a function such that $\eta \leq \iota$ implies that $f(\eta) \leq f(\iota)$. Then $f^{-1}((-\infty, \varpi])$ is a cubical complex denoted by κ_ϖ, and $\varpi_1 \leq \varpi_2$ implies that $\kappa_{\varpi_1} \subseteq \kappa_{\varpi_2}$, thus yielding a *filtration of cubical complexes* with ϖ as its *filtration parameter*. One can generate a persistent homology by varying the value of filtration parameter and computing the basis of homology group vector spaces for each simplicial complex corresponding to the filtration parameter value. A graphical representation of $H_t(\mathbf{T})$ based on its homology group vector space basis is called a *barcode diagram*. A barcode diagram helps to recognize the persistent topological features of a topological space. Figure 7.2 illustrates an example of a barcode diagram for a cubical complex. The diagram plots a set of horizontal line segments whose x-axis spans a range of ϖ (filtration parameter values) and whose y-axis depicts the homology group vector space basis in an arbitrary ordering. The number of arrows in the diagram indicates the count of persistent topological features of $\mathbf{T}$.

7.5.2 Separating occupied and unoccupied grid cells with adaptive thresholding

Now we describe our technique of separating occupied grid cells from free ones applying persistent homology [39], a topological data analysis (TDA) technique based on algebraic topology [60], to automatically find a threshold based on an occupancy map's $\mathbb{P}_{m_j^i}$. This TDA-based technique provides an adaptive method for thresholding an occupancy grid map of a domain that contains obstacles at various length scales. In fact, it can also be used with other occupancy grid mapping methods for automated thresholding of the occupancy grid map. In this approach, we threshold $\mathbb{P}_{m_j^i}$ at various levels, compute the numbers of topological features (obstacles) in the domain corresponding to each level of thresholding, and identify the threshold value above which topological features persist. As mentioned in [73], a filtration of cubical complexes based on a parameter called the filtration

parameter is required to compute the persistent homology. In order to be consistent with definition of filtration, $\mathbb{P}_{m_j^i}$ of the unexplored cells are set to unity. We define the filtration parameter ϖ as a threshold for identifying obstacle-free grid cells m_j^i if $\mathbb{P}_{m_j^i} < \varpi$. The methods starts by adding those 2-cubes(square $\square_2$) to a cubical complex whose all four vertices($\square_0$) belong to the center of grid cells with probability of occupancy is less than ϖ. This process is repeated for increasing sequence of ϖ values, which results in a filtration. It is to be noted that a 2-cube is added to the cubical complex only if it is not previously present in the complex. Once a filtration is constructed, a *barcode diagram* is used to identify the number of topological features in the domain, which is given by the number of barcode arrows in each homology group. The threshold ϖ_{cls} for classification of the grid cells is defined as the minimum value of ϖ for which all the topological features are captured by the corresponding cubical complex. In other words, ϖ_{cls} is the value of ϖ for which all the barcode segments except the arrows are annihilated for all values of ϖ greater than this value. This computation is done in practice by taking the maximum value of ϖ that is spanned by the terminating barcode segments in all the homology groups.

The persistent homology computation were done using the C++ program Perseus [92] and barcode diagram plots were generated using MATLAB. Since Perseus accepts only integers as filtration parameter, $\mathbb{P}_{m_j^i}$ values of each map were scaled between 0 and 255 before inputing the data into software. Since the results presented in this paper are only for two-dimensional domains, we restricted the persistent homology computations to dimensions zero and one.



(a) (b) (c) (d) (e)

Figure 7.3: Environments used for simulation. (a) a cave environment (b) a plain environment (c) floor plan of an autonomy lab (d) rough sketch of university of Frieburg (e) floor of a robotics lab in University of Auckland. All these maps were taken from [150]

7.6 SIMULATION RESULTS

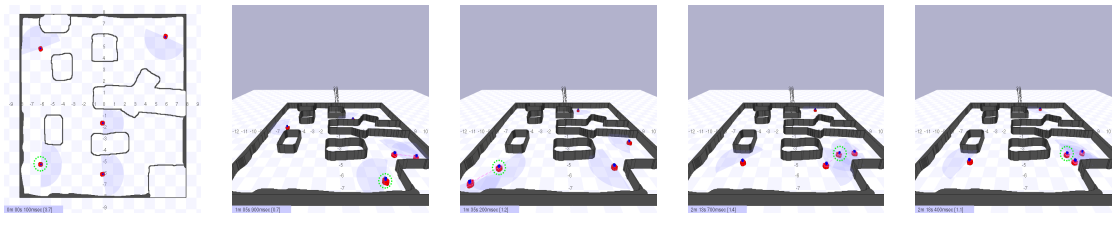
In this section, the proposed approach of information based exploration and distributed mapping technique is assessed in simulation and the results obtained are presented. We implemented the proposed methodology using kinematic robots equipped with on-board laser range sensors on a swarm robot simulator known as Stage simulator [149]. Figure 7.3 depicts various environments used for simulations. The robots were controlled through velocity commands and had maximum speed of 40 cm/s. Also the simulated robots has an on-board laser range sensor with a maximum range of 2 meters. In order for the robots to perform the information correlated Lévy walk in the super diffusive regime, the Lévy exponent α was chosen to be 1.5.

Figure 7.4 shows the post exploration map stored in a robot belonging to the swarm which explored the domains depicted in Figure 7.3. Each map in Figure 7.4 are outcomes of exploring domains of various size, shapes and layouts, as given in the description of Figure 7.4. It very clear from Figure 7.4 that the resulting maps estimate for the environments shown in Figure 7.3 reasonably well.

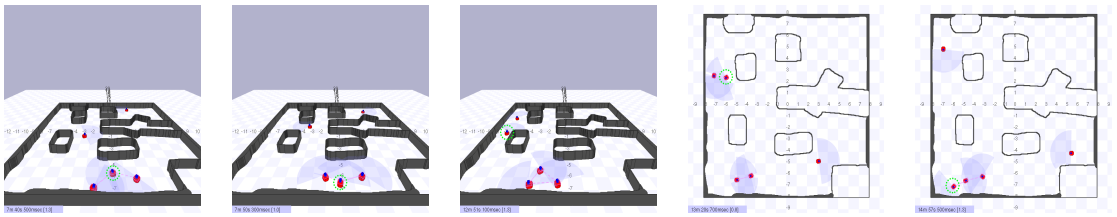
(a) (b) (c) (d) (e)

Figure 7.4: Occupancy grid maps generated by a robot for various environments through information based exploration and our distributed mapping strategy. The black regions indicate a low probability of obstacle occupancy. Gray region indicates the unexplored region. Evidently, the white regions show high probability of obstacle occupancy (a) map generated when 5 robots explored a domain having dimensions $16m \times 16m$ with the layout depicted in Figure 7.3a for $1200s$ (b) map generated when a single robot explored a domain having dimensions $16m \times 16m$ with the layout depicted in Figure 7.3b for $1800s$ (c) map generated when 20 robots explored a domain having dimensions $40m \times 30m$ with the layout depicted in Figure 7.3c for $900s$ (d) map generated when 50 robots explored a domain having dimensions $90m \times 80m$ with the layout depicted in Figure 7.3d for $3600s$ (e) map generated when 10 robots explored a domain having dimensions $40m \times 20m$ with the layout depicted in Figure 7.3e for $1500s$.

Figure 7.5 shows the snapshot of simulation for a cave type environment (Figure 7.3a) at various time instants. Also the occupancy grid map constructed by the robot shown in a green dotted circle at these time instants are displayed in Figure 7.6. The accompanying multimedia attachment shows the video of this simulation with the occupancy grid map of two robots overlaid at the corners of the simulated environment. From the video attachment and Figure 7.6, it is interesting to examine how the robot's occupancy map gets updated when it receive



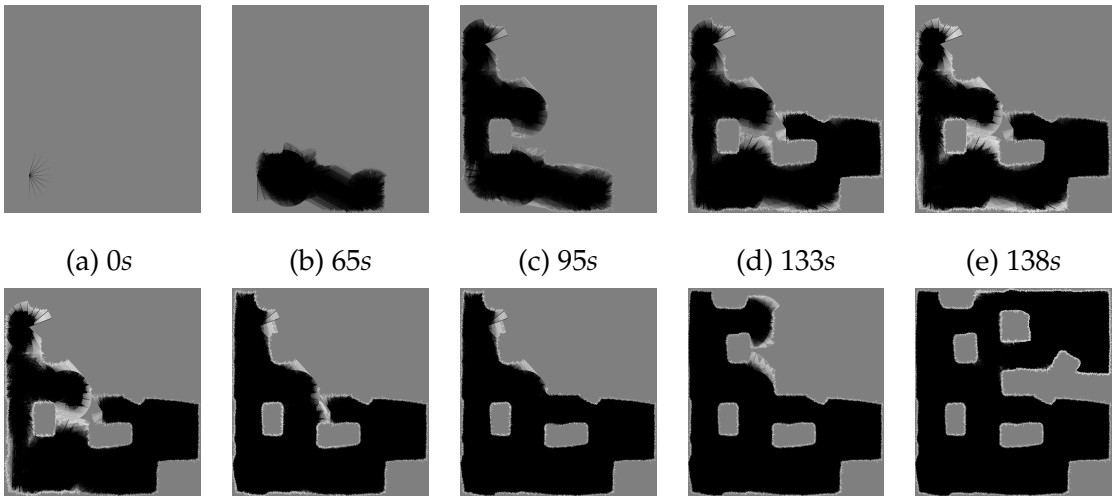
(a) 0*s* (b) 65*s* (c) 95*s* (d) 133*s* (e) 138*s*



(f) 460*s* (g) 470*s* (h) 771*s* (i) 800*s* (j) 897*s*

Figure 7.5: Screenshots of robots exploring the cave shown in Figure 7.3a. Caption under each figure specify the time(in seconds) at which the screenshots were taken. Figure 7.5a, Figure 7.5i and Figure 7.5j gives the top view of the simulation environment. Figure 7.5b to Figure 7.5h portraits a perspective view of the simulation arena. The red dotted lines among robots in the figures indicate that they are in communication range of each other.

new laser range sensor measurements and when it interacts with its neighbors. In order to understand asymptotic consensus on the occupancy grid map, we conducted simulations using a swarm of 50 robots explored an environment of size $90m \times 80m$ having a layout as shown in Figure 7.3d for an hour. Figure 7.11a shows the normalized $2-norm$ of the 2D matrix $\bar{\mathbf{P}}_{M^i}$(rearranged as a 2D matrix) of all 50 robots after the exploration time. The normalization is performed by dividing $2-norm$ of $\bar{\mathbf{P}}_{M^i}$ with the maximum of the $\bar{\mathbf{P}}_{M^i}$ $2-norm$ of values over all robots. The plots show that norms of the occupancy grid maps of all robots are quite close. The occupancy grid maps of some of the robots are shown in Figure 7.7. The effect of size of swarm robots on covering a domain having same size and layout as in



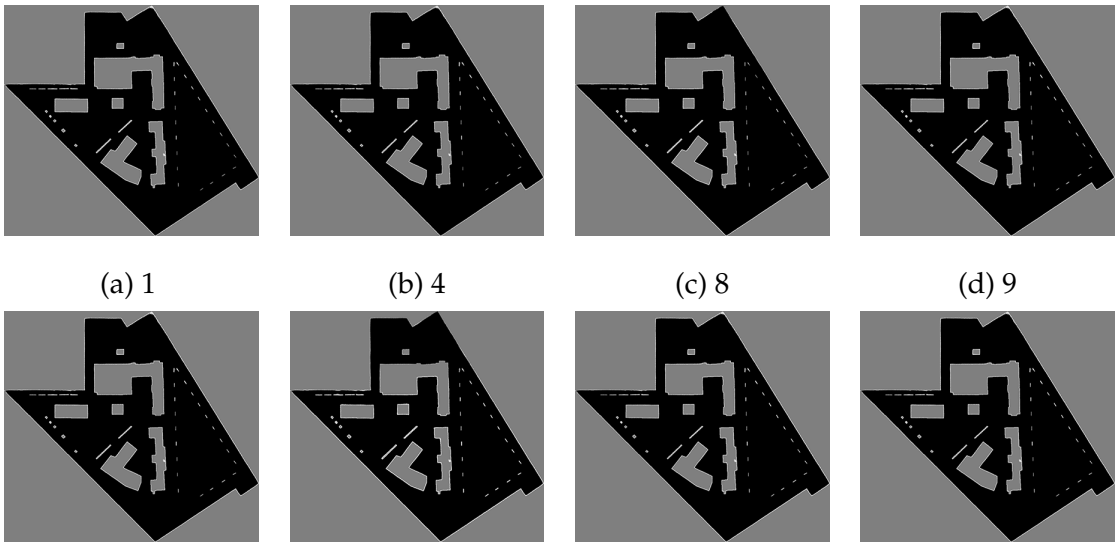
(a) 0s (b) 65s (c) 95s (d) 133s (e) 138s

(f) 460s (g) 470s (h) 771s (i) 800s (j) 897s

Figure 7.6: The occupancy map stored in the robot marked with a green dotted circle in Figure 7.5 at different time instants.

Figure 7.3d is depicted in Figure 7.11b. Robot swarm of sizes $10, 20, 30, 40$ and 50 were deployed for time interval of $3600s$. The experiment was repeated 10 times for each swarm size and the outcomes are portrayed in Figure 7.11b. Y-axis show the percentage coverage of the domain by robots. X-axis represent the number of robots deployed for exploration and mapping. The percentage coverage is calculated as the ratio of amount of area covered by the robot to total area of the domain times 100.The error bars in the plot represent 25^{th} and 75^{th} percentile of the data over the trails. From results shown in Figure 7.11b we find the converge increases with increase in population size.

Figure 7.8 compares the variation of occupancy grid map's entropy[142] over time of exploration, when robots explored the domain through Lévy walk strategy and with our exploration strategy(Section 7.3). The simulated experiments were repeated 10 times. In all the three cases shown in the Figure 7.8, entropy of the occupancy map is reduced faster by our information based exploration strategy



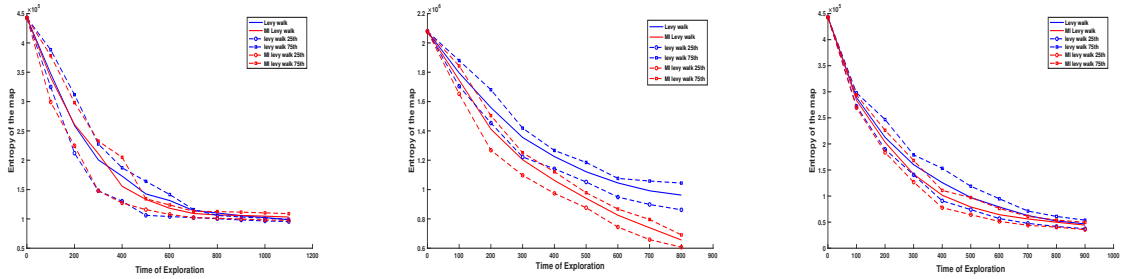
(a) 1 (b) 4 (c) 8 (d) 9

(e) 12 (f) 14 (g) 37 (h) 45

Figure 7.7: Consensus in the occupancy grid maps stored in various robots of a swarm of 50 robots after exploring for an hour. A swarm of fifty robots explored an environment of size $90m \times 80m$ having a layout as shown in Figure 7.3d. The captions of under each figure indicates label of the robot that constructed the map.

when compared to standard Lévy walk based exploration. Similarly, Figure 7.9, delineate the variation for domain converge over time of exploration for the two exploration strategies considered earlier. By careful inspection of Figure 7.9 we find that, our exploration strategy perform better in covering the domain compared to Lévy walk. From Figure 7.8 and Figure 7.9 we find the performance of our strategy is not very significant in case of cave and plain environment. This should be because the domain is small and robots' planning metric could be giving similar value for all its generated paths, forcing them to have a more Lévy walk kind of behavior.

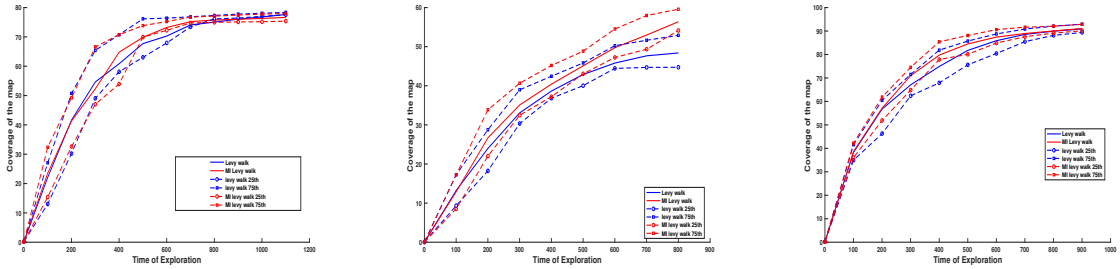
Finally, in order to demonstrate usefulness of the TDA based adaptive thresholding on occupancy grid maps to isolate occupied grid cells from free ones described



(a) cave environment (b) autonomy lab (c) plain environment

Figure 7.8: Comparison of variation of occupancy grid map's entropy[142] over time of exploration, when robots explored the domain using Lévy walk strategy and with our exploration strategy(Section 7.3). The simulated experiments were repeated 30 times. The x-axis of the plots indicates time of exploration. Y-axis represents the entropy of occupancy grid map. Legends of the plots are as follows; solid blue line : mean entropy for Lévy walk strategy, solid red line : mean entropy for our information based exploration strategy(indicated as MI levy walk in plot legends), dash with circle blue line : 25^{th} percentile value of trails in the case of Lévy walk, dash with square blue line : 75^{th} percentile value of trails in the case of Lévy walk, dash with circle red line : 25^{th} percentile value of trails in the case of our strategy(MI levy walk), dash with square blue line : 75^{th} percentile value of trails in the case of our strategy(MI levy walk). (a) Results of the simulations conducted on a cave environment with layout Figure 7.3a and size $16m \times 16m$, using a swarm of 5 robots. (b) Results of the simulations conducted on an autonomy lab with layout Figure 7.3c and size $40m \times 30m$, using a swarm of 20 robots. (c) Results of the simulations conducted on a plain with layout Figure 7.3b and size $16m \times 16m$. , using a swarm of 5 robots.

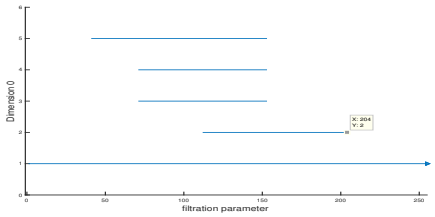
in Section 7.5. We apply the technique on maps shown Figure 7.4a and Figure 7.4c and results are shown in Figure 7.10. Figure 7.10a and Figure 7.10b has one long



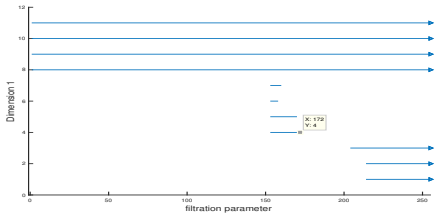
(a) cave environment (b) autonomy lab (c) plain environment

Figure 7.9: Comparison of variation of domain's coverage vs time of exploration, when robots explored the domain using Lévy walk strategy and with our exploration strategy(Section 7.3). The simulated experiments were repeated 30 times. The x-axis of the plots indicates time of exploration. Y-axis represents the percentage coverage of the domain. Legends of the plots are as follows; solid blue line : mean percentage coverage of the domain for Lévy walk strategy, solid red line : mean percentage coverage of the domain for our information based exploration strategy(indicated as MI levy walk in plot legends), dash with circle blue line : 25^{th} percentile value of trails in the case of Lévy walk, dash with square blue line : 75^{th} percentile value of trails in the case of Lévy walk, dash with circle red line : 25^{th} percentile value of trails in the case of our strategy(MI levy walk), dash with square blue line : 75^{th} percentile value of trails in the case of our strategy(MI levy walk). (a) Results of the simulations conducted on a cave environment with layout Figure 7.3a and size $16m \times 16m$, using a swarm of 5 robots. (b) Results of the simulations conducted on an autonomy lab with layout Figure 7.3c and size $40m \times 30m$, using a swarm of 20 robots. (c) Results of the simulations conducted on a plain with layout Figure 7.3b and size $16m \times 16m$. , using a swarm of 5 robots.

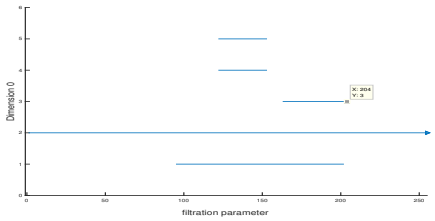
arrow and four long arrows indicating that Figure 7.4a the map of Figure 7.3a has one connected component and four obstacles respectively. Since when comparing



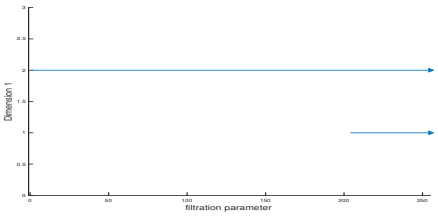
(a) cave(H_0)



(b) cave(H_1)



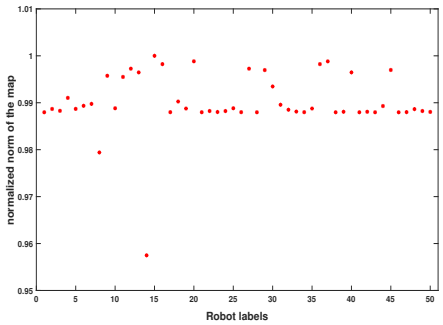
(c) autonomy lab(H_0)



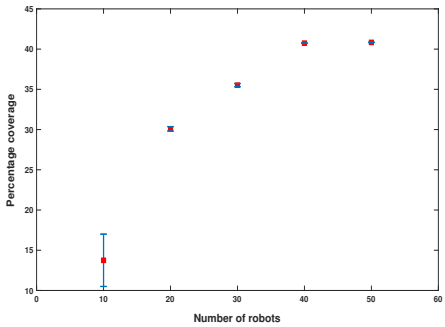
(d) autonomy lab(H_1)

Figure 7.10: Barcodes of the maps generated for two environments. X-axis of the graphs represents filtration parameter (ω), which varies from 0 to 250 representing the pixel intensity of the occupancy grid map. The pixel intensity of m^i is a scaled value of $\mathbb{P}_{m^i_j}$ scaled between 0 and 255. The y-axis shows number of Homology generators for dimensions zero(Figure 7.10a & Figure 7.10c) and one(Figure 7.10b & Figure 7.10d). Figure 7.10a and Figure 7.10b are barcode diagrams generated for map shown in Figure 7.4a. Comparing the termination points of the line segments(non arrows) in Figure 7.10a and Figure 7.10b, we find that grid cells with pixel intensity greater than 204 should occupied. Figure 7.10c and Figure 7.10d are barcode diagrams generated for map shown in Figure 7.4c. Again comparing the termination points of the line segments(non arrows) in Figure 7.10c and Figure 7.10d, we find that grid cells with pixel intensity greater than 204 should occupied.

Figure 7.10a and Figure 7.10b, we find that the maximum filtration value for which all the line segments(non-arrow) representing non persistent topological features



(a) Consensus over maps



(b) Coverage vs Swarm size

Figure 7.11: (a) Consensus over the occupancy grid map constructor by 50 robots which explored an environment of size $90m \times 80m$ having a layout as shown in Figure 7.3d, for a time period of $3600s$. Each point in the plot represent the normalized $2-norm$ of the 2D matrix $\bar{\mathbf{P}}_{M^i}$(rearranged as a 2D matrix) of robot R^i. The normalization is performed by dividing $2-norm$ of $\bar{\mathbf{P}}_{M^i}$ with the maximum of the $\bar{\mathbf{P}}_{M^i}$ $2-norm$ of values over all robots. The plots show that norms of the occupancy grid maps of all robots are quite close. The occupancy grid maps of some of the robots are shown in Figure 7.7. (b) The effect of size of swarm robots on covering a domain having same size and layout as in Figure 7.11a. Robot swarm of sizes $10, 20, 30, 40$ and 50 were deployed for time interval of $3600s$. The experiment was repeated 10 times for each swarm size. Y-axis show the percentage coverage of the domain by robots. X-axis represent the number of robots deployed for exploration and mapping. The percentage coverage is calculated as the ratio of amount of area covered by the robot to total area of the domain times 100.The error bars in the plot represent 25^{th} and 75^{th} percentile of the data over the trails.

terminate is 204. Therefore, any grid cell whose probability of occupancy is greater than or equal to 0.8 is occupied. We arrive at a similar conclusion while examining Figure 7.10c and Figure 7.10d.

Figure 7.12: The experimental arena with obstacles. To minimize early robot interaction, the robots were placed in specific starting locations. For consistency in testing, the same positions were used for each trial.

7.7 EXPERIMENTS

To validate the theory and simulations established in the previous sections, a physical experiment using a group of Lidar equipped, differential-drive robots was formulated. We will expand upon the equipment, obstacle setup, assumptions, and results used in the following sections.

7.7.1 Experimental Setup

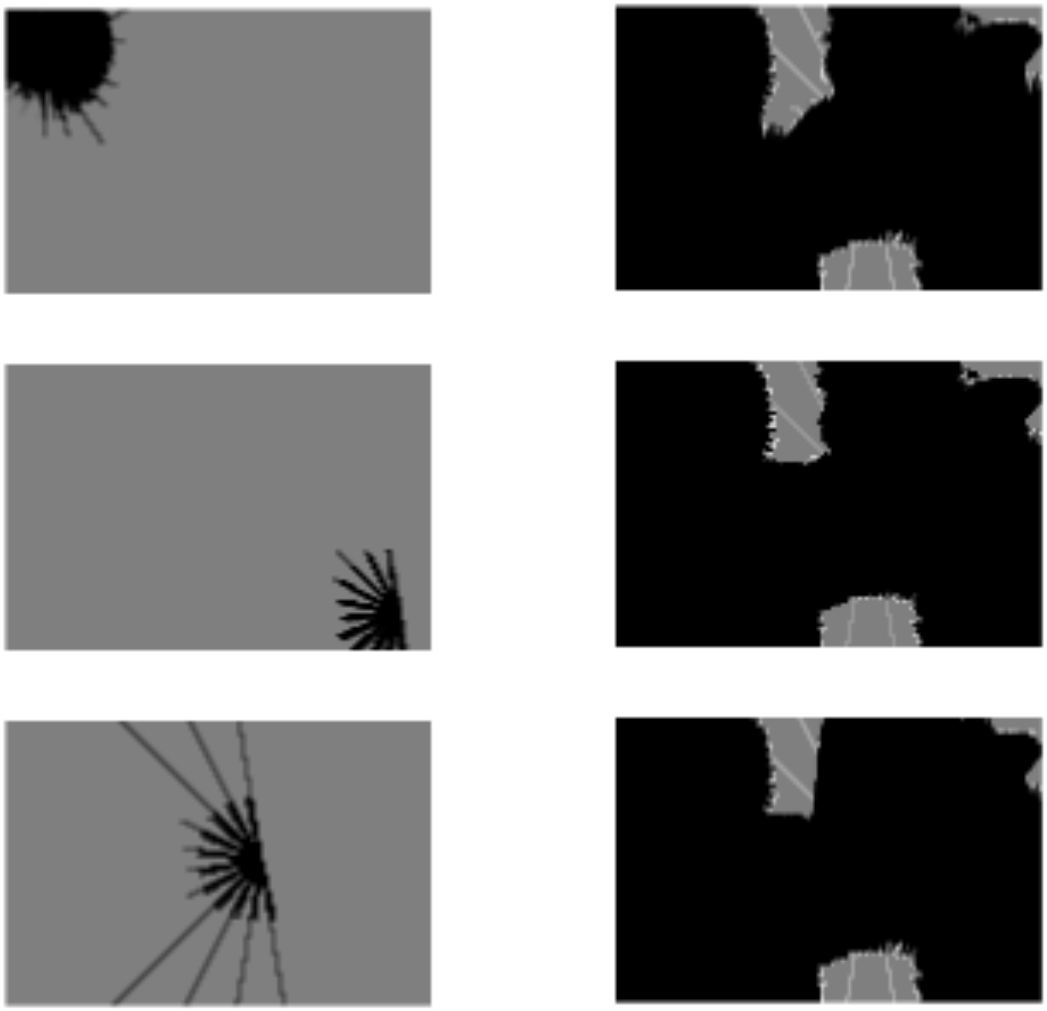
For the experiment, a 2.6 m x 1.6 m arena filled with obstacles of varying size, shown in Figure 7.12, was created. The height of each obstacle was consistent with

the bordering walls of the arena. To monitor the robots, a Microsoft LifeCam was mounted to the ceiling of the laboratory and connected to a central computer. It should be noted that the size of the arena was restricted due to the ceiling height on which the overhead camera was mounted and having the camera option restrictions arising from the OS on the central computer (Ubuntu).

Three *Turtlebot 3 Burger* robots were used to conduct the experiment. The robot is equipped with a *Raspberry Pi 3* for higher-level computation, an *OpenCR* controller board containing a microcontroller and numerous sensors, and a 360° Lidar. The robot maneuvers by means of two *Dynamixal XL430-W250* actuators in a differential-drive configuration. For the higher level control, each robot runs the Kinetic Kame version of the Robot Operating System (ROS) middleware on Ubuntu Mate 16.04 installed on their respective Raspberry Pi 3.

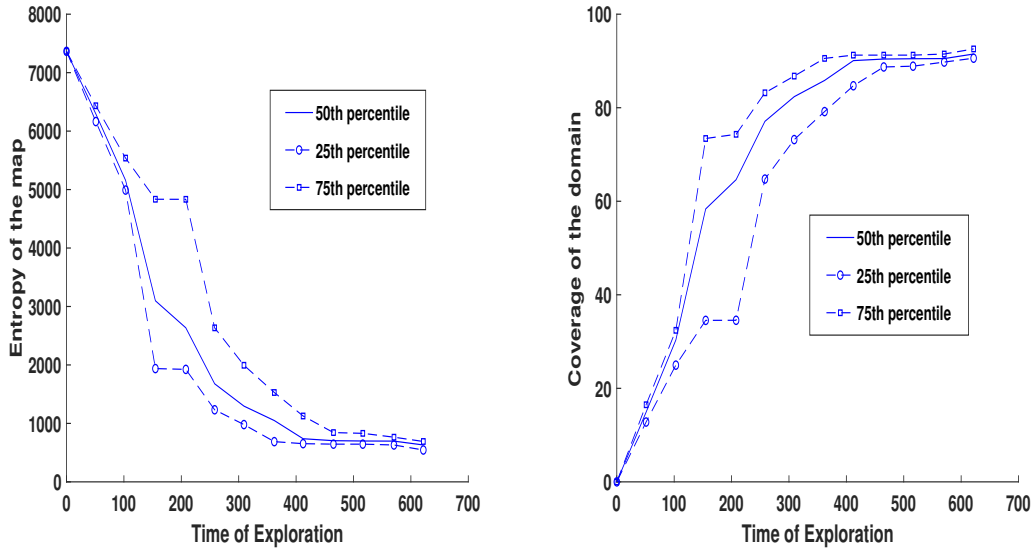
The code used in simulation was modified to work in each robot and for other experimental conditions presented by the arena. For example, the small arena space required each robot's Lidar range to be limited to 40 *cm* in front of the robotic agent instead of the default 2 *m* range. The sensor range of the Lidar was limited to 180° instead of the full 360°. A 3D printed mount was created to easily place an *ArUco* fiducial marker on top of the robot without disrupting the Lidar sensor readings.

Each robot was placed in different locations in the arena to prevent interactions and information exchange at the beginning of the experiment. Figure 7.12 shows the starting locations of each agent in the arena. The robots may move anywhere in the arena and will avoid obstacles that are within their 180° sensing range.



(a) Iteration 1 (b) Iteration 20

Figure 7.13: Generated maps by each robot (top to bottom: robot 1, robot 2, and robot 3) over the course of a 10 minute trial. Every 30 seconds a map is saved along. (a) are the first maps saved by the robots, and (b) are the last.



(a) Entropy of map

(b) Coverage of domain

Figure 7.14: Experimental results of the data taken from a robot after ten trails. (a) The x-axis of the plot indicate time of exploration. Y-axis represent the entropy of occupancy grid map. The solid blue line indicate the 50^{th} percentile of the data at various times. dash with circle blue line and dash with square blue line represent the 25^{th} and 75^{th} percentile value of trails respectively. (b) The x-axis of the plot indicate time of exploration. Y-axis represent the percentage coverage of the domain. The solid blue line indicate the 50^{th} percentile of the data at various times. dash with circle blue line and dash with square blue line represent the 25^{th} and 75^{th} percentile value of trails respectively.

7.7.2 Software Architecture

By using ROS, we were able to distribute much of the computing necessary for the experiment. To do this, ROS utilizes the concept of *Nodes*, which contain code for computing certain tasks. The information generated from these nodes can then be sent to other nodes or can be self-contained. For our experiment, we distributed

nodes across the multiple robots and in our central computer to better demonstrate a distributed algorithm.

The central computer acted as a *ROS Master*, which connects nodes distributed across multiple machines, and contained nodes pertaining to information acquisition from the overhead camera used to monitor the robots and to disseminate that information to the robots. The ArUco markers on the robots are used to calculate both position and orientation by the overhead camera node. Each robot requires this information when building their maps and to identify neighbors who are in their vicinity for sharing maps.

Each Turtlebot 3 Burger contained two nodes: one to send-receive information sent to-and-from the OpenCR controller board, and another that provided high-level control pertaining to the algorithm. All three agents communicate with the central computer using the on-board *WiFi* modules located on the Raspberry Pi 3.

7.7.3 Experimental Results

A total of 10 experiment were run lasting 10 minutes each. The same robot configuration was used for every test to keep the data generated consistent. After every test, the maps generated every 30 seconds and entropy and coverage data were collected from each robot for analysis. Figure 7.14a and Figure 7.14b shows the variation of entropy of the occupancy map stored in a robot and percentage coverage of the domain shown in Figure 7.12 over time of exploration respectively.

Chapter 8

SCALAR FIELD ESTIMATION BY LARGE SENSOR NETWORKS WITH PARTIALLY ACCESSIBLE MEASUREMENTS

Source: **Ragesh K. Ramachandran** and Berman [130]

Funding: NSF Awards CMMI-1363499

ABSTRACT

This chapter studies the problem of reconstructing a two-dimensional scalar field using measurements from a subset of a network with local communication between nodes. The problem considered in this chapter is the communication network of the nodes to form either a chain or a grid topology. The reconstruction problem is formulated as an optimization problem that is constrained by first-order linear dynamics on a large interconnected system. To solve this problem, an optimization-based scheme is employed that uses a gradient-based method with an analytical computation of the gradient. The main contribution of the chapter is a derivation of bounds on the trace of the observability Gramian of the system, which can be used to quantify and compare the field estimation capabilities of chain and grid networks. A comparison based on a performance measure related to the $\mathcal{H}_2$ norm of the system is also used to study the robustness of the network topologies. The results are validated in simulation using both Gaussian scalar fields and actual ocean salinity data.

The work in this chapter was first presented in [130]. In this work [130], we investigate the effect of network topology on the accuracy and robustness of a method that we devise for reconstructing a static scalar field from partial observations. We note that our method can also be adapted to estimate a time-varying scalar field whose dynamics are slower than the network information dynamics. This method uses temporal data collected by the accessible nodes in the network to estimate the initial measurements of the field that were obtained by the full set of nodes. The nodes share their measurements with their neighbors through a fixed communication network. The network is assigned either a grid or chain topology, which are common candidates for approximating 1D and 2D domains in practical applications. We specify that the information flow in the network is governed by a first-order linear dynamical model. This simple model of information dynamics represents the case where no data is stored in the nodes and a single item of information is transmitted between nodes at a time. In addition, this model yields diffusive information dynamics that eventually approach a steady state, which allows us to determine a time at which the data values at the nodes have largely stabilized and thus gives a fixed time interval over which the data can be retrieved. From a control theory perspective, the estimation problem addressed by our method is equivalent to finding the initial condition of a linear dynamical system given its inputs and outputs. The solution to this problem is associated with the *observability* of the system.

We adopt a quantitative measure of observability, based on the trace of the observability Gramian, that is similar to [102, 101, 157, 45, 90], departing from the graph-theoretic methods used in [69, 83, 97, 100]. Our analysis makes use of necessary and sufficient conditions for the observability spectral properties of chain and grid networks, which are well-understood [97, 41]. In the main result

of this paper, we derive bounds on the trace of the observability Gramian of an undirected network and use these bounds to compare the estimation performance of networks with either grid or chain topologies. We evaluate this performance for our novel method of estimating the initial condition of a large network with linear dynamics, which constitutes another contribution of this paper. We use an optimization framework to address this estimation problem and derive the gradient required to solve it. A third contribution of the paper is our characterization of a network's robustness to noise using a performance measure based on the $\mathcal{H}_2$ norm of the system. We find that even with simple first-order information dynamics, the topology of the network significantly affects its estimation performance and its robustness to noise. We illustrate our approach on both simulated and actual two-dimensional scalar fields.

8.1 Related Work

Although there is a great deal of literature on optimal control, little work has addressed the optimal estimation of initial conditions other than through the inversion of the observability Gramian [74]. In general, the observability of a linear dynamical system can be verified by using the Kalman rank condition [62]. However, checking the rank condition for large interconnected systems is computationally intensive due to the high dimensionality of the observability Gramian. For this reason, a less computationally intensive graph-theoretic characterization of observability has been more widely used than a matrix-theoretic characterization for large complex networked systems. The observability of complex networks is studied in [83] using a graph-based approach, which presents a general result that holds true for most of the chosen network parameters (the edge weights). In [69], a graph-theoretic approach based on equitable partitions of graphs is used to

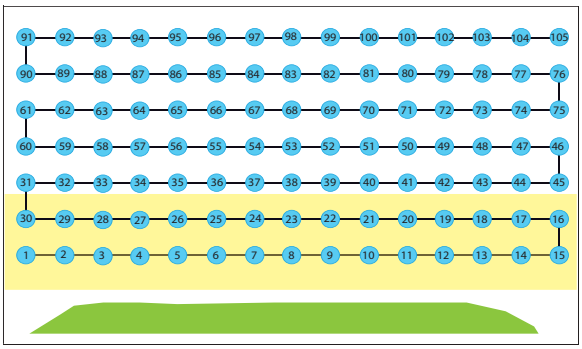
derive necessary conditions for observability of networks. Alternately, [158] uses a matrix-theoretic approach to develop a maximum multiplicity theory to characterize the exact controllability of a network in terms of the minimum number of required independent controller nodes based on the network spectrum.

8.2 Mathematical Preliminaries

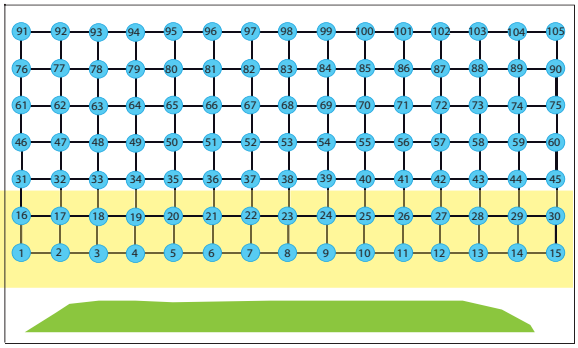
A graph $\mathcal{G}$ can be defined as the tuple $(V(\mathcal{G}), E(\mathcal{G}))$, where $V(\mathcal{G})$ is a set of N vertices, or *nodes*, and $E(\mathcal{G}) = \{(i,j) : i \neq j,\ i,j \in V(\mathcal{G})\}$ is a set of M *edges*. Nodes i and j are called *neighbors* if $(i,j) \in E(\mathcal{G})$. The set of neighbors of node i is denoted by $\mathcal{N}_i = \{j : j \in V(\mathcal{G}), (i,j) \in E(\mathcal{G})\}$. The *degree* d_i of a node i is defined as $|\mathcal{N}_i|$. We assume that $\mathcal{G}$ is finite, simple, and connected unless mentioned otherwise.

A graph $\mathcal{G}$ is associated with several matrices whose spectral properties will be used to derive our results. The *incidence matrix* of a graph with arbitrary orientation is defined as $\mathbf{B}(\mathcal{G}) = [b_{ij}] \in \mathbb{R}^{N \times M}$, where the entry $b_{ij} = 1$ if i is the initial node of some edge j of $\mathcal{G}$, $b_{ij} = -1$ if i is the terminal node of some edge j of $\mathcal{G}$, and $b_{ij} = 0$ otherwise. It can be shown that the left nullspace of $\mathbf{B}(\mathcal{G})$ is $c\mathbf{1}_N$, $c \in \mathbb{R}$, where $\mathbf{1}_N$ is the $N \times 1$ vector of ones [56]. The *degree matrix* $\Delta(\mathcal{G})$ of a graph is given by $\Delta(\mathcal{G}) = Diag(d_1, ..., d_N)$. The *adjacency matrix* $\mathbf{A}(\mathcal{G}) = [a_{ij}] \in \mathbb{R}^{N \times N}$ has entries $a_{ij} = 1$ when $(i,j) \in E(\mathcal{G})$ and $a_{ij} = 0$ otherwise. The *graph Laplacian* can be defined from these two matrices as $\mathbf{L}(\mathcal{G}) = \Delta(\mathcal{G}) - \mathbf{A}(\mathcal{G})$. The Laplacian of an undirected graph is symmetric and positive semidefinite, which implies that it has real nonnegative eigenvalues $\lambda_i(\mathcal{G})$, $i = 1, ...n$. The eigenvalues can be ordered as $\lambda_1(\mathcal{G}) \leq \lambda_2(\mathcal{G}) \leq ... \leq \lambda_N(\mathcal{G})$, where $\lambda_1(\mathcal{G}) = 0$. The eigenvector corresponding to eigenvalue $\lambda_1(\mathcal{G})$ can be computed to be $\mathbf{1}_N$. By Theorem 2.8 of [86], the graph is connected if and only if $\lambda_2(\mathcal{G}) > 0$.

Several other matrices will be defined as follows. An $n_1 \times n_2$ identity matrix will



(a) Chain topology



(b) Grid topology

Figure 8.1: Illustration of the chain and grid network topologies. The blue circles are nodes and are labeled by numbers. Nodes in the yellow region are accessible nodes.

be denoted by $\mathbf{I}_{n_1 \times n_2}$, and an $n_1 \times n_2$ matrix of zeros will be denoted by $\mathbf{0}_{n_1 \times n_2}$. The matrix $\mathbf{J}_N$ is defined as $\mathbf{J}_N = \mathbf{1}_N \mathbf{1}_N^T$.

8.3 Problem Statement

Consider a set of N nodes with local communication ranges and local sensing capabilities. The nodes are arranged in a bounded domain as shown in figure 8.1. Each node is capable of measuring the value of a scalar field at its location and communicating this value to its neighbors, which are defined as the nodes that

are within its communication range. The nodes take measurements at some initial time and transmit this information using a nearest-neighbor averaging rule, which is described in Section 8.4. As shown in figure 8.1, we assume to have direct access only to the measurements of a small subset of the nodes, which we call the *accessible nodes*, which for instance may be closer to a particular boundary of the domain. We also assume that the node positions are predetermined and that the nodes employ feedback mechanisms to regulate their positions in the presence of external disturbances.

We address the problem of reconstructing the initial measurements taken by all the nodes from the measurements of the accessible nodes. This can be formulated as the problem of determining whether the information flow dynamics in the network are observable with respect to a set of given outputs. As discussed in the beginning of this chapter, we restrict our investigation to chain and grid communication topologies, whose structural observability properties are well-studied [97, 100]. We will focus on comparing the chain and grid topologies in terms of their utility as communication networks to be used in reconstructing an initial set of data.

8.4 Network Model

The communication network among the N nodes is represented by an undirected graph $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$, where vertex $i \in V(\mathcal{G})$ denotes node i, and nodes i and j can communicate with each other if $(i, j) \in E(\mathcal{G})$. Let $x_i(t) \in \mathbb{R}$ be a scalar data value obtained by node i at time t. We define the information flow dynamics of node i as

$$\frac{dx_i}{dt} = \sum_{(i,j)\in\mathcal{N}_i} (x_j - x_i). \tag{8.1}$$

The vector of all nodes' information at time t is denoted by $\mathbf{X}(t) = [x_1(t)\ x_2(t)\ ...\ x_N(t)]^T$. Using Equation 8.1 to define the dynamics of $x_i(t)$ for each node i, we can

define the information flow dynamics over the entire network as

$$\begin{aligned} \dot{\mathbf{X}}(t) &= -\mathbf{L}(\mathcal{G})\mathbf{X}(t), \\ \mathbf{X}(0) &= \mathbf{X}_0, \end{aligned} \tag{8.2}$$

where $\mathbf{X}_0 \in \mathbb{R}^N$ contains the unknown initial values of the data obtained by the nodes at time $t = 0$, which is the information that we want to estimate.

We define $Id = \{I_1, I_2, ..., I_k\} \subseteq V(\mathcal{G})$ as the index set of the *accessible nodes*. The output equation for the linear system Equation 8.2 is given by

$$\mathbf{Y}(t) = \mathbf{C}\mathbf{X}(t), \tag{8.3}$$

where $\mathbf{Y}(t) \in \mathbb{R}^k$ and $\mathbf{C} = [c_{ij}] \in \mathbb{R}^{k\times N}$ is a sparse matrix whose entries are defined as $c_{ij} = 1$ if $i = j$ and $i \in Id$, $c_{ij} = 0$ otherwise. If we number the nodes in such a way that the first k output nodes are ordered from 1 to k, then $\mathbf{C} = \left[\mathbf{I}_{k\times k}\ \mathbf{0}_{k\times(N-k)}\right]$.

As previously discussed, we focus on the case where the network has a chain or grid communication topology. The type of topology affects the network dynamics through its associated graph Laplacian $\mathbf{L}(\mathcal{G})$. Let $\mathcal{G}_c$ and $\mathcal{G}_g$ represent communication networks with a chain topology and a grid topology, respectively. When the nodes in each network are labeled as shown in figure 8.1a and figure 8.1b, then it can be shown that $\mathbf{L}(\mathcal{G}_c)$ and $\mathbf{L}(\mathcal{G}_g)$ [41] have the following structures:

$$\mathbf{L}(\mathcal{G}_c) = \begin{bmatrix} 1 & -1 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ -1 & 2 & -1 & \ddots & & & & \vdots \\ 0 & -1 & 2 & -1 & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & -1 & 2 & -1 & 0 \\ \vdots & & & & \ddots & -1 & 2 & -1 \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & -1 & 1 \end{bmatrix} \tag{8.4}$$

and

$$\mathbf{L}(\mathcal{G}_g) = \begin{bmatrix} \mathbf{D}_1 & -\mathbf{I} & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ -\mathbf{I} & \mathbf{D}_2 & -\mathbf{I} & \ddots & & & & \vdots \\ 0 & -\mathbf{I} & \mathbf{D}_2 & -\mathbf{I} & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & -\mathbf{I} & \mathbf{D}_2 & -\mathbf{I} & 0 \\ \vdots & & & & \ddots & -\mathbf{I} & \mathbf{D}_2 & -\mathbf{I} \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & -\mathbf{I} & \mathbf{D}_1 \end{bmatrix}, \tag{8.5}$$

where

$$\mathbf{D}_1 = \begin{bmatrix} 2 & -1 & \cdots & \cdots & 0 \\ -1 & 3 & -1 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & -1 & 3 & -1 \\ 0 & \cdots & \cdots & -1 & 2 \end{bmatrix},$$

$$\mathbf{D}_2 = \begin{bmatrix} 3 & -1 & \cdots & \cdots & 0 \\ -1 & 4 & -1 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & -1 & 4 & -1 \\ 0 & \cdots & \cdots & -1 & 3 \end{bmatrix}.$$

Here, $\mathbf{L}(\mathcal{G}_g)$ is a $(l_1 l_2) \times (l_1 l_2)$ matrix and $\mathbf{D}_1, \mathbf{D}_2$ are both $l_1 \times l_1$ matrices, with $l_1 l_2 = N$. Without loss of generality, we assume that the grid is square, meaning that $l_1 = l_2 = l$. We direct the reader to [10] for a numerical example of $L(\mathcal{G}_g)$.

The graph Laplacians $\mathbf{L}(\mathcal{G}_c)$ and $\mathbf{L}(\mathcal{G}_g)$ are constructed based on the numbering of the vertex sets $V(\mathcal{G}_c)$ and $V(\mathcal{G}_g)$ that is shown in figure 8.1. Graphs that are constructed by reordering the vertices of the graphs shown in figure 8.1 are isomorphic

to the graphs in the figure. Isomorphic graphs are also isospectral [153].

Since the system Equation 8.2 is linear, its solution is [62]

$$\mathbf{X}(t) = \mathbf{e}^{-\mathbf{L}(\mathcal{G})t}\mathbf{X}_0. \tag{8.6}$$

By combining Equation 8.3 and Equation 8.6, we obtain the map between the unknown initial data $\mathbf{X}_0$ and the measured output $\mathbf{Y}(t)$ as

$$\mathbf{Y}(t) = \mathbf{C}\mathbf{e}^{-\mathbf{L}(\mathcal{G})t}\mathbf{X}_0. \tag{8.7}$$

8.5 Scalar Field Reconstruction

The problem of scalar field reconstruction can now be framed as an inversion of the map given by Equation 8.7. From linear systems theory, the property of *observability* refers to the ability to determine an initial state $\mathbf{X}_0$ from the inputs and outputs of a linear dynamical system [62]. For systems defined by Equation 8.2 with an associated chain or grid topology, the conditions for observability are well-studied [97]. This ensures that the reconstruction problem can be solved for the types of networks that we consider.

We solve the scalar field reconstruction problem by posing it as an optimization problem. The optimization procedure uses observed data $\hat{\mathbf{Y}}(t)$ from the accessible nodes over the time interval $t \in [0\ T]$ to recover $\mathbf{X}_0$. The goal of the optimization routine is to find the state $\mathbf{X}_0$ that minimizes the normed distance between this observed data, $\hat{\mathbf{Y}}(t)$, and the output $\mathbf{Y}(t)$ computed using Equation 8.7. Therefore, we can frame our optimization objective as the computation of $\mathbf{X}_0$ that minimizes the functional $J(\mathbf{X}_0)$, defined as

$$J(\mathbf{X}_0) = \frac{1}{2}\int_0^T \left\|\mathbf{Y}(t) - \hat{\mathbf{Y}}(t)\right\|_2^2 dt + \frac{\lambda}{2}\left\|\mathbf{X}_0\right\|^2, \tag{8.8}$$

subject to the constraint given by Equation 8.7. Here, λ is the Tikhonov regularization parameter, which is added to the objective function to prevent $\mathbf{X}_0$ from

becoming large due to noise in the data [17].

The convexity of $J(\mathbf{X}_0)$ ensures the convergence of gradient descent methods to its global minima. We use one such method to compute the $\mathbf{X}_0$ that minimizes this functional. The method requires us to compute the gradient of $J(\mathbf{X}_0)$ with respect to $\mathbf{X}_0$. This is done by combining Equation 8.7 and Equation 8.8, then taking the Fréchet derivative of the resulting expression with respect to $\mathbf{X}_0$ [84]. Defining $\Psi(t) = \mathbf{C}\mathbf{e}^{-\mathbf{L}(\mathcal{G})t}$, the gradient of $J(\mathbf{X}_0)$ can be computed in this way as:

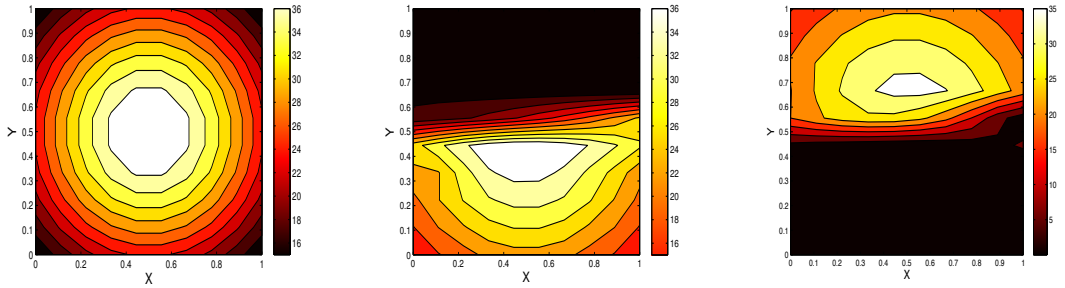
$$\delta J(\mathbf{X}_0) = \int_0^T (\Psi(t))^* \left(\Psi(t)\mathbf{X}_0 - \hat{\mathbf{Y}}(t)\right) dt + \lambda \mathbf{X}_0, \tag{8.9}$$

where $(\Psi(t))^*$ is the Hermitian adjoint of $\Psi(t)$, which in this case is simply the Hermitian transpose [84].

The most computationally intensive part of calculating Equation 8.9 is computing the matrix exponential in $\Psi(t)$. There has been a great deal of literature about approximate computation of the matrix exponential [64, 99], which by definition is an infinite matrix series. In general, finding the matrix exponential is a computationally hard problem for very large matrices and computing it can be error-prone if not done carefully, especially if spectral decomposition [127] of the matrix is not possible [88]. We can calculate the gradient by noting that $\mathbf{Y}(t) = \Psi(t)\mathbf{X}_0$ by Equation 8.7, applying a change of variables $\tau = T - t$ to the integral term in Equation 8.9, and defining $\hat{u}(\tau) \equiv \mathbf{Y}(T-\tau) - \hat{\mathbf{Y}}(T-\tau)$:

$$\begin{aligned}
&\int_0^T (\Psi(t))^* \left(\Psi(t)\mathbf{X}_0 - \hat{\mathbf{Y}}(t)\right) dt \\
&\quad = \int_0^T (\Psi(T-\tau))^* \left(\mathbf{Y}(T-\tau) - \hat{\mathbf{Y}}(T-\tau)\right) d\tau \\
&\quad = \int_0^T \mathbf{e}^{-\mathbf{L}^*(\mathcal{G})(T-\tau)} \mathbf{C}^* \hat{u}(\tau) d\tau.
\end{aligned}$$

This expression can be thought of as the solution $P(\tau)$ of the following differential



(a) Actual field (b) Estimated field (c) Absolute value of error

Figure 8.2: Gaussian function estimation using 100 nodes with a chain communication topology. Temporal data is acquired from 30 nodes over a time period of 50 sec.

equation at time $\tau = T$ [62]:

$$\frac{dP}{d\tau} = -\mathbf{L}^*(\mathcal{G})P(\tau) + \mathbf{C}^*\hat{u}(\tau), \quad P(0) = 0. \tag{8.10}$$

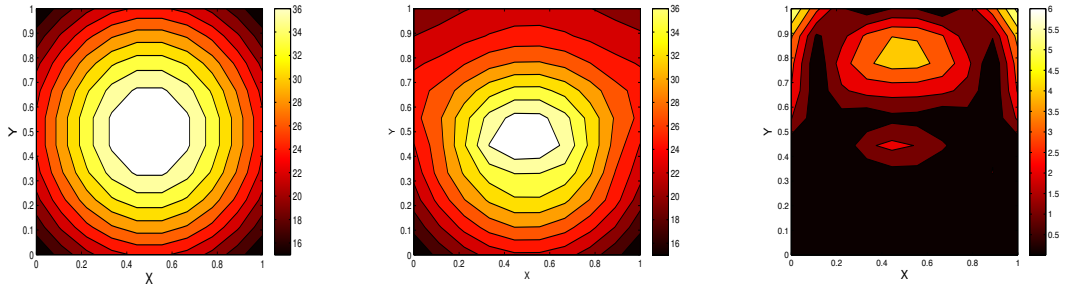
Using this result, the gradient Equation 8.9 can be written as

$$\delta J(\mathbf{X}_0) = P(T) + \lambda \mathbf{X}_0. \tag{8.11}$$

To compute the gradient, we can solve Equation 8.10 forward to find $P(T)$.

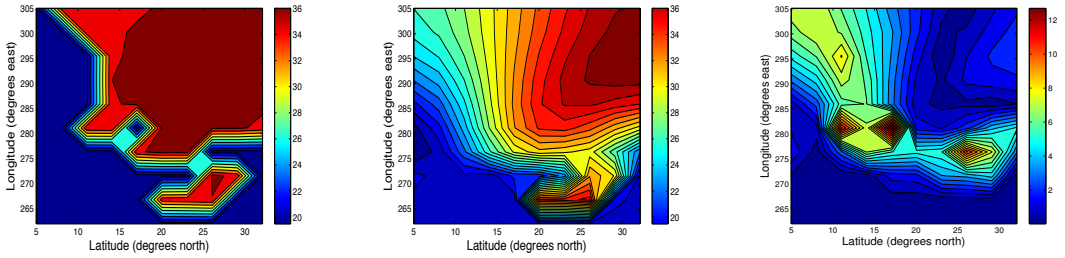
8.6 Simulations

We applied the method described in Section 8.5 to reconstruct a Gaussian scalar field using 100 nodes, whose communication network either has a chain topology or a grid topology. The simulations were performed on a normalized domain of size 1 m $\times$ 1 m. The field was reconstructed using data collected over a time period of 50 sec from a set of 30 accessible nodes. Figure 8.2 and Figure 8.3 illustrate the results from using the chain and grid topologies, respectively. Each figure shows the contour plots of the actual field, the estimated field, and absolute value of the



(a) Actual field (b) Estimated field (c) Absolute value of error

Figure 8.3: Gaussian function estimation using 100 nodes with a grid communication topology. Temporal data is acquired from 30 nodes over a time period of 50 sec.



(a) Actual field (b) Estimated field (c) Absolute value of error

Figure 8.4: Estimation of salinity (psu) over a section of the Atlantic Ocean at a depth of 25 m. The network consists of 100 nodes with a grid communication topology. Temporal data is acquired from 30 nodes over a time period of 50 sec.

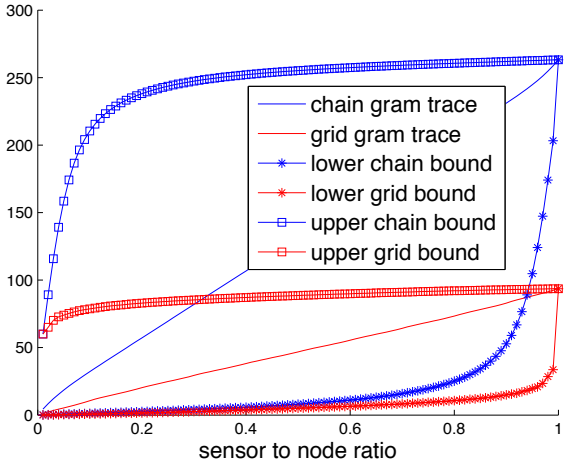
error between these plots. From these plots, it is evident that the grid topology yields a much more accurate reconstruction of the field than the chain topology, even though both networks can be characterized as observable.

In order to test the performance of our method in a practical scenario, we applied it in simulation to a set of real salinity data (psu), obtained from [93], over a section of the Atlantic Ocean at a depth of 25 m. The salinity field was reconstructed

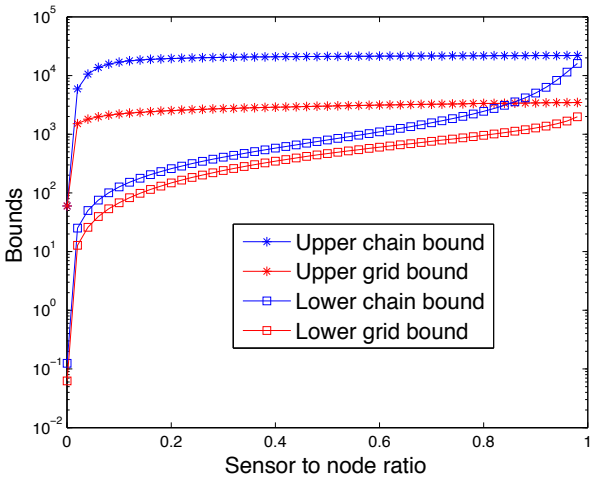
over a time period of 50 sec using 100 nodes with a grid communication topology and 30 accessible nodes whose temporal data were sampled at 10 Hz. During the simulation, each node measured the salinity at its position and transmitted this information to its neighboring nodes according to Equation 8.2. The temporal observations by the accessible nodes, obtained over 50 sec according to Equation 8.3, were used to reconstruct the salinity measurements taken by all the nodes using the techniques described in Section 8.5. The contour plots in Figure 8.4 show that the estimated salinity field reproduces the key features of the actual field with reasonable accuracy.

8.7 Effect of Network Topology on Estimation Performance

In this section, we analyze the effect of network topology on the accuracy of the field estimation as the number of nodes in the network increases. Comparing the results in Figure 8.2 and Figure 8.3, it is evident that there is some fundamental limitation arising from the network structure which makes the system with the chain topology practically unobservable. In the control theory literature, the *degree of observability* is used as a metric of a system's observability [90]. The *observability Gramian* $W_O(0, T)$ can be used to compute the initial state of an observable linear system from output data over time $t \in [0\ T]$ [62]. This makes it a good candidate for use in quantifying the relative observability among different systems. Due to the duality of controllability and observability, the results associated with one of these properties can be used for the other if interpreted properly. Commonly used measures of the degree of observability (controllability) are the smallest eigenvalue, the trace, the determinant, and the condition number of the observability (controllability) Gramian [101, 157, 45]. For large, sparse networked systems, the Gramian can be highly ill-conditioned, which makes numerical computation of its



(a) Networks with 100 nodes



(b) Networks with 10000 nodes

Figure 8.5: Comparison of the degree of observability based on the trace of the observability Gramian and its bounds. The trace shown in Figure 8.5a is computed numerically using the eigenvectors of $\mathbf{L}(\mathcal{G})$.

minimum eigenvalue unstable. Although researchers have computed bounds on the minimum eigenvalues of the Gramian [102], these bounds did not help to us arrive at a conclusion since they were too close together.

These factors prompted us to use the trace of the observability Gramian as our metric for the degree of observability. Analogous to the interpretation of the controllability Gramian in [102], the trace of the observability Gramian can be interpreted as the average sensing effort applied by a system to estimate its initial state. For a communication network represented by $\mathcal{G}$ with information flow dynamics given by Equation 8.2, the trace of the observability Gramian $W_O(0,T)$ is defined as

$$Trace(W_O(0,T)) = Trace\left(\int_0^T \mathbf{e}^{-\mathbf{L}(\mathcal{G})^* t}\mathbf{C}^*\mathbf{C}\mathbf{e}^{-\mathbf{L}(\mathcal{G})t}dt\right). \tag{8.12}$$

Following steps similar to those in [102], we use Theorem 8 below to derive upper and lower bounds on the trace of the observability Gramian for networks with chain and grid topologies. Figure 8.5 compares these lower and upper bounds for two network sizes as a function of the sensor-to-total-node ratio, where the *sensors* are defined as the accessible nodes. It is clear from the plots that the average sensing effort required by the chain network is greater than that of the grid network for a given measurement energy, which is defined as $\|\mathbf{Y}(t)\|^2_{L^2([0\ T],\mathbb{R}^k)}$ [102], where $\mathbf{Y}(t)$ is obtained from Equation 8.3.

Theorem 8. *Let $\mathcal{G}$ be an unweighted, undirected graph that represents the communication network of a set of N nodes with information dynamics and output map given by Equation 8.2 and Equation 8.3, respectively. If we label $V(\mathcal{G})$ such that $k \leq N$ sensor nodes in $V(\mathcal{G})$ are labeled as $1, 2, ..., k$, then $\mathbf{C} = \left[\mathbf{I}_{k\times k}\ \mathbf{0}_{k\times(N-k)}\right]$. Assuming that $\mathbf{L}(\mathcal{G})$ is diagonalizable and that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$ are its eigenvalues, there exist real constants $c_1 \leq c_2 \leq ... \leq c_N$ such that*

$$\sum_{i=1}^{k} c_i \ \leq\ Trace\left(W_O(0,T)\right) \ \leq\ \sum_{i=0}^{k-1} c_{N-i}, \tag{8.13}$$

where $c_i = \int_0^T e^{-2\lambda_i t}dt$.

Proof. From the definition of the trace operator, it can be shown that the trace and integral operators are commutative. Using this property and the property that the trace operator is invariant under cyclic permutation [66], Equation 8.12 can be written as

$$\begin{aligned} &Trace(W_O(0,T)) \\ &\qquad = \int_0^T Trace\left(\mathbf{e}^{-\mathbf{L}(\mathcal{G})^* t}\mathbf{C}^*\mathbf{C}\mathbf{e}^{-\mathbf{L}(\mathcal{G})t}\right)dt \\ &\qquad = \int_0^T Trace\left(\mathbf{C}^*\mathbf{C}\mathbf{e}^{(-\mathbf{L}(\mathcal{G})t-\mathbf{L}(\mathcal{G})^* t)}\right)dt. \end{aligned}$$

Since the Laplacian of an unweighted, undirected graph is a Hermitian matrix, this equation becomes

$$Trace(W_O(0,T)) = \int_0^T Trace\left(\mathbf{C}^*\mathbf{C}\mathbf{e}^{-2\mathbf{L}(\mathcal{G})t}\right)dt.$$

Let $\mathbf{L}(\mathcal{G}) = \mathbf{V}\Lambda\mathbf{V}^*$ such that $\Lambda = Diag(\lambda_1, \lambda_2, ..., \lambda_N)$ and the columns of $\mathbf{V} \in \mathbb{R}^{N\times N}$ are given by the corresponding eigenvectors of $\mathbf{L}(\mathcal{G})$. Then using the decomposition of the matrix exponential [127], the equation becomes

$$Trace(W_O(0,T)) = \int_0^T Trace\left(\mathbf{C}^*\mathbf{C}\mathbf{V}\mathbf{e}^{-2\Lambda t}\mathbf{V}^*\right)dt$$

$$Trace(W_O(0,T)) \;=\; Trace\left(\mathbf{C}^*\mathbf{C}\mathbf{V}\left(\int_0^T \mathbf{e}^{-2\Lambda t}dt\right)\mathbf{V}^*\right).$$

The matrix exponential $\int_0^T \mathbf{e}^{-2\Lambda t}dt$ is a diagonal matrix given by $Diag\left(\int_0^T \mathbf{e}^{-2\lambda_1 t}dt, \int_0^T \mathbf{e}^{-2\lambda_2 t}dt, ..., \int_0^T \mathbf{e}^{-2\lambda_N t}dt\right)$. We define $c_i = \int_0^T \mathbf{e}^{-2\lambda_i t}dt$. Then, since $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$, by definition we have that $c_1 \leq c_2 \leq ... \leq c_N$.

Let $\mathbf{M} = \mathbf{V}\left(\int_0^T \mathbf{e}^{-2\Lambda t}dt\right)\mathbf{V}^*$. Then we see that $\mathbf{M}$ is a Hermitian matrix with eigenvalues $c_1, c_2, ..., c_N$ and the same eigenvectors as $\mathbf{L}(\mathcal{G})$. Also, we find that $\mathbf{C}^*\mathbf{C}$ is a diagonal matrix with the first k diagonal elements equal to 1 and the rest equal

to 0. Defining $\mathbf{P} = \mathbf{C}^*\mathbf{C}$, we obtain a compact form for the trace of the observability Gramian,

$$Trace\,(W_O(0,T)) = Trace\,(\mathbf{PM})\,. \tag{8.14}$$

Equation 8.14 can be reduced to:

$$Trace\,(W_O(0,T)) = Trace\,(\mathbf{PM}) = \sum_{i=1}^{k} M_{ii},$$

where M_{ii} denotes the i^{th} diagonal entry of $\mathbf{M}$.

From Theorem 1 of [29], we obtain the following lower bound:

$$Trace\,(W_O(0,T)) = \sum_{i=1}^{k} M_{ii} \;\geq\; \sum_{i=1}^{k} c_i. \tag{8.15}$$

Now by applying *Von Neumann's trace inequality* [66] to Equation 8.14 and the fact that $W_O(0,T)$ is at least positive semidefinite, we find that

$$Trace\,(\mathbf{PM}) \;\leq\; \sum_{i=0}^{n-1} \sigma\,(\mathbf{P})_{n-i}\,\sigma\,(\mathbf{M})_{n-i}$$

where $\sigma(\cdot)_i$ is the i^{th} singular value of a matrix. The singular values are arranged in increasing order, $\sigma(\cdot)_1 \leq \sigma(\cdot)_2 \leq ... \leq \sigma(\cdot)_N$, and here they coincide with the eigenvalues of the matrices. Note that only the last k eigenvalues of $\mathbf{P}$ are nonzero and are equal to 1. Thus, we obtain the upper bound:

$$Trace\,(W_O(0,T)) \;\leq\; \sum_{i=0}^{k-1} c_{N-i}. \tag{8.16}$$

□

Since we can obtain the eigenvalues of $\mathbf{L}(\mathcal{G}_c)$ and $\mathbf{L}(\mathcal{G}_g)$ analytically [41], we can use Theorem 8 to analyze and compare the scaling properties of the chain and grid network topologies in a more precise fashion. For each type of network, we specify that k of the N total nodes in the network are sensors (accessible nodes),

where $k < \sqrt{N}$. Without loss of generality, we assume that the grid is square to simplify the analysis. By Theorem 8, the upper bound on $Trace\,(W_O(0,T))$ is given by $\sum_{i=0}^{k-1} c_{N-i}$, where $c_{N-i} = \int_0^T e^{-2\lambda_{N-i}t}dt$. Let λ^c_{N-i} and λ^g_{N-i} denote the $(N-i)^{th}$ eigenvalue of $\mathbf{L}(\mathcal{G}_c)$ and $\mathbf{L}(\mathcal{G}_g)$, respectively. Then from [41], $\lambda^c_{N-i} = 4\sin^2\left(\frac{\pi i}{2N}\right)$ and $\lambda^g_{N-i} = 4\sin^2\left(\frac{\pi i}{2\sqrt{N}}\right)$ for $i \in \{0,1,...,k-1\}$. Since $k < \sqrt{N}$ implies that $\frac{k}{N} < \frac{1}{\sqrt{N}}$, for networks with large N we have that $\lambda^c_{N-i} \approx (\frac{\pi i}{N})^2$ and $\lambda^g_{N-i} \approx (\frac{\pi i}{\sqrt{N}})^2$. Therefore, the upper bound on $Trace\,(W_O(0,T))$ for the chain network is given by:

$$\sum_{i=0}^{k-1} c^c_{N-i} \;=\; \sum_{i=0}^{k-1} \int_0^T e^{-2\frac{\pi^2 i^2}{N^2}t}dt, \tag{8.17}$$

which can be simplified to

$$\sum_{i=0}^{k-1} c^c_{N-i} \;=\; T + \frac{N^2}{2\pi^2}\left(\sum_{i=1}^{k-1} \frac{1 - e^{-2\frac{\pi^2 i^2}{N^2}T}}{i^2}\right). \tag{8.18}$$

Similarly, the upper bound on $Trace\,(W_O(0,T))$ for the grid network can be reduced to:

$$\sum_{i=0}^{k-1} c^g_{N-i} \;=\; T + \frac{N}{2\pi^2}\left(\sum_{i=1}^{k-1} \frac{1 - e^{-2\frac{\pi^2 i^2}{N}T}}{i^2}\right). \tag{8.19}$$

From Equation 8.18 and Equation 8.19, we observe that the upper bound on the average sensing effort required by the chain network scales quadratically with the total number of nodes N, whereas this upper bound for the grid network scales linearly with N.

8.8 Effect of Network Topology on Robustness to Noise

In this section, we analyze the effect of noise on the output of first-order linear dynamics that evolve on chain and grid network topologies. We assume that the data at each node in the network is affected by white noise with zero mean and unit covariance. Therefore, the augmented system dynamics described by Equation 8.2

can be written as

$$\dot{\mathbf{X}}(t) = -\mathbf{L}(\mathcal{G})\mathbf{X}(t) + W, \tag{8.20}$$

where $W \in \mathbf{R}^N$ denotes a zero mean, unit covariance white noise process. The output equation is the same as Equation 8.3.

As defined in the robust control literature, the $\mathcal{H}_2$ norm of a system gives the steady-state variance of the output when the input to the system is white noise and when $-\mathbf{L}(\mathcal{G})$ is Hurwitz [37]. However, for unstable systems, the finite steady-state variance can be computed only when the unstable modes are unobservable from the outputs [9]. For $\mathbf{L}(\mathcal{G})$, zero is the only unstable mode with corresponding eigenvector $\mathbf{1}_N$, which does not affect the steady-state variance of the output. If we can make the zero mode unobservable, then it is still possible to use the $\mathcal{H}_2$ norm as a measure to quantify the effect of noise on the system output.

In order to do so, we follow the approach in [122], which uses the *first-order Laplacian energy*. This quantity is essentially the $\mathcal{H}_2$ norm of a system if the matrix $\mathbf{C}$ in Equation 8.3 is chosen in such a way that it annihilates the vector $\mathbf{1}_N$. This can be done by defining $\mathbf{C}$ to be an incidence matrix of a graph $\mathcal{G}_k$. Denoting this new $\mathbf{C}$ by $\hat{\mathbf{C}}$, we have that $\mathbf{L}(\mathcal{G}_k) = \hat{\mathbf{C}}^T\hat{\mathbf{C}}$. Then $\mathbf{L}(\mathcal{G}_k)\mathbf{1}_N = 0$, which implies that $\hat{\mathbf{C}}\mathbf{1}_N = 0$ since $ker(\hat{\mathbf{C}}) = ker(\hat{\mathbf{C}}^T\hat{\mathbf{C}})$. Note that $\hat{\mathbf{C}}$ need not necessarily be the incidence matrix of a graph $\mathcal{G}_k$; the only condition required is that $\hat{\mathbf{C}}^T\hat{\mathbf{C}} = \mathbf{L}(\mathcal{G}_k)$.

Now, if $\mathcal{G}_k$ is chosen to be a weighted complete graph $\mathcal{K}_N$ whose edges all have weight $\frac{1}{N}$, then $\mathbf{L}(\mathcal{G}_k) = \mathbf{I}_{N\times N} - \frac{1}{N}\mathbf{J}_N$. The first-order Laplacian energy, $\mathcal{H}^{(1)}_{\mathcal{K}_N}(\mathbf{L}(\mathcal{G}))$, for the corresponding $\mathbf{C}$ can be defined from [122] as

$$\mathcal{H}^{(1)}_{\mathcal{K}_N}(\mathbf{L}(\mathcal{G})) = \sum_{i=1}^{N-1} \frac{1}{2\lambda_i}, \tag{8.21}$$

where $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N = 0$ are the eigenvalues of $\mathbf{L}(\mathcal{G})$.

In figure 8.6, we compare $\mathcal{H}^{(1)}_{\mathcal{K}_N}(\mathbf{L}(\mathcal{G}))$ for graphs with grid and chain network

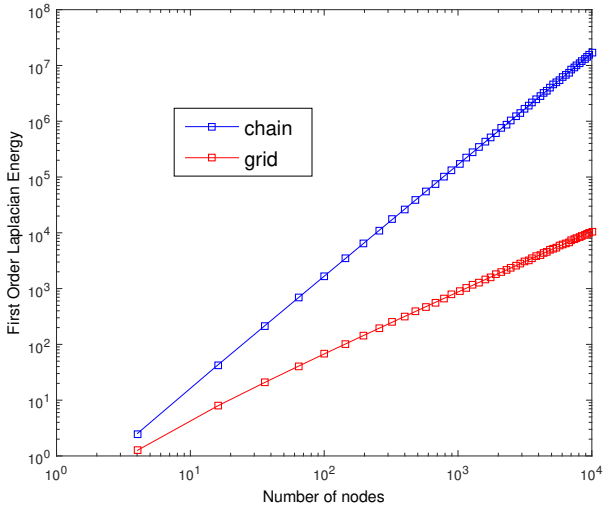


Figure 8.6: Performance measure based on the first-order Laplacian energy.

topologies as a function of the total number of nodes in the network. The plot shows that the grid network is more effective than the chain network at mitigating the effect of noise on the system output for a given number of nodes.

Chapter 9

CONCLUSION AND FUTURE WORK

This thesis has presented approaches for solving the problem of exploring and mapping an unknown environment using resource-constrained robotic swarms, outlined in Chapter 1. The thesis also introduces and solves the problem of reconstructing two-dimensional scalar field using measurements from a subset of a network with local communication between nodes in a sensor network. Chapter 3 describes an approach to *mapping the geometry of a single region of interest* using observations from a swarm of robots without communication or the ability to measure or estimate their location. The approach solves the mapping problem by formulating it as an optimal control problem for a set of partial differential equations. Chapter 4 and Chapter 5 focuses on quantifying the topological features and generating the map of an unknown environment respectively using uncertain position data from the swarm of robots that explored the unknown environment. The approaches presented in Chapter 4 and Chapter 5 use concepts from algebraic topology to perform the task. Also, Chapter 6 extended the strategy presented in Chapter 5 to construct a metric map of the unknown environment using data from the robots that explored the unknown environment. For the scenario described in this chapter, the robots use the data obtained during random exploration of the domain by combining onboard odometry information with noisy measurements of signals received from transmitters located outside the domain. The approach is substantiated with theoretical results to prove its completeness and to analyze its time complexity. A novel exploration strategy which combines information theoretic concepts with Lévy walk is delineated in Chapter 7. This chapter also outlines

an occupancy grid map based distributed mapping technique which invokes the ideas behind average consensus in time-varying networks. Unlike other mapping strategies presented in this thesis, the distributed mapping computes the map of the explored domain online. The strategy also ensures that the local occupancy grid map stored in each robot converges to the global map of the domain. Finally, in Chapter 8 the problem of *mapping a two-dimensional scalar field* using measurements from a subset of a network with local communication between nodes is studied. Apart from this the chapter also derives bounds on the trace of the observability Gramian of the large network system, which can be used to quantify and compare the field estimation capabilities of chain and grid networks. A comparison based on a performance measure related to the $\mathcal{H}_2$ norm of the system is also used to study the robustness of the network topologies. All strategies described in this thesis are validated through simulations. The strategies presented in Chapter 4 and Chapter 5 were experimentally validated using robotic platform *Pheeno*. The exploration and distributed mapping strategies present in Chapter 7 were validated using the *Turtlebot 3 Burger* robots. The following sections describe conclusions and possible future research directions that pertain to the work in Chapter 3 to Chapter 8.

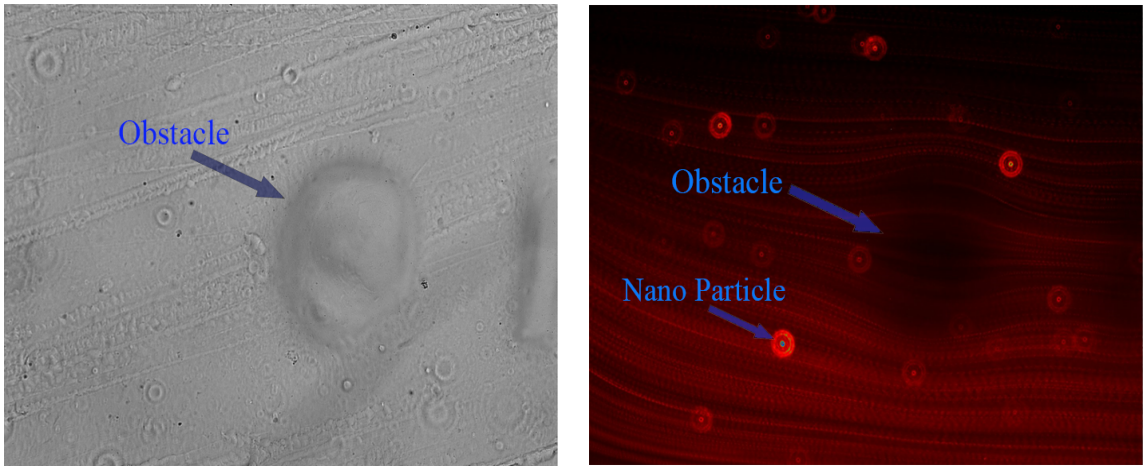
9.1 An Advection-Diffusion-Reaction Based Approach to Mapping an Environmental Feature

Chapter 3 presented a method for mapping an environmental feature using a robotic swarm that exhibits diffusive motion and lacks localization and inter-robot communication. The approach employs optimal control techniques to reconstruct a spatially varying function that represents the feature of interest. This function is estimated using temporal data on the proportion of active robots, which have not encountered the feature, at each instant of time. The simulation results indicate that

this methodology can accurately reconstruct the feature when the data is obtained from multiple swarm deployments that originate at different locations throughout the domain.

A major factor that contributes to mapping inaccuracies in the work presented in Chapter 3 is the decrease in number of active robots during a swarm deployment as robots encounter a feature of interest. This decrease in active robots can reduce the correspondence between the density fields of active robots in the macroscopic and microscopic models. This decrease can be prevented if the robots perform an obstacle avoidance maneuver upon encountering a feature, staying in the active state rather than entering the passive state. The corresponding macroscopic PDE model of the swarm population dynamics would need to include this avoidance behavior. An approach to incorporate the obstacle avoidance behavior in the PDE is to model the set of obstacles as a spatially-dependent potential function $\phi(\mathbf{x})$ called a navigation function [110]. In the microscopic model, the negative gradient of the potential function would generate a component of the robots' velocity that guides the robots away from obstacles. In the macroscopic model, this obstacle avoidance would be implemented by modifying the right-hand side of Equation 3.3 to be $\nabla \cdot (D\nabla u - \mathbf{v}(t)u + \nabla\phi(\mathbf{x})u)$. Then the problem of mapping the domain can be reduced to finding $\phi(\mathbf{x})$ or the support of this function [79] from the arrival times of robots near the boundary of the domain.

An experimental verification of the PDE based strategy would be an interesting future work. Since the robot capability requirements for the strategy is minimal, the experiment can be conducted using a swarm of nanoparticles [61] that advect and diffuse through fluid flow in a microfluidic channel with a circular obstacle (Figure 9.1), which represents a cellular structure of interest such as a tumor. The primary goal of these experiments is to estimate the distance of the obsta-



(a) Circular obstacle (b) Nanoparticles near the circular obstacle

Figure 9.1: *Left:* Grayscale image of a circular obstacle in a microfluidic channel, as seen under an electron microscope. *Right:* 2μm Nanoparticles (in red) near the obstacle. The image is a superposition of a time-lapse series and was taken using an inverted fluorescent microscope. *Images are from the Bristol Robotics Laboratory, directed by Prof. Sabine Hauert.*

cle from the entrance to the channel using observations of the nanoparticles over time. Nanoparticles are a suitable platform for testing this mapping approach: they can be deployed in massive numbers, thus justifying the use of a macroscopic PDE model, and they can be approximated as point-mass robots, given their extremely small dimensions in comparison to their environment. Unlike the scenario described in Section 3.2, the nanoparticles do not have the ability to identify the obstacle and store their encounter times with it. To implement the mapping strategy, the nanoparticles will need to be engineered to adhere to the obstacle upon contact, and the number of "active" (i.e., moving) nanoparticles over time will need to be counted using image processing software.

9.2 A Probabilistic Topological Approach to Quantifying Environmental Features

Chapter 4 described the formulation of a new approach to identifying the numbers of topological features in an unknown domain by applying tools from Topological Data Analysis (TDA) to data collected by a robotic swarm. The proposed methodology was shown to be effective through simulations on different domains and experiments with mobile robots. It may be noted that the point cloud generated in the procedure is embedded with a metric, making it a *metric space*. In future, this metric could be incorporated into the approach to constructing a metric map of the unknown environment.

9.3 A Probabilistic Approach to Constructing Topological Maps of an Environment

In Chapter 5 a novel approach for automated generation of the topological map of an unknown environment using data collected by a robotic swarm without global localization or communication was formulated. The procedure combines tools from topological data analysis with graph-based algorithms to construct an approximate Generalized Voronoi Diagram, which yields collision-free paths through the environment for safe robot navigation. The proposed methodology was validated through simulations of domains with different numbers of obstacles and experiments with mobile robots.

9.4 Automated Construction of Metric Maps using a Stochastic Robotic Swarm Leveraging Received Signal Strength

A technique for automatically generating the metric map of an unknown environment as an occupancy grid map using data collected by a robotic swarm without global localization or inter-robot communication was proposed in Chapter 6. This data combines the robots' odometry information with their noisy measurements of signals from transmitters outside the domain. The approach was validated through simulations on domains with different numbers of obstacles and robots and different deployment times. We plan to validate our method with multi-robot experiments and investigate the effect of distance-varying sensor noise models on the map accuracy.

9.5 Future work for topological approach to mapping

In future, topological methods presented in Chapter 4 to Chapter 6 can be extended to develop alternate mapping techniques using *Signal Embedding Theorem* proved in [112]. Signal Embedding Theorem states that a set of five signals that vary smoothly within an environment can be used to localize a receiver in the environment. Examples of signals that could be used are the intensity of a radio signal or a smoothly-varying heat profile. Based on the theorem mapping strategy that uses topological data analysis techniques to compute the topological features of an unknown domain from measurements of five distinct types of signals that are received by a swarm of robots during random exploration can developed. One could use a non-linear dimensionality reduction technique like laplacian eigenmaps[85] to invert the map from two dimensional coordinates to five dimensional signals. Schwartz et. al [119] approaches the localization problem in a similar way by com-

bining the non-linear dimensionality reduction techniques with artificial neural networks.

9.6 Information correlated Lévy walk exploration and distributed mapping

In Chapter 7, a novel methodology to map an unknown environment in form of occupancy grid map using a swarm of robots in a distributed fashion was developed. We also developed an exploration strategy based on information correlated Lévy walk, which directed Lévy walking robots to regions with more information gain. Finally, we demonstrated that our TDA based approach developed in our early works which was initially used to construct topological maps of unknown domains can also be used for adaptively thresholding occupancy grid maps. The various techniques developed in this paper is validated through simulation and experiments on domains with different size and shapes. Overall, we believe this to be an effective distributed mapping approach for robot swarms due to its scalability and robustness to robot failures. In future work, it would be interesting to extend our mapping strategy to the case of weak or no global localization information and also analysis the technique to understand the kind of guarantee our strategy can provide in a weak localization setting. This could become a viable technology in future as our approach can then be coupled with existing ultra wide band(UWB) or received signal strength indicator(RSSI) technologies to perform mapping on large complex environments. We would also like to derive analytic equations describing the evolution of robot population dispersing in a domain according to our information correlated Lévy walk strategy.

9.7 Scalar field estimation using a sensor network

A methodology to estimate the initial state of a large networked system of nodes with first-order linear information dynamics using output measurements from a subset of the nodes was presented in Chapter 8. The work in Chapter 8 outlines the advantages of a grid network over a chain network in the estimation of a two-dimensional scalar field, even though both networks can be made observable by construction. A performance measure based on the $\mathcal{H}_2$ norm of the network was used to characterize the robustness of the network dynamics based on its structure.

As an extension to the work presented in Chapter 8, the effect of sensor network topology on distributed estimation of scalar fields by a network can be investigated. This work would require the application of graph theoretic methods to derive conditions on the network topology that will enable distributed estimation of a scalar field. The goal of this work would be to investigate the ability of a particular communication network topology to perform distributed estimation of scalar field in a robust fashion. In steady state, each node would have an estimate of the whole network field.

REFERENCES

[1] Milton Abramowitz. *Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables*. Dover Publications, 1974. ISBN 0486612724.

[2] Henry Adams and Gunnar Carlsson. On the nonlinear statistics of range image patches. *SIAM Journal on Imaging Sciences*, 2(1):110–117, Jan 2009. ISSN 1936-4954.

[3] Henry Adams, Andrew Tausz, and Mikael Vejdemo-Johansson. javaPlex: A research software package for persistent (co)homology. In *Mathematical Software – ICMS*, pages 129–136. Springer Berlin Heidelberg, 2014.

[4] Ayaz Ahmad, Gian Diego Tipaldi, Pedro Lima, and Wolfram Burgard. Cooperative robot localization and target tracking based on least squares minimization. In *IEEE Int'l. Conference on Robotics and Automation (ICRA)*, pages 5696–5701, 2013.

[5] Mahmuda Ahmed, Brittany Terese Fasy, and Carola Wenk. Local persistent homology based distance between maps. In *ACM SIGSPATIAL Int'l. Conf. on Advances in Geographic Information Systems (SIGSPATIAL)*, pages 43–52, 2014. ISBN 978-1-4503-3131-9.

[6] Gianluca Antonelli and Stefano Chiaverini. Kinematic control of platoons of autonomous vehicles. *IEEE Trans. on Robotics*, 22(6):1285–1292, 2006.

[7] R. Aragues, J. Cortes, and C. Sagues. Distributed consensus on robot networks for dynamically merging feature-based maps. *IEEE Transactions on Robotics*, 28(4):840–854, Aug 2012. ISSN 1552-3098. doi: 10.1109/TRO.2012.2192012.

[8] Franz Aurenhammer. Voronoi diagrams–a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, September 1991. ISSN 0360-0300.

[9] B. Bamieh, M. R. Jovanović, P. Mitra, and S. Patterson. Coherence in large-scale networks: dimension dependent limitations of local feedback. *IEEE Trans. on Automatic Control*, 57(9):2235–2249, Sept. 2012.

[10] Timothy Banham. The discrete Laplacian and the hotspot conjecture. `https://www.math.washington.edu/~reu/papers/2006/tim/laplace.pdf`, August 2006.

[11] Aziz Belmiloudi. *Stabilization, Optimal and Robust Control: Theory and Applications in Biological and Physical Sciences*. Springer-Verlag London, 2008.

[12] S. Bhattacharya, R. Ghrist, and V. Kumar. Persistent homology for path planning in uncertain environments. *IEEE Transactions on Robotics*, 31(3): 578–590, June 2015. ISSN 1552-3098.

[13] Mauro Birattari. Automatic design of robot swarms: achievements and challenges. *Frontiers in Robotics and AI*, 3:29, 2016.

[14] A. Birk and S. Carpin. Merging occupancy grid maps from multiple robots. *Proceedings of the IEEE*, 94(7):1384–1397, July 2006. ISSN 0018-9219. doi: 10.1109/JPROC.2006.876965.

[15] R. Biswas, B. Limketkai, S. Sanner, and S. Thrun. Towards object mapping in dynamic environments with mobile robots. In *Int'l. Conf. on Intelligent Robots and Systems (IROS)*, 2002.

[16] Alfio Borzì and Volker Schulz. *Computational Optimization of Systems Governed by Partial Differential Equations*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2012.

[17] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521833787.

[18] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.

[19] Stefano Carpin. Fast and accurate map merging for multi-robot systems. *Autonomous Robots*, 25(3):305–316, Oct 2008. ISSN 1573-7527. doi: 10.1007/s10514-008-9097-4. URL `https://doi.org/10.1007/s10514-008-9097-4`.

[20] S. H. Chang, P. C. Cosman, and L. B. Milstein. Chernoff-type bounds for the Gaussian error function. *IEEE Transactions on Communications*, 59(11):2939–2944, November 2011. ISSN 0090-6778.

[21] Benjamin Charrow, Gregory Kahn, Sachin Patil, Sikang Liu, Ken Goldberg, Pieter Abbeel, Nathan Michael, and Vijay Kumar. Information-theoretic planning with trajectory optimization for dense 3d mapping. In *Proc of Robotics: Science and Systems*, 2015.

[22] Benjamin Charrow, Sikang Liu, Vijay Kumar, and Nathan Michael. Information-theoretic mapping using cauchy-schwarz quadratic mutual information. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4791–4798, May 2015.

[23] H. Chintakunta and H. Krim. Distributed localization of coverage holes using topological persistence. *IEEE Transactions on Signal Processing*, 62(10):2531–2541, May 2014. ISSN 1053-587X.

[24] G.S. Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Applied and Numerical Harmonic Analysis. Birkhäuser Boston, 2011. ISBN 9780817649449. URL `https://books.google.co.in/books?id=hZ1CAAAAQBAJ`.

[25] Howie Choset and Keiji Nagatani. Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 17:125–137, 2001.

[26] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George A. Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, Pittsburgh, PA, June 2005.

[27] Nikolaus Correll and Heiko Hamann. Probabilistic modeling of swarming systems. In W. Pedrycz J. Kacprzyk, editor, *Springer Handbook of Computational Intelligence*, pages 1423–1431. Springer, 2015. ISBN 9783662435045.

[28] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, New York, NY, USA, 2006. ISBN 0471241954.

[29] Jamil Daboul. *Inequalities among partial traces of Hermitian operators and partial sums of their eigenvalues*. PhD thesis, Ben Gurion University of the Negev, 1990.

[30] Raffaello D'Andrea. Guest editorial: A revolution in the warehouse: A retrospective on kiva systems and the grand challenges ahead. *IEEE Transactions on Automation Science and Engineering*, 9(4):638–639, 2012.

[31] Jaydev P Desai, James P Ostrowski, and Vijay Kumar. Modeling and control of formations of nonholonomic mobile robots. *IEEE Trans. on Robotics and Automation*, 17(6):905–908, 2001.

[32] Cristina Dimidov, Giuseppe Oriolo, and Vito Trianni. Random walks in swarm robotics: An experiment with kilobots. In Marco Dorigo, Mauro Birattari, Xiaodong Li, Manuel López-Ibáñez, Kazuhiro Ohkura, Carlo Pinciroli, and Thomas Stützle, editors, *Swarm Intelligence*, pages 185–196, Cham, 2016. Springer International Publishing. ISBN 978-3-319-44427-7.

[33] Alireza Dirafzoon, Joseph Betthauser, Jeff Schornick, Daniel Benavides, and Edgar Lobaton. Mapping of unknown environments using minimal sensing from a stochastic swarm. In *Int'l. Conf. on Intelligent Robots and Systems (IROS)*, September 14–18, 2014.

[34] Alireza Dirafzoon, Alper Bozkurt, and Edgar J. Lobaton. Dynamic topological mapping with biobotic swarms. *CoRR*, abs/1507.03206, 2015.

[35] Alireza Dirafzoon, Alper Bozkurt, and Edgar Lobaton. A framework for mapping with biobotic insect networks: From local to global maps. *Robotics and Autonomous Systems*, 2016. ISSN 0921-8890.

[36] Marco Dorigo, Mauro Birattari, and Manuele Brambilla. Swarm robotics. *Scholarpedia*, 9(1):1463, 2014.

[37] Geir E. Dullerud and Fernando G. Paganini. *A course in robust control theory: A convex approach*. Texts in Applied Mathematics. Springer, New York, NY, USA, 2000.

[38] H Edelsbrunner and J. L Harer. *Computational topology : an introduction*. American Mathematical Society, Providence (R.I.), 2010. ISBN 978-0-8218-4925-5.

[39] Herbert Edelsbrunner and John L Harer. Persistent homology - a survey. *Contemporary Mathematics*, 453:257–282, 2008.

[40] Herbert Edelsbrunner and Dmitriy Morozov. Persistent homology: Theory and practice. *the European Congress of Mathematics*, pages 31–50, 2012.

[41] Thomas Edwards. The discrete Laplacian of a rectangular grid. `https://www.math.washington.edu/~reu/papers/2013/tom/Discrete%20Laplacian%20of%20a%20Rectangular%20Grid.pdf`, August 2013.

[42] Karthick Elamvazhuthi and Spring Berman. Optimal control of stochastic coverage strategies for robotic swarms. In *Int'l. Conf. on Robotics and Automation (ICRA)*, May 26–30, 2015.

[43] Karthik Elamvazhuthi. A variational approach to planning, allocation and mapping in robot swarms using infinite dimensional models. Master's thesis, Arizona State University, 2014.

[44] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, June 1989. ISSN 0018-9162. doi: 10.1109/2.30720.

[45] Chinwendu Enyioha, Mohammad Amin Rahimian, George J. Pappas, and Ali Jadbabaie. Controllability and fraction of leaders in infinite network. *CoRR*, abs/1410.1830, 2014. URL `http://arxiv.org/abs/1410.1830`.

[46] Lawrence C. Evans. *Partial differential equations*. American Mathematical Society, Providence, R.I., 2010. ISBN 9780821849743 0821849743.

[47] Hector O. Fattorini. *Infinite Dimensional Optimization and Control Theory*, volume 54. Cambridge University Press, 1999.

[48] Brian Ferris, Dirk Hahnel, and Dieter Fox. *Gaussian processes for signal strength-based location estimation*, volume 2, pages 303–310. MIT Press Journals, 2007. ISBN 9780262693486.

[49] R. Fujisawa and S. Dobata. Lvy walk enhances efficiency of group foraging in pheromone-communicating swarm robots. In *Proceedings of the 2013 IEEE/SICE International Symposium on System Integration*, pages 808–813, Dec 2013. doi: 10.1109/SII.2013.6776760.

[50] C. W. Gardiner. *Stochastic Methods: A Handbook for the Natural and Social Sciences*. Springer, Evanston, IL, USA, 4th edition edition, 2009.

[51] Eroi Gelenbeabc, Nestor Schmajukab, John Staddonb, and John Reifc. Autonomous search by robots and animals: A survey. *Robotics and Autonomous Systems*, 22:23–34, 1997.

[52] R. Ghrist. *Elementary Applied Topology*. Createspace, 1st edition, 2014.

[53] R. Ghrist, D. Lipsky, J. Derenick, and A. Speranzon. Topological landmark-based navigation and mapping. *University of Pennsylvania, Department of Mathematics, Tech. Rep. Vol. 8*, 2012.

[54] Robert Ghrist. Barcodes: the persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.

[55] Robert Ghrist and Abubakr Muhammad. Coverage and hole-detection in sensor networks via homology. In *IPSN*, 2005. ISBN 0-7803-9202-7.

[56] Chris Godsil and Gordon Royle. *Algebraic Graph Theory*, volume 207 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2001.

[57] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006. ISBN 013168728X.

[58] David Gonzalez-Arjona, Alberto Sanchez, Fernando López-Colino, Angel de Castro, and Javier Garrido. Simplified occupancy grid indoor mapping optimized for low-cost robots. *ISPRS International Journal of Geo-Information*, 2(4):959–977, 2013.

[59] Markus Haid. Learn to walk and run a little bit. `https://ayorho.wordpress.com/category/kuliah-course/prof-dr-ing-markus-haid-tpu-task-planning-in-unstructured-environments/`.

[60] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, New York, NY, 2002.

[61] Sabine Hauert, Spring Berman, Radhika Nagpal, and Sangeeta N. Bhatia. A computational framework for identifying design guidelines to increase the penetration of targeted nanoparticles into tumors. *Nano Today*, 8(6): 566 – 576, 2013. ISSN 1748-0132. doi: http://dx.doi.org/10.1016/j.nantod.2013.11.001. URL `http://www.sciencedirect.com/science/article/pii/S1748013213001138`.

[62] Joao P. Hespanha. *Linear System Theory*. Princeton University Press, Princeton, NJ, USA, 1st edition, 2009.

[63] Micheal Hinze, Rene Pinnau, Micheal Ulbrich, and Stefan Ulbrich. *Optimization with PDE Constraints*, volume 23. Springer Netherlands, 2009.

[64] Marlis Hochbruck and Christian Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 34(5):1911–1925, 1997. doi: 10.1137/S0036142995280572. URL `http://dx.doi.org/10.1137/S0036142995280572`.

[65] Kenneth E. Hoff, III, John Keyser, Ming Lin, Dinesh Manocha, and Tim Culver. Fast computation of generalized Voronoi diagrams using graphics hardware. In *Proc. 26th Annual Conf. on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 277–286, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co. ISBN 0-201-48560-5.

[66] Roger A. Horn and Charles R. Johnson, editors. *Matrix Analysis*. Cambridge University Press, New York, NY, USA, 1986. ISBN 0-521-30586-1.

[67] Andrew Howard. Multi-robot simultaneous localization and mapping using particle filters. *The International Journal of Robotics Research*, 25(12):1243–1256, 2006. doi: 10.1177/0278364906072250. URL `https://doi.org/10.1177/0278364906072250`.

[68] S. Sitharama Iyengar and Richard R. Brooks. *Distributed Sensor Networks: Sensor Networking and Applications*. CRC press, 2012.

[69] Meng Ji and M. Egerstedt. Observability and estimation in distributed sensor networks. In *IEEE Conf. on Decision and Control (CDC)*, pages 4221–4226, Dec. 2007. doi: 10.1109/CDC.2007.4434659.

[70] Zutao Jiang, Jihua Zhu, Yaochen Li, Zhongyu Li, and Huimin Lu. Simultaneous merging multiple grid maps using the robust motion averaging. *arXiv preprint arXiv:1706.04463*, 2017.

[71] B. J. Julian. *Mutual Information-based Gradient-Ascent Control for Distributed Robotics*. PhD thesis, Massachusetts Institute of Technology, 2013.

[72] B. J. Julian, S. Karaman, and D. Rus. On mutual information-based control of range sensing robots for mapping applications. In *IEEE/RSJ Int'l. Conf. on Intelligent Robots and Systems (IROS)*, pages 5156–5163, November 2013.

[73] T. Kaczynski, K. Mischaikow, and M. Mrozek. *Computational Homology*. Springer-Verlag New York, 2004.

[74] Josip Kasac, Vladimir Milic, Branko Novakovic, Dubravko Majetic, and Danko Brezak. Initial conditions optimization of nonlinear dynamic systems with applications to output identification and control. In *Mediterranean Conf. on Control & Automation (MED)*, pages 1247–1252. IEEE, 2012.

[75] Jonghoek Kim, Fumin Zhang, and Magnus Egerstedt. A provably complete exploration strategy by constructing Voronoi diagrams. *Autonomous Robots*, 29(3):367–380, 2010.

[76] D. B. Kingston and R. W. Beard. Discrete-time average-consensus under switching network topologies. In *2006 American Control Conference*, pages 6 pp.–, June 2006. doi: 10.1109/ACC.2006.1657268.

[77] Andreas Kirsch. *An Introduction to the Mathematical Theory of Inverse Problems*, volume 120. Springer-Verlag New York, 2 edition, 2011.

[78] H. Kretzschmar and C. Stachniss. Information-theoretic compression of pose graphs for laser-based slam. *Int. J. Robot. Research*, 31(11):1219–1230, August 2012.

[79] Erwin O. Kreyszig. *Introductory Functional Analysis with Applications*. Wiley classics library. Wiley India Pvt. Limited, 2007. ISBN 9788126511914. URL `https://books.google.com/books?id=osXw-pRsptoC`.

[80] Benjamin Kuipers and Yung-Tai Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Int'l. J. of Robotics and Autonomous Sys.*, 8(1–2):47–63, 1991.

[81] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, New York, NY, USA, 2006. ISBN 0521862051.

[82] Sikang Liu, Kartik Mohta, Shaojie Shen, and Vijay Kumar. Towards collaborative mapping and exploration using multiple micro aerial robots. In *Int'l. Symp. on Experimental Robotics (ISER)*, June 2014.

[83] Yang-Yu Liu, Jean-Jacques Slotine, and Albert-László Barabási. Observability of complex systems. *Proc. of the National Academy of Sciences*, 110(7):2460–2465, 2013. doi: 10.1073/pnas.1215508110. URL `http://www.pnas.org/content/110/7/2460.abstract`.

[84] David G. Luenberger. *Optimization by Vector Space Methods*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1997. ISBN 047118117X.

[85] Yunqian Ma and Yun Fu. *Manifold Learning Theory and Applications*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 2011. ISBN 1439871094, 9781439871096.

[86] M. Mesbahi and M. Egerstedt. *Graph Theoretic Methods for Multiagent Networks*. Princeton University Press, Princeton, NJ, USA, 1st edition, Sept. 2010.

[87] Daniel Meyer-Delius, Maximilian Beinhofer, and Wolfram Burgard. Occupancy grid models for robot mapping in changing environments. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pages 2024–2030. AAAI Press, 2012. URL `http://dl.acm.org/citation.cfm?id=2900929.2901014`.

[88] Cleve Moler and Charles Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003. doi: 10.1137/S00361445024180. URL `http://dx.doi.org/10.1137/S00361445024180`.

[89] L. Moreau. Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, 50(2):169–182, Feb 2005. ISSN 0018-9286. doi: 10.1109/TAC.2004.841888.

[90] P. C. Müller and H. I. Weber. Analysis and optimization of certain qualities of controllability and observability for linear dynamical systems. *Automatica*, 8(3):237–246, May 1972. ISSN 0005-1098. doi: 10.1016/0005-1098(72)90044-1. URL http://dx.doi.org/10.1016/0005-1098(72)90044-1.

[91] J.R. Munkres. *Topology: A First Course*. Prentice-Hall, 1988. ISBN 9780876922903. URL https://books.google.co.in/books?id=wFpYPwAACAAJ.

[92] Vidit Nanda. Perseus, the persistent homology software. http://www.sas.upenn.edu/~vnanda/perseus, May 2018.

[93] National Centers for Environment Information. Salinity of Atlantic ocean. http://ecowatch.ncddc.noaa.gov/thredds/catalog/amseas/catalog.html, Sept. 2015.

[94] Iñaki Navarro and Fernando Matía. An introduction to swarm robotics. *ISRN Robotics*, 2013, 2012.

[95] Iñaki Navarro and Fernando Matía. A survey of collective movement of mobile robots. *International Journal of Advanced Robotic Systems*, 10(1):73, 2013.

[96] Xuanlong Nguyen, Michael I. Jordan, and Bruno Sinopoli. A kernel-based learning approach to ad hoc sensor network localization. *ACM Trans. Sen. Netw.*, 1(1):134–152, August 2005. ISSN 1550-4859.

[97] Giuseppe Notarstefano and Gianfranco Parlangeli. Controllability and observability of grid graphs via reduction and symmetries. *IEEE Trans. on Automatic Control*, 58(7):1719–1731, July 2013.

[98] R. Olfati-Saber and R. M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, Sept 2004. ISSN 0018-9286. doi: 10.1109/TAC.2004.834113.

[99] Lorenzo Orecchia, Sushant Sachdeva, and Nisheeth K. Vishnoi. Approximating the exponential, the Lanczos method and an $\tilde{O}(m)$-time spectral algorithm for balanced separator. In *Proc. of the ACM Symposium on Theory of Computing*, STOC '12, pages 1141–1160, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1245-5. doi: 10.1145/2213977.2214080. URL http://doi.acm.org/10.1145/2213977.2214080.

[100] Gianfranco Parlangeli and Giuseppe Notarstefano. On the reachability and observability of path and cycle graphs. *IEEE Trans. on Automatic Control*, 57 (3):743–748, March 2012.

[101] F. Pasqualetti and S. Zampieri. On the controllability of isotropic and anisotropic networks. In *IEEE Conf. on Decision and Control (CDC)*, pages 607–612, Dec 2014. doi: 10.1109/CDC.2014.7039448.

[102] Fabio Pasqualetti, Sandro Zampieri, and Francesco Bullo. Controllability metrics and algorithms for complex networks. *CoRR*, abs/1308.1201, 2013. URL `http://arxiv.org/abs/1308.1201`.

[103] K. Pathak, A. Birk, J. Poppinga, and S. Schwertfeger. 3d forward sensor modeling and application to occupancy grid based sensor fusion. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2059–2064, Oct 2007. doi: 10.1109/IROS.2007.4399406.

[104] Florian T. Pokorny, Carl Henrik Ek, Hedvig Kjellström, and Danica Kragic. Persistent homology for learning densities with bounded support. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in NIPS 25*, pages 1826–1834. 2012.

[105] Florian T Pokorny, Johannes A Stork, and Danica Kragic. Grasping objects with holes: A topological approach. In *IEEE Int'l. Conf. on Robotics and Automation (ICRA)*, pages 1100–1107, 2013.

[106] Amanda Prorok and Alcherio Martinoli. Accurate indoor localization with ultra-wideband using spatial models and collaboration. *Int. J. Rob. Res.*, 33(4): 547–568, April 2014. ISSN 0278-3649. doi: 10.1177/0278364913500364. URL `http://dx.doi.org/10.1177/0278364913500364`.

[107] R. Ramaithitima, M. Whitzer, S. Bhattacharya, and V. Kumar. Automated creation of topological maps in unknown environments using a swarm of resource-constrained robots. *IEEE Robotics and Automation Letters*, 1(2):746–753, July 2016. ISSN 2377-3766.

[108] K. Reif, S. Gunther, E. Yaz, and R. Unbehauen. Stochastic stability of the discrete-time extended Kalman filter. *IEEE Transactions on Automatic Control*, 44(4):714–728, Apr 1999. ISSN 0018-9286.

[109] Wei Ren and R. W. Beard. Consensus of information under dynamically changing interaction topologies. In *Proceedings of the 2004 American Control Conference*, volume 6, pages 4939–4944 vol.6, June 2004. doi: 10.23919/ACC.2004.1384632.

[110] E. Rimon and D. E. Koditschek. The construction of analytic diffeomorphisms for exact robot navigation on star worlds. In *Proceedings, 1989 International Conference on Robotics and Automation*, pages 21–26 vol.1, May 1989. doi: 10.1109/ROBOT.1989.99962.

[111] Patrick Robertson, Michael Angermann, and Bernhard Krach. Simultaneous localization and mapping for pedestrians using only foot-mounted inertial sensors. In *Int'l. Conf. on Ubiquitous Computing (Ubicomp)*, pages 93–96, 2009. ISBN 978-1-60558-431-7.

[112] M. Robinson and R. Ghrist. Topological localization via signals of opportunity. *IEEE Transactions on Signal Processing*, 60(5):2362–2373, May 2012. ISSN 1053-587X. doi: 10.1109/TSP.2012.2187518.

[113] Michael Robinson. *Topological Signal Processing*. Springer-Verlag Berlin Heidelberg, 2014. ISBN 978-3-642-36103-6.

[114] C. Röhrig, , D. Heß, and F. Künemund. Global localization of mobile robots using signal strength readings from floor-installed RFID transponders. In *Proceedings of the International MultiConference of Engineers and Computer Scientists 2016 (IMECS 2016)*, pages 179–184, March 2016.

[115] Sajad Saeedi, Liam Paull, Michael Trentini, and Howard Li. Occupancy grid map merging for multiple robot simultaneous localization and mapping. *International Journal of Robotics and Automation*, 30(2):149–157, 2015.

[116] Sajad Saeedi, Liam Paull, Michael Trentini, and Howard Li. Occupancy grid map merging for multiple robot simultaneous localization and mapping. *International Journal of Robotics and Automation*, 30(2):149–157, 2015.

[117] Sajad Saeedi, Michael Trentini, Mae Seto, and Howard Li. Multiple-robot simultaneous localization and mapping: A review. *J. Field Robot.*, 33(1): 3–46, January 2016. ISSN 1556-4959. doi: 10.1002/rob.21620. URL `http://dx.doi.org/10.1002/rob.21620`.

[118] Erol Şahin. Swarm robotics: From sources of inspiration to domains of application. In *Swarm Robotics*, pages 10–20. Springer, 2004.

[119] A. Schwartz and R. Talmon. Intrinsic Isometric Manifold Learning with Application to Localization. *ArXiv e-prints*, June 2018.

[120] Robert Sedgewick and Kevin Wayne. *Algorithms, 4th Edition.* Addison-Wesley, 2011. ISBN 978-0-321-57351-3.

[121] Scott Seidel and Theodore Rappaport. 914mhz path loss prediction models for indoor wireless communications in multi-floored buildings. *IEEE Trans. on Antennas*, (2):207–217, 1992.

[122] Milad Siami and Nader Motee. Graph-theoretic bounds on disturbance propagation in interconnected linear dynamical networks. *arXiv preprint arXiv:1403.1494*, 2014.

[123] *LMS200/211/221/291 Laser Measurement Systems*. SICK sensor intelligence, 2006.

[124] P. Skraba, M. Ovsjanikov, F. Chazal, and L. Guibas. Persistence-based segmentation of deformable shapes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, pages 45–52, 2010.

[125] J. Smisek, M. Jancosek, and T. Pajdla. 3d with kinect. In *IEEE International Conference on Computer Vision Workshops*, pages 1154–1160, November 2011.

[126] Randall Smith, Matthew Self, and Peter Cheeseman. A stochastic map for uncertain spatial relationships. In *Proceedings of the 4th International Symposium on Robotics Research*, pages 467–474, Cambridge, MA, USA, 1988. MIT Press. ISBN 0-262-02272-9. URL `http://dl.acm.org/citation.cfm?id=57425.57472`.

[127] Gilbert Strang. *Linear Algebra and Its Applications*. Brooks/Cole, NY, USA, 3rd edition edition, 1988.

[128] D. Sutantyo, P. Levi, C. Möslinger, and M. Read. Collective-adaptive Lévy flight for underwater multi-robot exploration. In *IEEE International Conference on Mechatronics and Automation*, pages 456–462, 2013. doi: 10.1109/ICMA.2013.6617961.

[129] T. Tao. *An Introduction to Measure Theory*. Graduate studies in mathematics. American Mathematical Society, 2013. ISBN 9781470409227. URL `https://books.google.co.in/books?id=SPGJjwEACAAJ`.

[130] **Ragesh K. Ramachandran** and Spring Berman. The effect of communication topology on scalar field estimation by large networks with partially accessible measurements. In *Proceedings of the 2017 American Control Conference (ACC)*, Seattle, WA, USA, May 24–26, 2017.

[131] **Ragesh K. Ramachandran** and Spring Berman. Automated construction of metric maps using a stochastic robotic swarm leveraging received signal strength. *IEEE Transactions on Robotics*, 2018. [In revision].

[132] **Ragesh K. Ramachandran**, Karthik Elamvazhuthi, and Spring Berman. An optimal control approach to mapping GPS-denied environments using a stochastic robotic swarm. In *Int'l. Symp. on Robotics Research (ISRR)*, 2015.

[133] **Ragesh K. Ramachandran**, Sean Wilson, and Spring Berman. A probabilistic topological approach to feature identification using a stochastic robotic swarm. In *Int'l. Symp. on Distributed Autonomous Robotic Systems (DARS)*, 2016.

[134] **Ragesh K. Ramachandran**, Sean Wilson, and Spring Berman. A probabilistic approach to automated construction of topological maps using a stochastic robotic swarm. *IEEE Robotics and Automation Letters*, 2(2):616–623, April 2017. ISSN 2377-3766.

[135] **Ragesh K. Ramachandran**, Zahi Kakish, and Spring Berman. Information correlated lévy walk exploration and distributed mapping using a swarm of robots. *IEEE Transactions on Robotics*, 2018. [In preparation].

[136] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *Int'l. Journal of Robotics Research*, 20(5):335–363, 2001.

[137] S. Thrun and A. Bücken. Integrating grid-based and topological maps for mobile robot navigation. In *Proc. AAAI 13th National Conf. on Artificial Intelligence*, 1996.

[138] Sebastian Thrun. Robotic mapping: A survey. In *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[139] Sebastian Thrun. Learning occupancy grid maps with forward sensor models. *Autonomous Robots*, 15(2):111–127, Sep 2003. ISSN 1573-7527. doi: 10.1023/A:1025584807625. URL https://doi.org/10.1023/A:1025584807625.

[140] Sebastian Thrun and Arno Bü. Integrating grid-based and topological maps for mobile robot navigation. In *Proc. 13th Nat'l. Conf. on Artificial Intelligence - Vol. 2*, AAAI'96, pages 944–950. AAAI Press, 1996. ISBN 0-262-51091-X.

[141] Sebastian Thrun and Arno Bü. Integrating grid-based and topological maps for mobile robot navigation. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, AAAI'96, pages 944–950. AAAI Press, 1996. ISBN 0-262-51091-X. URL http://dl.acm.org/citation.cfm?id=1864519.1864527.

[142] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN 0262201623.

[143] Sheng Tong, Eli J. Fine, Yanni Lin, Thomas J. Cradick, and Gang Bao. Nanomedicine: Tiny particles and machines give huge gains. *Annals of Biomedical Engineering*, 42(2):243–259, 2014.

[144] Benjamín Tovar, Luis Guilamo, and Steven M LaValle. Gap navigation trees: Minimal representation for visibility-based tasks. In *Algorithmic Foundations of Robotics VI*, pages 425–440. Springer, 2004.

[145] F. Tröltzsch and J. Sprekels. *Optimal Control of Partial Differential Equations: Theory, Methods, and Applications*. Graduate studies in mathematics. American Mathematical Society, 2010. ISBN 9780821849040. URL https://books.google.co.in/books?id=04yDAwAAQBAJ.

[146] Fredi Tröltzsch. *Optimal Control of Partial Differential Equations: Theory, Methods, and Applications*, volume 112. American Mathematical Society, Providence, RI, USA, 2010.

[147] Valery V. Tuchin. *Tissue Optics, Light Scattering Methods and Instruments for Medical Diagnosis*, volume PM254. Spie Press Book, 3rd edition, February 9 2015. ISBN 9781628415162.

[148] Valery V. Tuchin. *Tissue Optics, Light Scattering Methods and Instruments for Medical Diagnosis*, volume PM254. Spie Press Book, 3rd edition, February 9 2015. ISBN 9781628415162.

[149] Richard Vaughan. Massively multi-robot simulation in stage. *Swarm Intelligence*, 2(2):189–208, Dec 2008. ISSN 1935-3820. doi: 10.1007/s11721-008-0014-4. URL https://doi.org/10.1007/s11721-008-0014-4.

[150] Richard Vaughan. Stage. https://github.com/rtv/Stage, 2018.

[151] D. Williams. *Probability with Martingales*. Cambridge mathematical textbooks. Cambridge University Press, 1991. ISBN 9780521406055. URL https://books.google.co.in/books?id=RnOJeRpk0SEC.

[152] S. B. Williams, G. Dissanayake, and H. Durrant-Whyte. Towards multi-vehicle simultaneous localisation and mapping. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 3, pages 2743–2748, 2002. doi: 10.1109/ROBOT.2002.1013647.

[153] Richard C. Wilson and Ping Zhu. A study of graph spectra for comparing graphs and trees. *Pattern Recognition*, 41(9):2833 – 2841, 2008. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/j.patcog.2008.03.011. URL http://www.sciencedirect.com/science/article/pii/S0031320308000927.

[154] S. Wilson, R. Gameros, M. Sheely, M. Lin, K. Dover, R. Gevorkyan, M. Haberland, A. Bertozzi, and S. Berman. Pheeno, a versatile swarm robotic research and education platform. *IEEE Robotics and Automation Letters*, 1(2):884–891, July 2016. ISSN 2377-3766.

[155] Sean Wilson, Theodore P. Pavlic, Ganesh P. Kumar, Aurélie Buffin, Stephen C. Pratt, and Spring Berman. Design of ant-inspired stochastic control policies for collective transport by robotic swarms. *Swarm Intelligence*, 8(4):303–327, Dec 2014. ISSN 1935-3820. doi: 10.1007/s11721-014-0100-8. URL https://doi.org/10.1007/s11721-014-0100-8.

[156] Yaxiong Xie, Zhenjiang Li, and Mo Li. Precise power delay profiling with commodity wifi. In *Proc. 21st Annual Int'l. Conf. on Mobile Computing and Networking*, MobiCom '15, pages 53–64, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3619-2.

[157] Gang Yan, Georgios Tsekenis, Baruch Barzel, Jean-Jacques Slotine, Yang-Yu Liu, and Albert-Laszlo Barabási. Spectrum of controlling and observing complex networks. *Nat. Phys.*, 11(9):779–786, Sep 2015. ISSN 1745-2473. URL http://dx.doi.org/10.1038/nphys3422. Article.

[158] Zhengzhong Yuan, Chen Zhao, Zengru Di, Wen-Xu Wang, and Ying-Cheng Lai. Exact controllability of complex networks. *Nat. Commun.*, 4, Sept 2013. URL http://dx.doi.org/10.1038/ncomms3447.

[159] V Zaburdaev, S Denisov, and J Klafter. Lévy walks. *Reviews of Modern Physics*, 87(2):483, 2015.

[160] Tim Zajic, Ravi B. Ravichandran, Ronald P. S. Mahler, Raman K. Mehra, and Michael J. Noviskey. Joint tracking and identification with robustness against unmodeled targets. *Signal Processing, Sensor Fusion, and Target Recognition XII*, 5096:279–290, 2003.

[161] Bin Zhang and Gaurav S Sukhatme. Adaptive sampling for estimating a scalar field using a robotic boat and a sensor network. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3673–3680. IEEE, 2007.

[162] Afra J. Zomorodian, M. J. Ablowitz, S. H. Davis, E. J. Hinch, A. Iserles, J. Ockendon, and P. J. Olver. *Topology for Computing (Cambridge Monographs on Applied and Computational Mathematics)*. Cambridge University Press, New York, NY, 2005. ISBN 0521836662.

APPENDIX A

ADJOINT EQUATION

The adjoint equation $\nabla_{\mathbf{u}}\mathcal{L} = 0$ implies that $[\nabla_{u_1}\mathcal{L}, ..., \nabla_{u_i}\mathcal{L}, ..., \nabla_{u_N}\mathcal{L}] = 0$. From Equation 3.11,

$$\begin{aligned}\nabla_{u_i}\mathcal{L} &= \nabla_{u_i}\mathbf{J}(\mathbf{u}, K) + \nabla_{u_i}\sum_{j=1}^{N}\langle p_j, \Psi_j(u_j, K)\rangle \\ &= \nabla_{u_i}\mathbf{J}(u_i, K) + \nabla_{u_i}\langle p_i, \Psi_i(u_i, K)\rangle, \end{aligned} \tag{A.1}$$

since a term in the sum is a function of u_i only when $i = j$. By Equation 3.10,

$$\nabla_{u_i}\mathbf{J}(u_i, K) = \nabla_{u_i}\sum_{j=1}^{N} W_j J_j(u_j) = W_i \nabla_{u_i} J_i(u_i). \tag{A.2}$$

From Equation 3.9,

$$\nabla_{u_i} J_i(u_i) = \nabla_{u_i}\left(\frac{1}{2}\left\|(Du_i)(t) - g_i(t)\right\|^2_{L^2([0,T])}\right), \tag{A.3}$$

where $D := U \to L^2([0,T])$ and $(Du_i)(t) = \int_\Omega u_i(\vec{x}, t)d\vec{x}$. Then, by the chain rule of differentiation [63, 16], the directional derivative of $J_i(u_i)$, $\nabla_{u_i} J_i(u_i)$, is given by

$$\langle \nabla_{u_i} J_i(u_i), s\rangle_U = \langle (Du_i)(t) - g_i(t), Ds\rangle_{L^2([0,T])} = \langle D^*((Du_i)(t) - g_i(t)), s\rangle_U. \tag{A.4}$$

Here, $D^* := L^2([0,T]) \to U$ and $(D^* f)(t) = f(t) \cdot \vec{1}_\Omega(\vec{x})$, where $f(t) \in L^2([0,T])$ and $\vec{1}_\Omega$ is the indicator function of $\Omega \subset \mathbb{R}^2$. We can show that $\langle Dy, f\rangle = \langle y, D^* f\rangle \ \forall y \in U,\ f \in L^2([0,T])$. Therefore,

$$\nabla_{u_i} J_i(u_i) = D^*((Du_i)(t) - g_i(t)). \tag{A.5}$$

By definition,

$$\langle p_i, \nabla_{u_i}\Psi_i(u_i, K)s\rangle = \langle \nabla_{u_i}\Psi_i(u_i, K)^* p_i, s\rangle \ \ \forall s \in U, \tag{A.6}$$

where $\nabla_{u_i}\Psi_i(u_i, K)^*$ is the adjoint operator of $\nabla_{u_i}\Psi_i(u_i, K)$ corresponding to the inner product of the Hilbert space. Now, by taking the directional derivative of $\Psi_i(u_i, K)$ at u_i in the direction of s, we obtain

$$\nabla_{u_i}\Psi_i(u_i, K)s = \frac{\partial s}{\partial t} - (\nabla \cdot (D\nabla s - \mathbf{v}_i(t)s) - kK(\vec{x})s). \tag{A.7}$$

Substituting Equation A.7 into Equation A.6 yields

$$\langle p_i, \nabla_{u_i}\Psi_i(u_i, K)s\rangle = \int_0^T \langle p_i, \frac{\partial s}{\partial t}\rangle_{L^2(\Omega)} - \langle p_i, D\nabla^2 s\rangle + \langle p_i, \nabla \cdot \mathbf{v}_i(t)s\rangle + \langle p_i, kK(\vec{x})s\rangle.$$

Using integration by parts on the integral term in the equation above, we get

$$\int_0^T \langle p_i, \frac{\partial s}{\partial t}\rangle_{L^2(\Omega)} = \langle p_i(T), s(T)\rangle - \langle p_i(0), s(0)\rangle - \int_0^T \langle s, \frac{\partial p_i}{\partial t}\rangle_{L^2(\Omega)}.$$

As this is true for all $s \in U$, we could choose the s with $s(0) = 0$ and construct $p_i(T)$ such that $\int_0^T \langle p_i, \frac{\partial s}{\partial t}\rangle_{L^2(\Omega)} = \int_0^T \langle -\frac{\partial p_i}{\partial t}, s\rangle_{L^2(\Omega)}$. Thus, we choose the final condition of the adjoint equation as $p_i(T) = 0$. We now make use of the following lemma:

Lemma 2. *Let L and L^* be operators defined by $L : L^2(0,T;V) \rightarrow L^2(0,T;V^*)$ and $L^* : L^2(0,T;V) \rightarrow L^2(0,T;V^*)$, respectively. The variational form of L is:*

$$\langle Lu, \phi \rangle_{V^*,V} = -\left\langle D\nabla u, \nabla\phi \right\rangle_{L^2(\Omega)} - \langle \mathbf{v} \cdot \nabla u, \phi \rangle_{L^2(\Omega)} + \int_{\partial\Omega} \vec{n} \cdot (\mathbf{v} u \phi) dx$$

$\forall \phi \in V$. Also, by Lagrange's identity, $\langle Lu, p \rangle_{V^*,V} = \langle u, L^* p \rangle_{V,V^*}$ $\forall u, p \in L^2(0,T;V)$. We use the zero-flux boundary condition in Equation 3.4 to compute the variational form of the operator L^* to be $\langle L^* p, \phi \rangle_{V^*,V} = -\left\langle D\nabla p, \nabla\phi \right\rangle_{L^2(\Omega)} + \langle \mathbf{v} \cdot \nabla p, \phi \rangle_{L^2(\Omega)}$ $\forall p \in L^2(0,T;V)$ and $\forall \phi \in V$.

Using the variational form of the Laplacian as in Equation 3.8 and applying lemma 2 and integration by parts, we can show that $-\langle p_i, D\nabla^2 s \rangle + \langle p_i, \nabla \cdot \mathbf{v}_i(t) s \rangle$ can be transformed into $-\langle D\nabla^2 p_i, s \rangle - \langle \nabla \cdot \mathbf{v}_i(t) p_i, s \rangle$ with the boundary condition $\vec{n} \cdot \nabla p_i = 0$. Finally, we observe that $\langle p_i, K(\vec{x}) s \rangle = \langle p_i K(\vec{x}), s \rangle$. By combining these results with Equation A.1, Equation A.4, and Equation A.6, we obtain

$$\langle \nabla_{u_i} J_i(u_i), s \rangle + \langle -\frac{\partial p_i}{\partial t} - D\nabla^2 p_i - \nabla \cdot \mathbf{v}_i(t) p_i + p_i k K(\vec{x}), s \rangle = 0.$$

Thus, the set of adjoint equations for the system defined by the i^{th} set of constraints, $\Psi_i(u_i, K)$, with respect to the objective functional, **J**, is given by

$$-\frac{\partial p_i}{\partial t} = \nabla \cdot (D\nabla p_i + \mathbf{v}_i(t) p_i) - p_i k K(\vec{x}) - \nabla_{u_i} J_i(u_i) \quad in \;\; L \tag{A.8}$$

with the Neumann boundary conditions

$$\vec{n} \cdot \nabla p_i = 0 \quad on \;\; \Gamma, \quad p_i(T) = 0, \quad i = 1, ..., N. \tag{A.9}$$

Here, Equation A.8 with Equation A.9 has a solution in the weak sense.

APPENDIX B

GRADIENT EQUATION

Using a similar analysis to the one in appendix A, we find that $\nabla_K \mathcal{L}$ reduces to

$$\nabla_K \mathcal{L} = \nabla_K \mathbf{J}(\mathbf{u}, K) + \sum_{i=1}^{N} \nabla_K \langle p_i, \Psi_i(u_i, K) \rangle. \tag{B.1}$$

From Equation 3.10, we can derive the following expressions:

$$\nabla_K \mathbf{J}(\mathbf{u}, K) = \nabla_K \frac{\lambda}{2} \|K(\vec{x})\|^2_{L^2(\Omega)}, \quad \langle \nabla_K \mathbf{J}(\mathbf{u}, K), s \rangle = \langle \lambda K(\vec{x}), s \rangle. \tag{B.2}$$

As in appendix A, we could express $\langle p_i, \nabla_K \Psi_i(u_i, K)s \rangle$ as $\langle \nabla_K \Psi_i(u_i, K)^* p_i, s \rangle$ $\forall s \in L^2(\Omega)$, where $\nabla_K \Psi_i(u_i, K)^*$ is the adjoint operator of $\nabla_K \Psi_i(u_i, K)$ corresponding to the inner product of the Hilbert space. Now, by taking the directional derivative of $\Psi_i(u_i, K)$ at K in the direction of s, we find that $\nabla_K \Psi_i(u_i, K)s = k u_i s$. Therefore, with further simplification, we can show that

$$\langle \nabla_K \Psi_i(u_i, K)^* p_i, s \rangle = \langle (\Xi(k u_i p_i))(\vec{x}), s \rangle_{L^2(\Omega)}, \tag{B.3}$$

where $\Xi := L^2(0, T; \Omega) \to L^2(\Omega)$ and $(\Xi f)(\vec{x}) = \int_0^T f dt$ for all $f \in L^2([0, T]; \Omega)$ and $\vec{x} \in \Omega$. By combining Equation B.1-Equation B.3, we formulate the objective functional derivative as

$$\mathbf{J}' = \sum_{i=1}^{N} (\Xi(k u_i p_i))(\vec{x}) + \lambda K(\vec{x}). \tag{B.4}$$

Thus, the computation of $\mathbf{J}'$ requires u_i and p_i, which can be obtained by solving $\Psi_i(u_i, K)$ forward and solving Equation A.8, Equation A.9 backward.

APPENDIX C

DERIVATION OF FORMULA TO COMPUTE $\mathbf{I}[\mathbf{C}^{I,A}; Z_{\tau}^{I,A}]$

By definition,

$$\mathbf{I}[\mathbf{c}^{i,a}; z_\tau^{i,a}] = \int_{z \in z_\tau^{i,a}} \sum_{c \in \mathbf{c}^{i,a}} \mathbb{P}(c,z) \log\left(\frac{\mathbb{P}(c,z)}{\mathbb{P}(c)\mathbb{P}(z)}\right) dz$$

this can be rewritten as,

$$\begin{aligned}\mathbf{I}[\mathbf{c}^{i,a}; z_\tau^{i,a}] \quad = \quad & \int_{z \in z_\tau^{i,a}} \sum_{c \in \mathbf{c}^{i,a}} \mathbb{P}(c,z) \log\left(\mathbb{P}(z,c)\right) dz \\ & - \int_{z \in z_\tau^{i,a}} \sum_{c \in \mathbf{c}^{i,a}} \mathbb{P}(c,z) \log\left(\mathbb{P}(z)\mathbb{P}(c)\right) dz\end{aligned}$$

Let

$$\mathbf{P1} = \int_{z \in z_\tau^{i,a}} \sum_{c \in \mathbf{c}^{i,a}} \mathbb{P}(c,z) \log\left(\mathbb{P}(z,c)\right) dz$$

and

$$\mathbf{P2} = \int_{z \in z_\tau^{i,a}} \sum_{c \in \mathbf{c}^{i,a}} \mathbb{P}(c,z) \log\left(\mathbb{P}(z)\mathbb{P}(c)\right) dz$$

Now we focus on **P1**, expanding **P1** using Equation 7.8 yields,

$$\begin{aligned}\mathbf{P1} \quad = \quad & \int_{z \in z_\tau^{i,a}} \sum_{p=0}^{|\mathbf{c}^{i,a}|} (\mathbb{P}(z|e_p)\mathbb{P}(e_p) \log\left(\mathbb{P}(z|e_p)\right) \\ & + \mathbb{P}(z|e_p)\mathbb{P}(e_p) \log\left(\mathbb{P}(z)\right)) dz\end{aligned}$$

rearranging,

$$\begin{aligned}\mathbf{P1} \quad = \quad & \sum_{p=0}^{|\mathbf{c}^{i,a}|} \mathbb{P}(e_p) \int_{z \in z_\tau^{i,a}} \mathbb{P}(z|e_p) \log(\mathbb{P}(z|e_p)) dz \\ & + \sum_{p=0}^{|\mathbf{c}^{i,a}|} \mathbb{P}(e_p) \log(\mathbb{P}(e_p)) \int_{z \in z_\tau^{i,a}} \mathbb{P}(z|e_p) dz\end{aligned}$$

using the fact that $\int_{z \in z_\tau^{i,a}} \mathbb{P}(z|e_p) dz$ is unity and plugging in $\mathbb{P}(z|e_k)$ in the Gaussian forward sensor model(Equation 7.3)in the above equation, after simplification give rise to,

$$\mathbf{P1} = -\log(\sqrt{2\pi}\sigma) - 0.5 + \sum_{p=0}^{|\mathbf{c}^{i,a}|} \mathbb{P}(e_p)\log(\mathbb{P}(e_p)) \tag{C.1}$$

Looking at **P2**, the expression can be split as a summation of two terms:

$$\begin{aligned}\mathbf{P2} \quad = \quad & \int_{z\in z_\tau^{i,a}} \sum_{c\in\mathbf{c}^{i,a}} \mathbb{P}(c,z)\log(\mathbb{P}(c))dz \\ & + \int_{z\in z_\tau^{i,a}} \sum_{c\in\mathbf{c}^{i,a}} \mathbb{P}(c,z)\log(\mathbb{P}(z))dz\end{aligned}$$

We denote the first and second terms of **P2** as $\mathbf{P2}_a$ and $\mathbf{P2}_b$ respectively. By using Equation 7.8 $\mathbf{P2}_a$ can be written as

$$\mathbf{P2}_a = \sum_{p=0}^{|\mathbf{c}^{i,a}|} \mathbb{P}(e_p)\log(\mathbb{P}(e_p)) \int_{z\in z_\tau^{i,a}} \mathbb{P}(z|e_p)dz$$

As $\sum_{c\in\mathbf{c}^{i,a}} \mathbb{P}(c,z)$ reduces to $\mathbb{P}(z)$(marginalization) . $\mathbf{P2}_b$ can be expressed as,

$$\mathbf{P2}_b = \int_{z\in z_\tau^{i,a}} \mathbb{P}(z)\log(\mathbb{P}(z))dz$$

Finally $\mathbf{P1}-\mathbf{P2}$ and using fact the $\int_{z\in z_\tau^{i,a}} \mathbb{P}(z|e_p)dz$ is one, obtains the desired result:

$$\begin{aligned}\mathbf{I}[\mathbf{c}^{i,a}; z_\tau^{i,a}] \quad = \quad & -\int_{z\in z_\tau^{i,a}} \mathbb{P}(z)\log(\mathbb{P}(z))dz \\ & + \ -\log(\sqrt{2\pi}\sigma) - 0.5\end{aligned} \tag{C.2}$$

□

APPENDIX D

PROOF OF CONSENSUS THEOREM (THEOREM 1 IN CHAPTER 7)

Proof. We begin by taking the negative log of Equation 7.13, then the equation can be expressed as,

$$\mathbb{LP}_{m_j^i}(k+1) = \sum_{\hat{n} \in \mathbb{N}_k^i \cup i} a_{i,\hat{n}}(k) \left(\mathbb{LP}_{m_j^{\hat{n}}}(k) \right) + \mathbb{L}u_e^{i,j}(k) \tag{D.1}$$

Where $\mathbb{LP}_{m_j^i}(k+1)$ and $\mathbb{L}u_e^{i,j}(k)$ are given by $-\log\left(\mathbb{P}_{m_j^i}(k+1)\right)$ and $-\log\left(u_m(m_j^i, \mathbf{x}_k^i, \mathbf{z}_k^i)\right)$ respectively.

If we stack the variables $\mathbb{LP}_{m_j^i}(k)$ and $\mathbb{L}u_e^{i,j}(k)$ for every robot index $i \in \{1, ..., N_R\}$ into a column vectors denoted as $\mathbb{LP}_{m_j}[k]$ and $\mathbb{L}\mathbf{u}_e^j[k]$ respectively, then Equation D.1 can be converted to the following matrix equation:

$$\mathbb{LP}_{m_j}[k+1] = \mathbb{A}[k]\mathbb{LP}_{m_j}[k] + \mathbb{L}\mathbf{u}_e^j[k] \tag{D.2}$$

Where $\mathbb{A}[k]$ is the adjacency matrix[56] of the time varying robot interaction graph $\mathbb{G}(k)$.

Now if we define $\Phi_{\mathbb{A}}[k, k_0] = \mathbb{A}[k]\mathbb{A}[k-1]\cdots\mathbb{A}[k_0]$ then at each time step k the information dynamics can be described by

$$\mathbb{LP}_{m_j}[k+1] = \Phi_{\mathbb{A}}[k, 0]\mathbb{LP}_{m_j}[0] + \sum_{d \in \mathbf{d}} \Phi_{\mathbb{A}}[k, d]\mathbb{L}\mathbf{u}_e^j[d] \tag{D.3}$$

From [76, Theorem 1] we obtain that, if Assumption 1 holds and $\mathbb{A}[k]$ is doubly stochastic matrix for each k then,

$$\lim_{k\to\infty} \Phi_{\mathbb{A}}[k, k_0] = \frac{1}{N_R}\mathbf{1}\mathbf{1}^T$$

$\mathbf{1}$ is a column vector of N_R ones. Assumption 2 and choosing $a_{i,\hat{n}}(k) = 0.5$ whenever $\mathbb{N}_k^i$ is non empty ensure that $\mathbb{A}[k]$ is doubly stochastic for all k.

Taking lim of Equation D.3 and using the above result yields,

$$\begin{aligned} \lim_{k\to\infty} \mathbb{LP}_{m_j}[k] \quad = \quad & \mathbf{1}\left(\frac{1}{N_R}\mathbf{1}^T\mathbb{LP}_{m_j}[0]\right) \\ & + \mathbf{1}\left(\sum_{d \in \mathbf{d}} \frac{1}{N_R}\mathbf{1}^T\mathbb{L}\mathbf{u}_e^j[d]\right) \end{aligned}$$

Applying $\frac{1}{N_R}\mathbf{1}^T$ on a column vector with N_R elements is equivalent to taking the arithmetic mean of the elements of the corresponding column vector. Therefore, if we denote $< \cdot >$ as the arithmetic mean operator, then we can rewrite the above equation in compact form as,

$$\lim_{k\to\infty} \mathbb{L}\mathbb{P}_{m_j}[k] = \mathbf{1}\left\langle \mathbb{L}\mathbb{P}_{m_j}[0]\right\rangle + \mathbf{1}\sum_{d\in\mathbf{d}}\left\langle \mathbb{L}\mathbf{u}_e^j[d]\right\rangle$$

Assumption 3 is required to ensure the convergence of the sum $\sum_{d\in\mathbf{d}}\left\langle \mathbb{L}\mathbf{u}_e^j[d]\right\rangle$. Reducing the above equation for the i^{th} robot yields,

$$\lim_{k\to\infty} \mathbb{L}\mathbb{P}_{m_j^i}(k) = \left\langle \mathbb{L}\mathbb{P}_{m_j}[0]\right\rangle + \sum_{d\in\mathbf{d}}\left\langle \mathbb{L}\mathbf{u}_e^j[d]\right\rangle$$

Finally, by taking the negative exponential of the above equation we obtain the desired result:

$$\lim_{k\to\infty} \mathbb{P}_{m_j^i}[k] =< \mathbb{P}_{m_j}[0] >_{gm} \cdot \prod_{d\in\mathbf{d}} < \mathbf{u}_j[d] >_{gm}$$

□

AGREEMENT

I am pleased to have worked with and gained permission from my co-authors to include the material in this dissertation from my collaborators Dr. Spring Berman, Dr. Sean Wilson, Karthik Elamvazhuthi, and Zahi Kakish.