DISSERTATION

USING LOCALLY OBSERVED SWARM BEHAVIORS TO INFER GLOBAL FEATURES OF HARSH ENVIRONMENTS

Submitted by

Megan R. Emmons

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Fall 2021

Doctoral Committee:

Advisor: Anthony A. Maciejewski
Co-Advisor: Edwin K. P. Chong

Charles W. Anderson
Peter M. Young

ABSTRACT

USING LOCALLY OBSERVED SWARM BEHAVIORS TO INFER GLOBAL FEATURES
OF
HARSH ENVIRONMENTS

Robots in a swarm are programmed with individual behaviors but then interactions with the environment and other robots produce more complex, emergent swarm behaviors. A partial differential equation (PDE) can be used to accurately quantify the distribution of robots throughout the environment at any given time if the robots have simple individual behaviors and there are a finite number of potential environments. A least mean square algorithm can then be used to compare a given observation of the swarm distribution to the potential models to accurately identify the environment being explored. This technique affirms that there is a correlation between the individual robot behaviors, robot distribution, and the environment being explored. For more complex behaviors and environments, there is no closed-form model for the emergent behavior but there is still a correlation which can be used to infer one property if the other two are known. A simple, single-layer neural network can replace the PDE and be trained to correlate local observations of the robot distribution to the environment being explored. The neural network approach allows for more sophisticated robot behaviors, more varied environments, and is robust to variations in environment type and number of robots. By replacing the neural network with a simulated human rescuer who uses only locally observed velocity information to navigate a disaster scenario, the impact of fundamental swarm properties can be systematically explored. Further, the baseline swarm resilience can be quantified. Collectively, this development lays a foundation for using minimalist swarms, where robots have simple motions and no communication, to achieve collective sensing which can be leveraged in a variety of applications where no other robotic solutions currently exist.

ACKNOWLEDGEMENTS

A work of this magnitude is the culmination of many years of research and included a full spectrum of emotions from high points of excitement, curiousity, and joy to admittedly low points of frustration and uncertainty. I am so proud of this final work and know I would not have been able to complete my dissertation without the support and confidence of an entire community at CSU and especially in the ECE department. Katya, Karen, Courtney, Alauna, and Dezarai were always so patient and helped me navigate the obstacle course of class registrations, graduate forms, time sheets, and general life stresses. My labmates turned friends were also instrumental in my research. They commiserated with the struggles of graduate school and celebrated the small victories - usually with tea in both cases to keep me moving forward!

This work would absolutely have not been possible without the support of my wonderful dissertation committee. Drs. Edwin Chong, Chuck Anderson, and Peter Young provided valuable insight into technical challenges I faced during my research but also served as role models who treated each student and advisee with respect. Thank you for taking the time to understand my research goals – and for accepting the marmot picture which ended each of my presentations with a bit of personality!

I especially want to thank my advisor, Dr. Anthony Maciejewski for his guidance throughout my doctoral journey. I came to CSU with an idea somewhat outside the ongoing research efforts but he promised his support and never waivered, even when I modified the name of his lab with a note taped to the door! He was always mindful of my life goals and I appreciate his mentorship which helped me grow professionally in a direction to support all of my ambitions.

DEDICATION

Although I highly doubt this document could have been created without mountains and chai, I know it would not have been possible without the love and support of my friends and family. I dedicate this dissertation to each of them with a heartfelt THANK YOU.

Thank you for understanding when I chose my computer over other activities in the moment.

Thank you for your confidence in my abilities when I was uncertain.

Thank you for being supportive of my dreams.

Thank you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Robots have long captured the human imagination. From ancient mythology to science fiction and modern pop culture, many people have come to envision robots as highly intelligent or super strong human assistants. Indeed, the term "robot" was first introduced in a 1920 play by Czech writer Karel Čapek to describe a class of artificial human workers in his futuristic play "R.U.R.". It was a modification of the Slavic word "robota" which translated to forced laborer and was used to describe the indentured peasant working class of that time [1].

Today's robots have expanded beyond Čapek's early vision in some ways as they can be found assisting people in a much wider range of domains from medical surgeries to self-driving vehicles. On the other hand, fundamental challenges still limit the versatility of robots. In particular, one important topic of interest in the robotics community which underlines many other applications is the navigation of harsh environments. Collapsed buildings, old mines, and distant planets are all dangerous environments for a human to explore but the rough, unpredictable terrain also makes these locations extremely challenging for robots. One significant example is the Fukushima nuclear reactor site. At least seven robots failed in 2015 and were abandoned in the reactors [2]. These robots were often tethered with communication cables and directed by a human driver yet were still unable to safely navigate the terrain.

In response to the 2011 disaster at Fukushima, the Defense Advanced Research Projects Agency (DARPA) created the DARPA Robotics Challenge (DRC) and hosted the first trial in 2013 [3]. DARPA recognized both the danger of disaster sites for human aid workers as well as the shortcomings of current robotics technology. The DRC was designed to incentivize advancements in robot disaster response by presenting an obstacle course of potential disaster scenarios and offering a $3.5 million prize to the best robot design. During the challenge, robots needed to complete comparatively simple challenges like climb a small set of steps, walk across cement blocks, rotate a valve, open a door, etc. The first trials were held in 2013 and the finals were held in 2015.

Although two winning teams were ultimately crowned, the course proved too difficult for many of the challengers. Some robots simply fell over when attempting to climb a stair. Others ended up receiving incorrect sensor input and collapsing. The winning team with the fastest time ended up manually programming commands for every centimeter of the course into the robot and still required an agonizing 45 minutes to finish. Suffice to say, the approach is not a reliable solution in real disaster scenarios where the environment is not known.

Fukushima and the DRC demonstrate the need for robots to navigate or explore rough terrain but also the limitations of current disaster robots. To improve disaster robots, many researchers tend to focus on developing increasingly sophisticated robots but these robots may not be the best solution. A single robot can only explore so much terrain in a given amount of time whereas a team of robots can explore a much larger area. Similarly, a single robot is not very robust. Sophisticated robots often rely on complex localization strategies which fail in GPS-denied environments or when communication fails. Single robots for disaster scenarios also depend on a wide range of expensive sensors to ensure proper operation. When those sensors start to malfunction, the robot fails as was often the case in Fukushima. Adding components to the single robot necessarily increases the cost of the robot, the power requirements, and also the overall size which can make maneuvering in an unstable environment more treacherous. Further, if that single robot fails, the entire mission fails. By contrast, if one robot in a team of ten fails, the remaining robots can continue the mission. These observations are the principal motivation for using a large team or swarm of inexpensive robots to explore unknown, potentially harsh environments.

Swarms are a relatively new focus area within the robotics community but take inspiration from well-observed biological systems. Like their biologic counterparts, robots in a swarm are governed by local rules but frequent interactions with other robots and the environment generate a more complex, emergent behavior. Sophisticated foraging strategies and complex colony construction by ants reveal just a few of the potential advantages of emergent behavior because these actions are accomplished in a distributed and robust manner. Similar emergent behaviors in artificial systems will be extremely useful in many exploration tasks, particularly in harsh environments with a high

probability of robot failure. Despite the many important applications, research in robotic swarms has not yet matured to the point of reliable, real-world deployment.

One of the biggest challenges in swarm design is correlating individual robot behaviors with the collective behavior of the swarm which emerges over time for a given environment. In many cases, the individual robot behaviors are known, the environment being explored is unknown, and only a subset of the emergent behavior can be observed. Despite the limited information, local observations of the emergent swarm behavior can be used to identify features of the environment. For example, consider rafting a river. Each water particle is like a member of the river "swarm" while the flow of the river is the emergent behavior as water particles interact with each other and obstacles in the river. Observing the upstream flow allows an experienced person to predict downstream characteristics of the river such as rocks or other obstructions. As with the river, emergent behavior in robotic swarms can be observed from physical implementations or simulated with computer models but it requires the equivalent of experience to properly correlate observations of the emergent behavior with the desired environmental features.

This dissertation presents foundational work in using observations of emergent behavior in a simulated swarm of robots to predict environmental features. The robustness of a minimalist swarm is also quantified to establish a baseline metric for potential disaster rescue scenario applications. Chapter 2 presents our first work where partial differential equations were used to model the swarm distribution in simple 1D and 2D environments. We verified an observation of the swarm distribution in a central portion of the environment could be used to identify the entire environment by using a least squared error between the observed density and the set of potential PDE models. We extend the correlation between emergent behavior and unknown environments in Chapter 3. In Chapter 3 we train a single-layer, soft max neural network to predict which wall most likely contains a door given the central distribution of robots. The neural network increases the robustness of our methodology - allowing for variation in the simulated door placement with respect to the training data as well as a drastic 90% loss in the number of robots exploring the environment.

With a partial demonstration of swarm robustness verified through use of a simple neural network, Chapter 4 extends the robustness analysis. We quantify the impact of local swarm variables on the utility of the emergent behavior for a simulated rescuer attempting to locate a lone survivor in an unknown disaster environment. The simulated rescuer uses a minimalist interaction by only relying on the locally observable emergent swarm velocity and yet reliably locates the survivor despite significant changes in initial swarm size, environment shape, and local swarm parameters. The swarm can again undergo catastrophic failure but the distributed nature of the swarm allows the rescuer to successfully locate the survivor when as few as 12 of the original 300 robots are still functional. With these promising results, Chapter 5 summarizes the demonstrated advantages of swarms from our work and highlights important areas for future study to help swarms bridge the gap from simulation to real-world deployment.

Chapter 2

Modelling Emergent Swarm Behavior Using Continuum Limits for Environmental Mapping [1]

2.1 Summary

Robotic swarms are comprised of simple, individual robots but can collectively accomplish complex tasks through frequent interactions with other robots and the environment. One pertinent objective for swarms is mapping unknown, potentially hazardous environments. We show that even without communication or localization, the emergent behavior of a swarm observed at one area can be used to infer the presence of obstacles in an unknown environment. The main body of this work focuses on how partial differential equation (PDE) models of emergent swarm behavior can be derived by applying continuum limits to approximate discrete-time rules for individual robots in continuous-time. We illustrate our approach by demonstrating how obstacles can be located by comparing swarm observations to a base library of PDE models. As supported in this work, the PDE models accurately capture identifying characteristics of the emergent behavior and are solved in a few seconds allowing for fast feature identification.

2.2 Introduction

Swarms are a relatively new focus area within the robotics community but take inspiration from well-observed biological systems. Like a colony of ants, a robotic swarm is comprised of simple individual robots that can collectively accomplish complex tasks via frequent interactions with other robots and the environment [4]. Multi-robot systems provide an increased level of redundancy and may outperform individual, more sophisticated robots by working cooperatively to accomplish a given task [5]. Leveraging the number of robots, swarms have a variety of applica-

[1]Published in *Proceedings of the IEEE International Conference on Control and Automation*, June 2018, pp. 86–93.

tions in surveillance, medical treatment, and exploration [6]. Swarms have additional advantages in exploration; very simple, inexpensive robots can quickly reveal important features of an unknown environment, as shown in this paper.

Even in a worst-case scenario where robots within a swarm are reduced to random motion with no communication, their distribution can reveal important environmental information. The shape of an environment influences how robots are dispersed through the domain so observing the density of robots at one location can provide information about features throughout the environment, particularly the presence of obstacles. By way of analogy, the position of downstream blockages in a river can be inferred by observing the flow of water upstream. The identification requires no communication between robots and only local, rather than global, knowledge—making the system fully scalable. Identifying features becomes a matter of developing appropriate models and matching observed swarm behavior with the corresponding model. During the identification process, the emergent behavior of a swarm is more informative than individual behaviors, just as the river flow is more significant than individual water droplets.

For physical implementations, individual robots are programmed so local behaviors are known. The emergent behavior can be observed from a physical implementation or modeled through simulation but both methods are extremely time consuming, especially as swarm and environment size increase. In this paper, we propose a methodology to map discrete-time, probabilistic behaviors of individual robots to a partial differential equation (PDE) model of the emergent behavior. The PDE model is quickly solved to determine the emergent behavior at any desired time and position and thus allows for fast feature identification in an unknown environment.

This paper is organized as follows. In Section 3.3, a brief summary of related work is presented before we introduce the continuum limit methodology in Section 2.4. A detailed derivation of the PDE model for a one-dimensional (1D) random walk scenario is presented first to provide a theoretical foundation. The 1D model also allows for more intuitive visualization and validation of the proposed methodology. We then apply the same methodology to a two-dimensional (2D) environment. Validation of both 1D and 2D models is given in Section 2.5. In Section 2.6 we

provide a preview into how the PDE models can be used to infer the boundary-types of a simulated environment. The preview includes a illustrative simulated scenario. Finally, we conclude the work in Section 3.7 and present some directions for future work.

2.3 Related Work

A detailed review of early research in swarm robotics is provided by Navarro and Matía [4] where they also present important characteristics of swarm robotics. Citing Şahine [7], Navarro and Matía consider a swarm to be comprised of many autonomous robots with only local sensing capabilities. Another key feature is scalability of the system. The proposed methodology herein encompasses the swarm definition by assuming robots with no localization or communication.

Many swarm researchers have focused on designing control strategies to produce a desired emergent behavior. Strategies include the use of event-triggered controllers [8], chemical reaction models [9], [10], and potential fields [11]. These strategies all focus on developing rules for individual robots, in essence taking a top-down approach by dictating local rules to produce a desired final state.

Some researchers have linked partial differential equation (PDE) models to emergent behavior of the swarm. Elamvazhuthi and Berman [12] use a set of advection-diffusion-reaction PDEs to model swarm behavior but still take a top-down approach. The parameters of the PDE are optimized to achieve a desired emergent behavior and then the individual robots behave according to the tuned PDE.

This paper proposes a reverse approach. Rather than imposing a high-level control strategy and influencing individual robot actions to generate a desired distribution, we propose a methodology to map individual robot actions to the natural emergent behavior generated in varying environments. Berger et al. [13] similarly take a bottom-up approach to swarm modeling by proposing the use of compressive subspace learning to identify emergent behavior. The subspace learning still lacks a direct correlation between individual robot actions and the emergent behavior. It also does not provide information about environmental features and requires extensive computations.

The methodology proposed herein provides a direct mapping between local and emergent behaviors for varying environments. A PDE model is obtained based on local behavior rules. The boundary conditions of the PDE model encode environmental features—namely open-space or obstruction. The model derivation is based on the work of Zhang et al. [14] who apply continuum limits to derive PDE models for large-scale wireless networks.

2.4 Continuum Limit Methodology

2.4.1 Derivation of 1D Random Walk Model

To fully illustrate the proposed methodology, we start with a one-dimensional (1D) environment where robots can only move left or right. In this scenario, the robots are performing a random walk. The limited capability of the robot model demonstrates how even simple robots can be used to map unknown environments and represents a worse-case scenario in the spectrum of robot capabilities. The environment itself is composed of N bins evenly distributed in a one-dimensional line, $D = (0, 1)$, where N specifies the desired spatial resolution of the environment. Two additional bins, denoted $n = 0$ and $n = N + 1$ for the left and right-most bins respectively, map to the boundaries of the interval. There are M robots distributed in the N interior bins. At every discrete-time step, each robot randomly chooses to move left or right one bin with probability P_L or $(1 - P_L)$ respectively. As a result, the number of robots in each bin, the *bin occupancy*, varies over time according to a random process.

Let $\mu_{n,k}$ be the occupancy of bin n at time k. We wish to characterize the dynamics of $\mu_{n,k}$ over time. The value of $\mu_{n,k+1}$ depends on the number of robots moving from adjacent bins and hence is a random function of $\mu_{n-1,k}$ and $\mu_{n+1,k}$. To precisely characterize the evolution of bin occupancies over time, we use the method in [14] and define $\vec{\mu}_k = [\mu_{1,k}, \ldots, \mu_{N,k}]$ as the vector of bin occupancies at time k. Assume that for each robot, the choice to move left or right is independent over robots, time, and also of $\vec{\mu}_k$. Taking the conditional expectation of $\mu_{n,k+1}$ given $\vec{\mu}_k$, we obtain

$$E[\mu_{n,k+1}|\vec{\mu}_k] = P_L\mu_{n+1,k} + (1 - P_L)\mu_{n-1,k}. \tag{2.1}$$

The right-hand side of (2.1) has the form $f(\vec{\mu}_k)$. From this expression, we can write the mean-field equation, similar to the process in [14],

$$\vec{m}_{k+1} = f(\vec{m}_k), \tag{2.2}$$

the n^{th} component of which is

$$m_{n,k+1} = P_L m_{n+1,k} + (1 - P_L) m_{n-1,k}. \tag{2.3}$$

For simplicity, it is assumed robots move left or right with equal probability so

$$P_L = \frac{1}{2}, \tag{2.4}$$

and the mean occupancy of bin n at time $k + 1$ reduces to

$$m_{n,k+1} = \frac{1}{2}[m_{n+1,k} + m_{n-1,k}]. \tag{2.5}$$

The mean change in the occupancy of bin n between discrete-time steps k and $k + 1$ can be expressed as

$$\Delta m_n = \frac{1}{2}[m_{n+1,k} + m_{n-1,k}] - m_{n,k}. \tag{2.6}$$

Equation (2.6) represents the discrete-time change in mean bin occupancy for interior bins along a one-dimensional line. To map the discrete-time model to continuous time, the following substitutions are made:

$$\begin{aligned} n \pm 1 &\rightarrow s \pm \Delta s \\ k + 1 &\rightarrow t + \Delta t. \end{aligned} \tag{2.7}$$

The continuous-time model for a one-dimensional random walk becomes

$$\Delta m(s,t) = \frac{1}{2}[m(s + \Delta s, t) + m(s - \Delta s, t)] - m(s,t). \tag{2.8}$$

The continuous-time model can be represented by a partial differential equation by first performing a second-order Taylor Series expansion where

$$\begin{aligned} m(s \pm \Delta s, t) = \quad & m(s,t) \pm \tfrac{\partial m(s,t)}{\partial s}\Delta s \\ & + \tfrac{1}{2}\tfrac{\partial^2 m(s,t)}{\partial s^2}\Delta s^2 + o(\Delta s^2). \end{aligned} \tag{2.9}$$

Substituting (2.9) into (2.8) and performing algebraic simplifications, the continuous-time model then becomes

$$m(s, t+\Delta t) - m(s,t) = \frac{1}{2}\frac{\partial^2 m(s,t)}{\partial s^2}\Delta s^2 + o(\Delta s^2). \tag{2.10}$$

Now let $\Delta t = \Delta s^2$ and divide both sides of (2.10) accordingly to reach

$$\frac{m(s, t+\Delta t) - m(s,t)}{\Delta t} = \frac{1}{2}\frac{\partial^2 m(s,t)}{\partial s^2} + \frac{o(\Delta s^2)}{\Delta s^2}. \tag{2.11}$$

Taking the limit as $\Delta t \to 0$, the continuous-time model converges to the standard heat equation

$$\frac{\partial m(s,t)}{\partial t} = \frac{1}{2}\frac{\partial^2 m(s,t)}{\partial s^2} \tag{2.12}$$

for interior bins. The interior model of (2.12) becomes increasingly accurate as the variables of interest, namely environment size (N) and number of robots (M), approach infinity (see [14]). To fully define the PDE solution, it is necessary to determine boundary conditions and define corresponding initial conditions.

Boundary Conditions for 1D Random Walk

Equation (2.12) describes the one-dimensional random walk scenario for internal bins, $n \in [1, N]$, in continuous time but does not address the boundary conditions which are represented in discrete time by bins 0 and $N+1$. To derive continuous-time rules for the boundary bins, it is important to note that all internal bins must follow the rule described by (2.5). Let general bin a represent a boundary bin, b the adjacent internal bin, and c the internal bin adjacent to b.

Two different types of boundaries are considered: sinks and walls, roughly corresponding to open exits and obstacles, respectively. When a robot chooses to move into bin a where a is a sink, the robot is removed from the system. The discrete-time model for the population of bin b at time $k+1$ is therefore

$$m_{b,k+1} = \frac{1}{2} m_{c,k} \tag{2.13}$$

but the internal rule requires

$$m_{b,k+1} = \frac{1}{2} m_{a,k} + \frac{1}{2} m_{c,k}. \tag{2.14}$$

The two equations can both be satisfied by choosing

$$m_{a,k} = 0 \; \forall k. \tag{2.15}$$

Physically, the discrete rule means the sink boundary bin should have an occupancy of zero at every discrete-time step which agrees with the initial intent of the model–robots that enter a sink are instantly removed. Extending the discrete rule into continuous time results in a Dirichlet boundary condition,

$$m(a,t) = 0 \; \forall t, \tag{2.16}$$

for the PDE in (2.12).

Discrete- and continuous-time rules for a wall boundary are a bit more complicated but follow the same general process. A robot that chooses move 'into' a wall boundary actually stays in its current bin for that time step. The population of bin b at time $k+1$ is therefore represented in discrete time as

$$m_{b,k+1} = \frac{1}{2} m_{b,k} + \frac{1}{2} m_{c,k} \tag{2.17}$$

Comparing to (2.14), both conditions are satisifed by choosing

$$m_{a,k} = m_{b,k} \; \forall k. \tag{2.18}$$

Noting that bin b is adjacent to bin a and using the substitutions from (2.7), the continuous-time rule becomes

$$m(a,t) - m(a - \Delta s, t) = 0 \; \forall t. \tag{2.19}$$

Dividing both sides by Δs and taking the limit as $\Delta s \to 0$, a Neumann boundary condition,

$$\frac{\partial m(a,t)}{\partial s} = 0 \; \forall t, \tag{2.20}$$

is obtained for the PDE model.

Initial Conditions for 1D Random Walk Scenario

With the internal PDE and boundary conditions determined, only the initial conditions are needed to fully define the PDE model. Assume that M robots are initially distributed in the N internal bins so bin j has occupancy determined by

$$m_{j+1} = \frac{\pi * M}{2(N+1)} \sin(\frac{\pi * j}{N+1}), \; j = 0, 1, \ldots, N-1 \tag{2.21}$$

where the actual population is rounded to the nearest integer. This initial distribution approximating a half sine wave was chosen as a simple example to validate the PDE model but could correspond to a physical system where the majority of robots are inserted far away from potential obstacles (boundaries in this illustration). To map the occupancy of robots in bins $1 \to N$ to the continuous time domain, $D = (0, 1)$, the PDE should have initial condition

$$m(s, 0) = \sin(\pi s). \tag{2.22}$$

2.4.2 Extension of Methodology to 2D Random Walk

Though the 1D case illustrates our proposed methodology, 2D environments are a more realistic representation for our domain of interest. When moving from 1D to 2D, the same general steps are applied. Now N bins are evenly distributed in a rectangle, $\mathbb{R}^{N_x \times N_y}$ where $N_x = \{1, 2, \ldots, X\}$

and $N_y = \{1, 2, \ldots, Y\}$, in order to designate the desired spatial resolution. Two additional bins are required for each row, denoted as 0 and $X + 1$, and column, 0 and $Y + 1$, to define the interval boundaries. At every discrete-time step, each of the M robots now randomly decides to move either up, right, down, or left one bin with probabilities P_U, P_R, P_D, or P_L respectively. To be a proper probability model, an additional constraint where the sum of the probabilities for each of the four different moves is equal to one applies. The mean occupancy, m, of interior bin $[i, j]$ at time $k+1$ is once again the sum of robots in neighboring bins which choose to move into bin $[i, j]$, mathematically expressed as

$$\begin{aligned} m_{[i,j],k+1} &= P_U m_{[i,j-1],k} + P_R m_{[i-1,j],k} + \\ & \quad P_D m_{[i,j+1],k} + P_L m_{[i+1,j],k}. \end{aligned} \tag{2.23}$$

For simplicity, we assume that the robots move to one of the four available bins with equal probability:

$$P_U = P_R = P_D = P_L = \frac{1}{4}. \tag{2.24}$$

Using a similar methodology as the 1D case, the mean occupancy of bin $[i, j]$ at time $k+1$ therefore reduces to

$$\begin{aligned} m_{[i,j],k+1} &= \tfrac{1}{4}(m_{[i,j-1],k} + m_{[i-1,j],k} + \\ & \quad m_{[i,j+1],k} + m_{[i+1,j],k}). \end{aligned} \tag{2.25}$$

The change in the mean occupancy of bin $[i, j]$ between time k and $k + 1$ can be expressed as

$$\begin{aligned} \Delta m_{[i,j]} &= \tfrac{1}{4}(m_{[i,j-1],k} + m_{[i-1,j],k} + m_{[i,j+1],k} + \\ & \quad m_{[i+1,j],k}) - m_{[i,j],k}. \end{aligned} \tag{2.26}$$

Equation (2.26) represents the mean discrete-time change in bin population for interior bins in a two-dimensional rectangle. To map the discrete-time model to continuous time, the following substitutions are made much as in the 1D case:

$$\begin{aligned} [i \pm 1, j \pm 1] &\rightarrow [x \pm \Delta x, y \pm \Delta y] \\ k+1 &\rightarrow t + \Delta t. \end{aligned} \tag{2.27}$$

The continuous-time model for the internal, two-dimensional random walk becomes

$$\begin{gathered} m([x,y],t+\Delta t) - m([x,y],t) = \\ \tfrac{1}{4}m([x,y-\Delta y],t) + \tfrac{1}{4}m([x-\Delta x,y],t) + \\ \tfrac{1}{4}m([x,y+\Delta y],t) + \tfrac{1}{4}m([x+\Delta x,y],t) - \\ m([x,y],t). \end{gathered} \tag{2.28}$$

By using a second-order Taylor Series expansion to approximate the continuous-time model of (2.28), performing algebraic simplification, defining $\Delta t = \Delta x^2$ and $\Delta t = \Delta y^2$ so both sides of the equation can be divided appropriately, and taking the limit as $\Delta t \rightarrow 0$, the continuous-time model converges to the standard multi-dimensional heat equation

$$\frac{\partial m(\vec{x},t)}{\partial t} = \frac{1}{4}\nabla^2 m(\vec{x},t) \tag{2.29}$$

where in this scenario

$$\vec{x} = [x,y] \in (0,1)^2. \tag{2.30}$$

Boundary Conditions in 2D

As for the 1D scenario, the key to boundary conditions is ensuring all internal bins follow the same rule as described by (2.25). The discrete-rule is satisfied for sink boundaries by again imposing a Dirichlet boundary condition. Similarly, the discrete-rule is satisfied for a wall boundary by imposing a modified Neumann boundary condition,

$$\vec{d} \cdot \nabla_x m(a,t) = 0 \; \forall t, \tag{2.31}$$

where $\vec{d}$ represents the direction of the boundary bin with respect to the internal bin of interest.

Initial Conditions in 2D

For the 2D random walk simulation, M robots are initially distributed in the N internal bins so bin $[i, j]$ has occupancy determined by

$$\begin{aligned} m_{i,j} &= M\tfrac{\pi}{2(Y+1)}\sin(\tfrac{\pi*i}{Y+1})\tfrac{\pi}{2(X+1)}\sin(\tfrac{\pi*j}{X+1}), \\ & i = 0, 1, \ldots, Y-1,\ j = 0, 1, \ldots, X-1 \end{aligned} \tag{2.32}$$

with $m_{i,j}$ rounded to the nearest integer. Mapping the discrete bin occupancy to continuous time as in the 1D scenario reveals the PDE should have initial conditions:

$$m([i, j], 0) = \sin(\pi * i)\sin(\pi * j). \tag{2.33}$$

2.5 Validation of Partial Differential Equation Models for Emergent Behavior

2.5.1 Overview

To evaluate the effectiveness of the proposed methodology for deriving a PDE model of emergent swarm behavior, scripts were written in MATLAB to simulate a swarm of robots exploring an environment using simple random walk motion. Robots are distributed in finite-sized bins evenly spaced throughout the environment. During every discrete-time step, each robot moves to a neighboring bin with appropriate probability. The MATLAB-based simulation records the population of all bins for each discrete-time step. Similarly, the corresponding PDE model is solved to produce a solution showing the time evolution of robot occupancy throughout the environment.

2.5.2 Scaling for 1D Random Walk Scenarios

We again start with the 1D model for validation before expanding into 2D. To simulate the 1D random walk, M robots are distributed across N bins according to the initial conditions of (2.21).

The corresponding internal PDE model is given by (2.12). Appropriate scaling factors are required for time, space, and population to accurately compare the simulation and corresponding PDE.

For the 1D line, a total of $N + 2$ bins are used in the simulation to account for boundary bins. Each bin is represented by its mid-point. The left-most midpoint corresponds to the left boundary of the PDE, namely 0. Similarly, the midpoint of the right-most bin is associated with the right PDE boundary. Hence, to correlate the simulation which has a variable number of bins to the fixed $(0, 1)$ PDE interval, the simulation is scaled by

$$\Delta s = \frac{1}{N+1}. \tag{2.34}$$

Increasing N improves the spatial resolution because more bins are mapped to the fixed $(0, 1)$ interval. With the spatial scaling specified by the user, temporal scaling is determined by $\Delta t = \Delta s^2$ as previously described.

The final scaling is to ensure comparable bin occupancies. The simulation is normalized by dividing the occupancy of each bin by the total number of robots in a simulation - namely M - so from (2.21) has a maximum possible value of

$$z_{\text{max}} = \frac{\pi}{2(N+1)}. \tag{2.35}$$

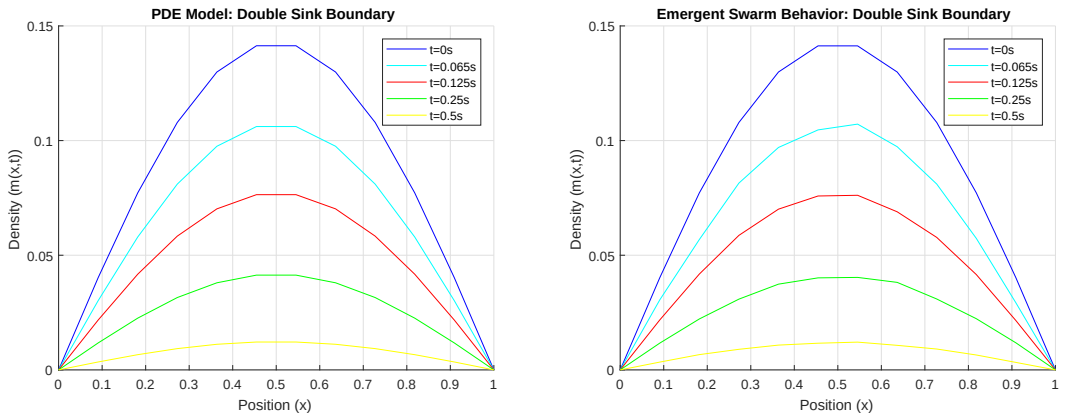
The PDE is initialized according to (2.22) which has a maximum value of one so for comparison the PDE model must be multiplied by z_{max} as well. With these scaling factors, a simulation can be compared to its associated PDE model to produce nearly identical bin populations with improved accuracy for increased values of M and N as supported in the following section.

2.5.3 Results of 1D Random Walk

To evaluate the effectiveness of the derived PDE model, the occupancy of each bin was plotted per time step and compared to the corresponding PDE solution at equivalent time samples. Both techniques generate a surface plot, showing the bin occupancy at every position for each time,

so slices of each surface were taken at equivalent times for better comparison. Using the scaling described in the previous section, the simulation and PDE generate very similar results. The time required to generate simulation results depends on the number of robots and the number of bins whereas the PDE solution is independent of both factors and was consistently solved in less than 0.8 seconds.

Running a simulation with sink boundaries at both ends to represent an obstacle-free environment and using $N = 10$ bins with $M \approx 10,000$ robots results in the expected behavior for both the PDE model and simulation as shown in Fig. 2.1. Even with this comparatively low resolution environment, the simulation required more than 21 times the computation time of the PDE.

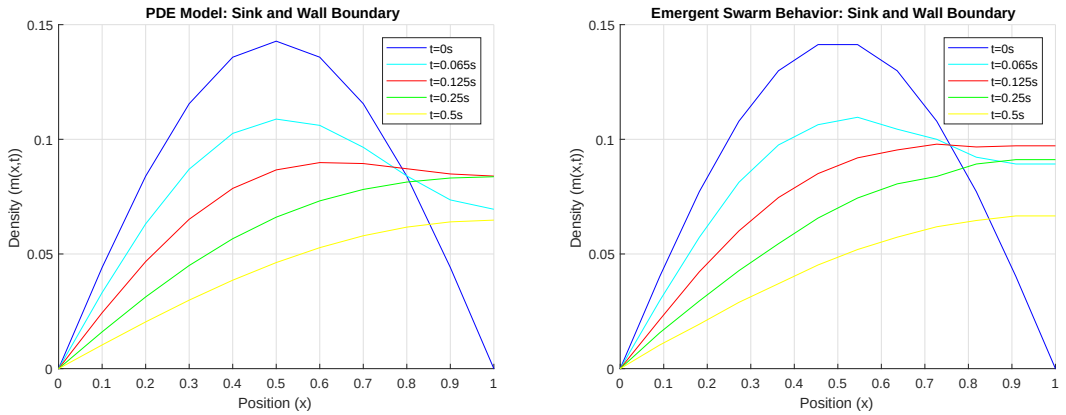


(a) PDE Model:
0.39 sec to generate

(b) Simulation Results:
8.45 sec to generate

Figure 2.1: Comparison of spatial occupancies for a 1D environment at fixed time intervals for the (a) PDE model and (b) discrete, 10-bin simulation with two sink boundaries.

Introducing a wall boundary at bin $n = N + 1$ to represent an obstacle causes a build-up in the occupancy of the adjacent bin throughout the simulation though the sink boundary allows for a steady decrease in bin occupancy as shown in Fig. 2.2. A slight difference in occupancy at the wall boundary between the simulation and PDE is the result of the low number of bins and the scaling of (2.34). In the simulation, robots only occupy N bins but the PDE necessarily considers $N + 1$ bins. As $N \to \infty$, the difference becomes negligible and even with $N = 10$ is minor.



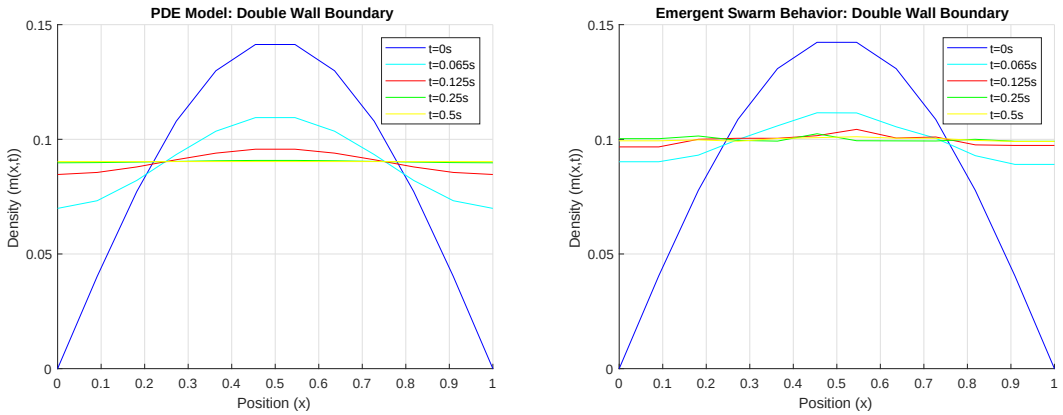
(a) PDE Model:
0.56 sec to generate

(b) Simulation Results:
19.77 sec to generate

Figure 2.2: Comparison of spatial occupancies for a 1D environment at fixed times for the (a) PDE model and (b) discrete, 10-bin simulation for a left sink and right wall boundary.

When both boundaries become obstructed, denoted as wall boundaries, the difference between the PDE and simulation becomes more pronounced because the steady-state solution is non-zero. With two wall boundaries, robots cannot 'escape' the simulation so instead approach a uniform distribution. A uniform distribution for the simulation results in a mean bin occupancy of 0.1 because $N = 10$ bins are available. By contrast, the PDE necessarily settles to $1/(N+1)$ or approximately 0.09, an artifact of the continuum limit mapping. As shown in Fig. 2.3, the PDE model nonetheless presents a distinguishable distribution when compared to the other boundary scenarios.

The overall behavior between the actual simulation and the PDE model for the double wall scenario becomes increasingly similar as the number of bins is increased. Using $N = 50$ increases the spatial resolution of the simulation and decreases the impact of the $N + 1$ artifact; hence, a closer correlation between the emergent behavior of the swarm and the PDE model is observed. The error between the PDE model and emergent behavior, averaged over ten simulation runs, is presented in Fig. 2.4 for the scenario with $N = 10$ as the solid lines and $N = 50$ as the dashed lines at three different sample times. Each 10-bin simulation required less than 47 seconds to



(a) PDE Model:
0.64 sec to generate

(b) Simulation Results:
19.67 sec to generate

Figure 2.3: Comparison of spatial occupancies for a 1D environment at fixed time intervals for the (a) PDE model and (b) discrete, 10-bin simulation with two wall boundaries.

run while the 50-bin simulations each took over 30 minutes. The PDE in both scenarios required approximately 7 seconds to complete.

Figure 2.4: Comparison of error between the PDE model and emergent behavior in simulated 10-bin (solid line) and 50-bin (dashed line) environments for double wall boundary conditions.

Although increasing the spatial resolution improves the model accuracy, a resolution of $N = 10$ bins was used for all three 1D scenarios. This chosen resolution balanced simulation time and model accuracy. Figure 2.5 shows the error between the robot densities for the simulation and PDE model in a 1D environment with double sink boundaries as the number of robots and

the environment resolution are systematically increased. The model accuracy improves as both variables increase but the amount of improvement decreases after $N \approx 10$ bins. It is further worth noting the significant difference in generation time: increasing the resolution from $N = 10$ to $N = 30$ internal bins increased the run-time from 14 seconds to over 15 minutes. Due to the long simulation time required for higher environment resolution and the evidence from Fig. 2.5 showing diminishing improvement in accuracy, the subsequent work maintains smaller N to focus on the effectiveness of the PDE models.

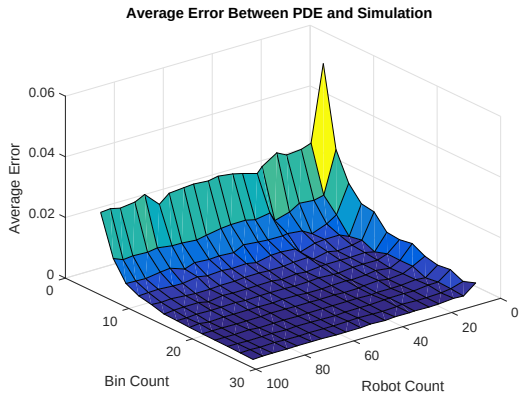
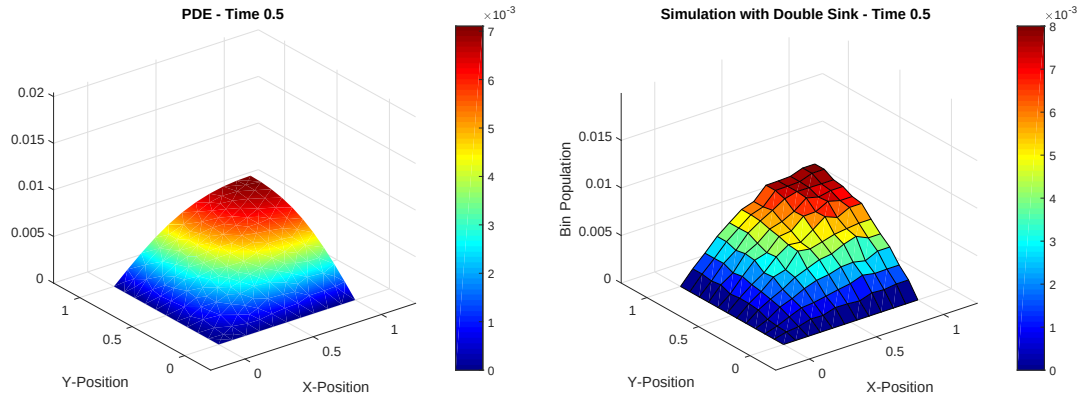


Figure 2.5: The PDE model for a 1D, double sink environment becomes increasingly accurate as the number of bins and number of agents increase.

2.5.4 Evaluation of 2D Model

With the 1D models validated, we again apply a similar strategy to investigate the more representative 2D models. Similar scaling is applied to the 2D PDE model and simulation to compare both techniques graphically. With N bins now distributed in a 2D configuration, time samples of robot distribution are surface plots rather than lines and a greater combination of boundary conditions exist. Systematically varying the boundary conditions for a 10×10 bin environment with $M \approx 100,000$ confirmed that the PDE model again effectively captured the emergent behavior with solutions obtained in seconds rather than minutes or even hours.

As an illustrative example, Fig. 2.6 shows the bin occupancies for a double wall, double sink boundary scenario. Such a scenario could represent a room corner. The PDE model captures the overall shape of the emergent behavior with a slightly lower magnitude at the wall boundaries–a discrepancy once again caused by the low bin count.



(a) PDE Model:
1.7 sec to generate

(b) Simulation Results:
171.5 sec to generate

Figure 2.6: PDE solution and simulation at one time sample with two wall boundaries and two sink boundaries.

2.6 Environmental Mapping from PDE Model

2.6.1 Simple Illustrative Use Case

Thus far, we have shown that PDE models derived from individual random walk behavior of robots using a continuum limit methodology reasonably capture the emergent behavior of a robotic swarm in varying environments. The PDE model is increasingly accurate at modeling the distribution of robots as the simulation resolution increases but the solution is significantly faster through the PDE. In this section, we will show how the PDE models can be used to infer environmental boundary conditions very quickly, but first we propose a relevant physical example.

Assume you are in a mine tunnel that has recently experienced a partial collapse. Two exits, either going left or right, are in the tunnel. You no longer know the state of either exit as one

or both may now be blocked. Rather than expend energy searching both directions, you deploy a swarm of robots. After some time you see how many robots are near you and from that observation are able to determine the state of both potential exits.

Many challenges need to be addressed before the proposed scenario is physically realizable, but this work lays a promising theoretical foundation. PDE models for each potential scenario can be quickly generated and the observed robot occupancy in the middle can be compared to each PDE to find the nearest model, as demonstrated next.

2.6.2 Boundary Identification for 1D Environment

Three different boundary condition pairs were considered for the 1D environment: double sink, double wall, and mixed. Using $M \approx 100,000$, $N = 10$, and an initial distribution placing the majority of robots in bins furthest away from the boundaries, only a few iterations (time steps) were required before the simulations revealed a distinguishable distribution of robots throughout the environment. The associated PDE models also captured the unique distribution but much more quickly.

By observing the number of robots in a central bin (furthest from the boundaries) and comparing the occupancy to each of the three PDE models, it is possible to use a least-squares error to determine which model is most similar to the observed behavior and therefore determine the boundary conditions. To demonstrate the effectiveness of this simple approach, a simulation for each boundary condition pair was executed in MATLAB with the middle bin occupancy compared to the three potential PDE models. As shown in Fig. 2.7, the simulation consistently and reliably converged to the appropriate model within 15 time steps.

Observing the middle bin is the worst-case scenario because it is furthest from the discriminating features and would be ineffective in determining whether the wall was to the ‘left’ or ‘right’ in the case of the sink and wall boundary scenario. Nonetheless, Fig. 2.7 demonstrates the potential of using the proposed continuum limit methodology to develop environmental models for quick identification. The presence or absence of an obstacle was determined reliably after simulated robots

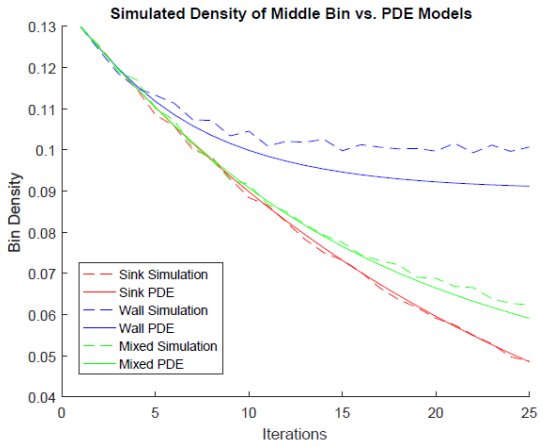


Figure 2.7: Convergence of central bin occupancy for 10-bin simulated environment to corresponding PDE model reliably occurs in 15 discrete-time steps.

within the swarm moved 15 times in a worse-case scenario where communication was absent and the robots were reduced to random movements.

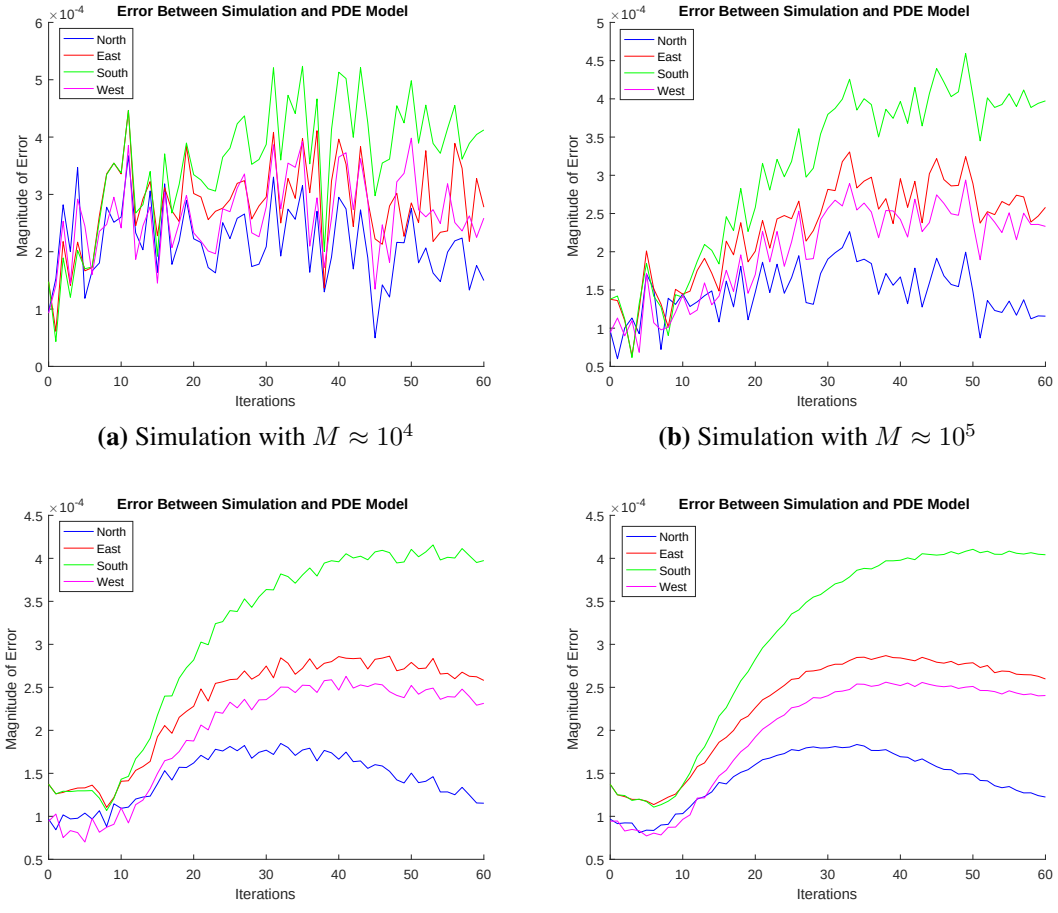
2.6.3 Doorway Detection in 2D Environment

Looking toward physical implementations, a 2D environment is more common but the same general methodology can be applied. Consider an office room with no light and a single doorway. It may be impossible to identify where the doorway is after a partial building collapse and extremely dangerous for survivors in the room to search for the door. A safer option may be to use a swarm of robots to safely locate the exit.

To demonstrate the feasibility of this illustrative scenario, a simulation and corresponding PDE models were developed using the presented methodology. The simulation consisted of a 10×10 bin interior environment. Additional boundary bins were placed around the environment, all corresponding to walls except for four consecutive, central bins in the 'north' wall that were modeled by sinks to represent the available doorway. Four separate PDE models were developed–all identical for the internal model but with varying boundary conditions to represent a central doorway in either the north, east, south, or west wall.

The bin occupancy for four central bins in the simulation was compared to corresponding points in each PDE model to create a 4×1 difference vector. The norm of the difference vector, hereby

referred to as the error, for each PDE model was averaged over ten simulations. The averaged error was then plotted at each time step to show the proximity of the simulation to each of the four PDE models over time. Results are shown in Fig. 2.8. When $M \approx 10^4$, a statistical discernment exists between the error for each PDE model. Increasing the number of robots by an order of magnitude clarifies the separation of the error between the models with the north model having the lowest error which is encouraging as it matches the simulation. Further increasing the number of robots refined the separation between models.



(a) Simulation with $M \approx 10^4$

(b) Simulation with $M \approx 10^5$

(c) Simulation with $M \approx 10^6$

(d) Simulation with $M \approx 10^7$

Figure 2.8: Error between the central observation area for a 10×10 simulated environment with a four-bin doorway in the north wall and each of the four PDE models for doorways in the north, east, south, and west walls.

It is important to not misconstrue the data from Fig. 2.8. Identifying which of the four walls contains the single doorway does not require $M \approx 10^5$ robots nor are error plots the desired method for environment detection. We show these plots to illustrate the convergence properties of the PDE model and to support the theoretical basis of the proposed methodology. With $M \approx 10^4$ or fewer robots, see Fig. 2.8(a), there is already a significant difference in the statistical properties associated with the four possible environments. In the following section, we will show how this statistical difference can be physically visualized.

Nonetheless, Fig. 2.8 supports the use of PDE models in place of simulations for determining emergent swarm behavior and using the resulting robot density to identify environmental features such as unobstructed doorways. Further work is needed to define the correlation between robot count, environment size, features, and resolution. More sophisticated robot behaviors will also impact these four key variables and will be our next focus. However, here we have provided a theoretical foundation for a worst-case scenario in terms of robot exploration. Robots unable to communicate and that are reduced to random motion can still be used to identify in which direction a doorway lies by observing the density of robots in another area.

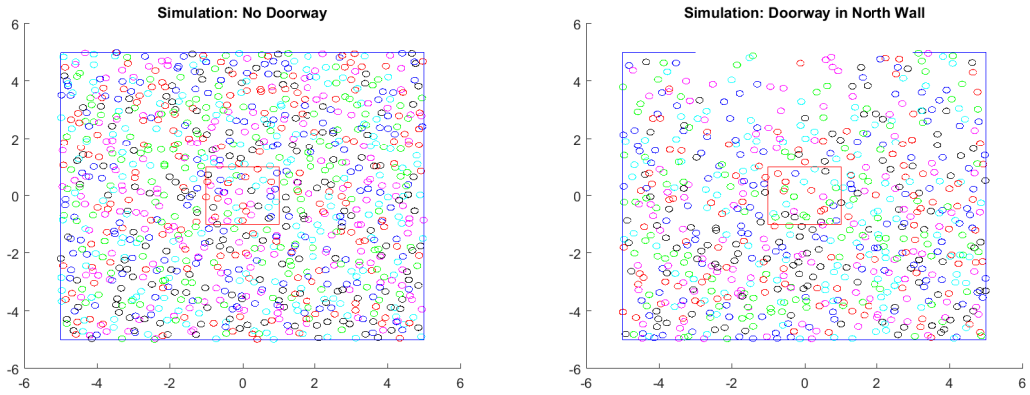
2.6.4 Illustration of Methodology for Particle Collisions

Our ultimate goal is to have a physical realization of a swarm wherein the observed robot density in one area can be used to infer the presence of obstacles in an unknown environment. This work provides the beginning theoretical foundation for achieving our goal but as a further illustration of how robot density in one region can be used to determine the presence of obstacles in another, even in a worse-case scenario, consider the collision of particles in a gas.

In a simplified model, each particle travels a straight trajectory until a collision occurs, at which point the particle is deflected in a way to conserve momentum [15]. As the number of particles in a bounded area increases, individual particle motions also become increasingly random and approach the behavior we have assumed for robots in this preliminary work. By extension,

there should be an observable difference in the number of particles present at the center of unique environments–a difference that can distinguish between the possible environments.

We illustrate the particle analogy in MATLAB. Initially, the square environment is fully bounded so no particles can escape and the particles approach a uniform distribution as expected. When a doorway is added to the north wall, paralleling the doorway detection simulations in Section 2.6.3, particles are able to escape and hence fewer are present in the environment. Snapshots of both environments are shown in Fig. 2.9 at equivalent times. The center red square aids in comparing the density of particles in the middle of the environment. Fewer particles are clearly present in the observation area for Fig. 2.9(b) and hence one can conclude that environment contains the doorway.



(a) Distribution of particles with no doorway **(b)** Distribution of particles with a single doorway

Figure 2.9: A comparison of particle density in two distinct environments confirms the density in one area is distinguishable for different environments.

Robots are capable of much more sophisticated behaviour than gas particles but this work is a first step toward modelling emergent behavior with PDEs and developing models to identify environmental features from observed swarm behavior. Gas particles approach the random motion assumed for the initial robot models in this work and provide a stepping stone toward physical demonstration of our proposed methodology.

2.7 Conclusions

This foundational work proposes the use of continuum limits to model the emergent behavior of a swarm of robots with stochastic behaviors. By extending discrete local rules into a continuous-time domain, a PDE model is obtained. The base PDE model describes the interior behavior of robots in the environment while the PDE boundary conditions encode environmental features—namely open space or obstacle.

Solutions to PDEs can be quickly computed and are independent of the number of robots and largely independent of environment size. By contrast, large-scale stochastic networks like robotic swarms require many hours or even days to run with the time depending on both the number of robots and the network or environment size. As a result, PDEs can serve to provide quickly generated environmental models.

Different environmental features cause unique, distinguishable distributions of robots in the environment so observations at one location can be correlated to environmental features in another location. As shown in this preliminary work, observing the number of robots in a central location can be used to identify the presence of sink or wall boundaries by comparing to already determined PDE models. The robots considered in this work represent a worse-case scenario where no communication or advanced decision making is implemented and motion is random. Nonetheless, valuable information can be obtained as partially illustrated in this work.

Future work will focus on extending the complexity of the robot behaviors modeled, introducing internal obstacles, and implementing the results on physical robots in real-world environments. Additional investigations into the relationship between number of robots, environment size, and exploration time will also be investigated to provide confidence bounds on the results obtained.

Chapter 3

Classifying Environmental Features from Local Observations of Emergent Swarm Behavior [2]

3.1 Summary

Robots in a swarm are programmed with individual behaviors but then interactions with the environment and other robots produce more complex, emergent swarm behaviors. One discriminating feature of the emergent behavior is the local distribution of robots in any given region. In this work, we show how local observations of the robot distribution can be correlated to the environment being explored and hence the location of openings or obstructions can be inferred. The correlation is achieved here with a simple, single-layer neural network that generates physically intuitive weights and provides a degree of robustness by allowing for variation in the environment and number of robots in the swarm. The robots are simulated assuming random motion with no communication, a minimalist model in robot sophistication, to explore the viability of cooperative sensing. We culminate our work with a demonstration of how the local distribution of robots in an unknown, office-like environment can be used to locate unobstructed exits.

3.2 Introduction

While individual or small teams of robots have been used for exploration in relatively controlled settings, harsh environments like partially collapsed buildings and underground mines remain an important challenge. Our goal is to leverage the domain of swarm robotics to expand the type of environments which can be reliably explored. In this work we provide a base level for what information can be obtained about features in an unknown environment from a minimalist swarm -

[2]Published in *IEEE/CA Journal of Automatica Sinica*, 2020

one comprised of very simple, inexpensive robots that contain no sensing or direct communication abilities.

Inspired by cooperative biological systems like ants and bees, robotic swarms are a relatively new area of robotics research that extend multi-robot systems by incorporating significantly more robots. The increase in robot numbers is frequently countered by a decrease in the individual robot complexity to ensure the entire system is scalable [4] and more easily managed by a human. Like multi-robot systems, swarms can accomplish complex tasks that exceed the capabilities of the individual robots but swarms have additional benefits in exploration applications. The swarm can cover an area more efficiently than an individual robot or small team, and, as we demonstrate in this paper, environmental features can be inferred without requiring robots to explicitly store or relay information, further increasing system robustness and decreasing exploration time.

Feature inference is achieved using local observations of the swarm distribution. Each individual robot is programmed with a set of known behaviors. Frequent robot-robot and robot-environment interactions naturally lead to more complex but often difficult to predict emergent behaviors. The emergent behavior can be quantified by different properties (see, e.g., [16]) but in this work we focus on how the robots are distributed. Hence there is a correlation between three key factors: individual robot behaviors, environment features, and the observable distribution of robots. If two factors are known, the third can be inferred. If the robot behaviors are independent and the environment is known, a partial differential equation (PDE) can be derived to exactly model the robot distribution [17]. With a finite number of environments, a least-squared error comparison between each derived PDE model and an observed robot distribution can be used to identify in which environment the robots are moving.

As individual robot behaviors become more sophisticated and environments become more varied, there is no plausible deterministic approach for predicting the robot distribution, but there is still a correlation. In this paper, we exploit the correlation by using a simple, single-layer neural network to demonstrate how known individual robot behaviors and locally observed robot distributions can accurately predict environmental features. We focus on a minimum sensing scenario

where the robots are limited to random motion and have no communication abilities. A simple, single-layer neural network is then trained to correlate the number of robots in a central region of the environment with the environment type itself. Despite the limited robot capabilities and local observations, this work shows the distribution of a swarm can be used to quickly and accurately infer environmental features.

The primary contribution of this work is providing a baseline feasibility study to affirm that environmental information can be obtained from a minimalist swarm where the robots are not equipped with sensors or communication. Current robotic exploration approaches have fundamental problems when applied to disaster scenarios because communication is often unreliable, traditional robots experience high failure rates, and sensing is limited. As such, our results can have significant impact on physical implementations of swarms for disaster scenarios. Rather than trying to design more complicated swarms to overcome these challenges, our work begins by assuming that robots have minimal capabilities. Even without sensing and communication, a local observation of the swarm can be used to infer environmental features in a simulated disaster scenario.

The remainder of this work is organized as follows. Related work is presented in Section 3.3. We then describe the simulation test platform and introduce the neural network used to correlate observed robot distributions with training environments in Section 3.4. Section 3.5 evaluates the performance of the implemented network. We extend the simulations in Section 3.6 to explore the robustness of the methodology with respect to variation in environmental features and swarm size. Conclusions and future work are presented in Section 3.7.

3.3 Related Work

The sophisticated emergent behaviors of cooperative biological systems in nature have inspired many researchers to focus on recreating observed swarm properties in robotic systems. Properties like group consensus ([18], [19]), task allocation ([20], [21], [22]), and localization ([23], [24]) have applications in standard exploration strategies. These approaches essentially expand multi-

robot strategies so they face fundamental scalability challenges by requiring global communication and/or localization.

In response, many works have limited the communication range to local communication between robots as in [25], [26], or [27] but these works still rely on sensory information which is not robust, especially in disaster scenarios, because they are approaching the design as a reduction of multi-robot abilities. By contrast, our work is establishes what information can be obtained from a minimalist swarm. Additional sensors can then be added to build up to the required level of performance as appropriate.

Several impressive works have been done fully in the swarm domain but these strategies incorporate some form of global knowledge as in [28], are computationally expensive like [29] or [30], or imbue the robots with sophisticated sensory knowledge to construct individual maps as in [31]. Unlike these works, our approach simulates robots with no sensory information or communication. We are focused on disaster scenarios where these other works will fail because sensory information is not robust and computational resources are limited.

Other key contributions to swarm research have focused on a top-down approach: the application is defined so individual robot control strategies are designed to reach the goal using optimization methods (e.g. [32], [9], [33], [11]). These approaches do produce the desired behaviors but therefore require first knowing what behavior is desierd. The top-down approach also obscures fundamental relationships between individual robots and the environment. Instead, we take a unique bottom-up approach by exploring what can be done with very simple systems and leveraging the number of interactions much more like biological systems.

Our work is simulation-based but the robots are modeled such that they do not exceed the physical capabilities of current mobile robots. There are a variety of commercially available mobile robots, including the Khepera [34] and e-puck [35] which can be used for small robot teams but are prohibitively expensive for swarm research. The constraint of scalability places additional limits on swarm platforms beyond component cost. Currently the kilobot [36] is one promising research platform because the robots can be managed collectively.

3.4 Formulation of Test Environment

3.4.1 Pattern Correlation

Individual robot behaviors, environmental features, and observable robot distributions are correlated so if two of the characteristics are known, the third can be inferred [17]. In this work, the individual robot behaviors are known and we want to use the local robot distribution to predict global environmental features. Environmental features are inferred using a simple, single-layer neural network that is trained to correlate local observations of robot distribution with the labeled environment in which the robots were simulated. Individual robots have no knowledge of the environment themselves. Instead, a central agent (human or computer with visual data) uses the distribution of robots immediately around them and a trained neural network to predict global environment properties not visible to the central, independent agent.

A single input neuron serves as a bias term while each additional neuron in the input layer considers the number of robots in a single observation bin at a given time. In order to reduce the impact of initial robot count on environmental correlation, the raw robot density data is first normalized before being applied to the neural network. The output is the likelihood the observations came from each of the potential environment classes. Logically, the goal of the neural network is to determine the probability of the training environment, C, being from class k given an observation of the local robot distribution, $\mathbf{x}$. Mathematically, the desired output from the trained neural network for a specific simulation run n can be formulated as

$$g_k(\mathbf{x}_n) = p(C = k|\mathbf{x}_n) \tag{3.1}$$

where the value of $g_k(\mathbf{x}_n)$ represents the probability that observation n came from environment class k .

It is further desired to have the conditional probability output be a function of tunable weights, $\mathbf{w}$, that can be trained to maximize the likelihood of the data distinguishing between all K potential environment classes. Introducing a function $f(\mathbf{x}_n, \mathbf{w}_k)$ and defining

$$g_k(\mathbf{x}_n) = \frac{f(\mathbf{x}_n, \mathbf{w}_k)}{\sum_{m=1}^{K} f(\mathbf{x}_n, \mathbf{w}_m)} \tag{3.2}$$

ensures that the probability of any given environment class, k, is between 0 and 1, and the probability across all K potential classes sums to 1 independent of the choice of weights. We then choose

$$f(\mathbf{x}_n, \mathbf{w}_k) = \exp(\mathbf{w}_k^{\mathsf{T}} \mathbf{x}_n) \tag{3.3}$$

to create a general softmax formulation [37]. To further understand our choice, we first define an indicator variable, $t_{n,k}$, for each robot density observation, n, and each potential environment, k, to identify from which environment class observation n came. The environment classes are labelled with an integer value from 1 to K. The indicator variable for observation n is therefore defined as

$$t_{n,k} = \begin{cases} 1, & \text{if } \mathbf{x}_n \text{ is from class } k \\ 0, & \text{otherwise.} \end{cases} \tag{3.4}$$

We form a measure of how distinguishable the classes are from each other by considering the product of the probabilities for all N environment observations, hereby called the *data likelihood*. Using the class indicator variables for a K-class scenario, the data likelihood, $L(\mathbf{w})$, is expressed as

$$L(\mathbf{w}) = \prod_{n=1}^{N} \prod_{k=1}^{K} g_k(\mathbf{x}_n)^{t_{n,k}}. \tag{3.5}$$

Maximizing the likelihood with respect to the tunable weights increases the network's ability to classify an observation sample. Finding the weight values that maximize the likelihood requires first finding the gradient of L in (3.5), but for computational efficiency we instead maximize the natural logarithm of the likelihood:

$$\log\ L(\mathbf{w}) = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{n,k} \log g_k(\mathbf{x}_n). \tag{3.6}$$

Substituting (3.2) back into (3.6) and taking the gradient with respect to the weights of a single output neuron, j, we get

$$\begin{aligned} \nabla_{\mathbf{w}_j} \log\ L(\mathbf{w}) &= \sum_{n=1}^{N} \sum_{k=1}^{K} \frac{t_{n,k}}{g_k(\mathbf{x}_n)} \nabla_{\mathbf{w}_j} g_k(\mathbf{x}_n) \\ &= \sum_{n=1}^{N} (t_{n,j} - g_j(\mathbf{x_n}))\mathbf{x_n}. \end{aligned} \tag{3.7}$$

Equation (3.7) is still nonlinear with respect to $\mathbf{x}$, so an iterative update process is needed to find which weights maximize the conditional probability in (3.2). This leads to a gradient ascent update rule for the weights of the form

$$\mathbf{w}_j(i+1) = \mathbf{w}_j(i) + \alpha \sum_{n=1}^{N} (t_{n,j} - g_j(\mathbf{x}_n))\mathbf{x}_n \tag{3.8}$$

where α is the learning rate. Throughout this work, a constant learning rate of $\alpha = 0.0001$ was used and the weights were updated over 500 iterations, as further iterations did not greatly change the final weight values. These values are not optimized as the focus of this work is not the neural network but rather demonstrating the feasability of using local observations of robot distributions to infer more global environmental features.

3.4.2 Simulated Robot Swarm

The focus of this work is establishing what features can be determined from a minimalist swarm for applications in disaster scenarios where resources are limited. A specific example use case for this work is locating viable exits in the case of a partial building collapse. To model this hypothetical scenario, we used MATLAB to simulate the robot distribution in 1D and 2D environments to model hallways and office rooms. Each simulation consisted of a user-defined environment that contained either a single line of N internal bins or a square of $N \times N$ internal bins for the one-dimensional (1D) hallway and two-dimensional (2D) room scenarios, respectively. Boundary bins form the perimeter of each environment and are specified as a sink or a wall to represent an opening or an obstruction in the environment. The use of bins to model the environment with

potential obstacles placed at the boundary is inspired by Yamauchi's occupancy grid approach to map generation [38].

At every iteration, each robot randomly selects a desired bin that is orthogonally adjacent (no diagonal motion) to its current bin using a uniform distribution so that each potential bin is equally likely. If the desired bin is not a boundary, the robot will move to the adjacent bin at the next time step. If the desired bin is a sink boundary, the robot is removed from the simulation because the sink represents an opening in the environment; however, if the desired bin is a wall, the robot instead stays in its current bin for the next time step. The robot density, that is, the number of robots in every bin of the environment, is stored at each iteration.

For this work, we considered three distinct environments, or classes, for the 1D scenario and four classes for the 2D scenario. Each 1D environment approximated a hallway with $N = 10$ internal bins and a potential exit at the left and right boundaries. The boundary bins were varied such that Class I had a sink boundary at each end so that the hallway was unobstructed, Class II had a wall boundary at each end so that robots could not escape the hallway, and Class III had a sink at the left edge and a wall at the right edge so that robots could only escape the hallway at the left side.

The 2D environments modeled a square office space consisting of $N \times N$ internal bins surrounded by wall boundaries. A set of five consecutive sink bins were placed near the middle of a single wall, spanning bins 5—9, to model an unobstructed doorway. The North Class contained a doorway in the 'north' wall, East Class had the doorway in the 'east' wall, and so forth.

We extend our hallway and office metaphors by evenly distributing the robots in the central-most bins of the 1D and 2D environments and observing the density in these central bins at each time iteration. This approach models a person who is in the middle of an unknown environment, equally far from all potential openings, and deploys a swarm of robots in the area around them. The person then uses the observed density of robots immediately around them to predict the location of a viable exit.

To evaluate the feasibility of using locally observed robot densities and a relatively generic neural network to predict the environment being explored, 200 simulations were run in MATLAB for each environment class in both the 1D and 2D scenarios. The number of robots in each interior bin of the environment was stored at every iteration to form a *full population* for each simulation. The simulation time in MATLAB increased significantly as the number of robots increased, but 1D simulations typically required just a few seconds to run while 2D simulations often required several minutes.

3.5 Evaluation of Swarm Methodology

3.5.1 Overview

Robot densities from the observed bins at the desired time are selected from the *full population* for the 1D or 2D scenario to form a complete *data set* for that environment type. Each row in a *data set* corresponds to a single MATLAB run. Each column in the *data set* represents the number of robots in one observation bin. Each column is normalized and then used with the neural network to form a correlation between the number of robots in locally observed bins and the environment being explored. In all cases, 70% of the *data set* was used to train the neural network with each environment type being equally represented. The remaining 30% was reserved for evaluating the classification accuracy of the trained network.

3.5.2 Performance in 1D Environment

The 1D hallway model was evaluated first as a benchmark for the feasibility of using local density observations to infer global environmental features like exits. For all 1D simulations, robots moved freely in the ten internal bins, numbered 2—11. Bins 1 and 12 were boundary bins, represented as either a sink or wall, and defined the three potential environment classes. It is assumed a physical implementation would have a person located in the center of the environment deploy a swarm of robots to help predict which of the two potential directions is a viable exit; hence, robots are initially distributed evenly in bins 6 and 7. The person can only observe the

number of robots in these same two bins to model the limited visibility likely in a disaster scenario. To model this scenario, the neural network weights were initialized using a uniform distribution on the open interval $(0, 1)$ and then updated using (3.8) with the observed robot density in bins 6 and 7.

Initially, the neural network was trained using the robot densities observed in bins 6 and 7 at a single time step while the number of robots placed in the environment was systematically varied from just 10 robots up to 1000 robots. The goal of this preliminary simulation was to determine how many robots and what length of time would ensure a sufficient number of interactions to encode environmental features. Two hundred simulations were run for each of the three environment classes creating a *data set* with 600 rows and just two columns, the first for observed densities in bin 6 and the second for bin 7. Fig. 3.1 summarizes the classification accuracies from this exploration with the results being averaged over 50 trials at the same robot count and time to reduce the impact of statistical variation.

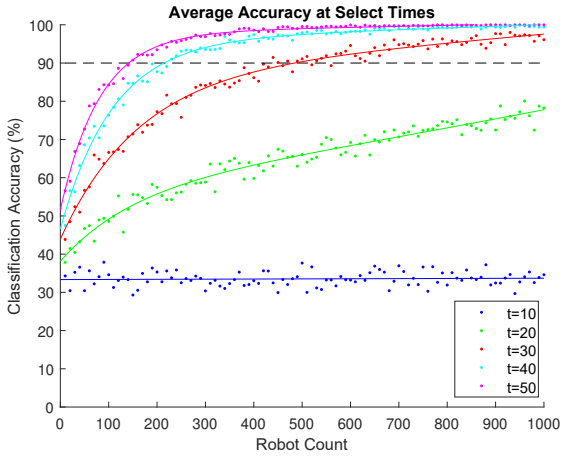


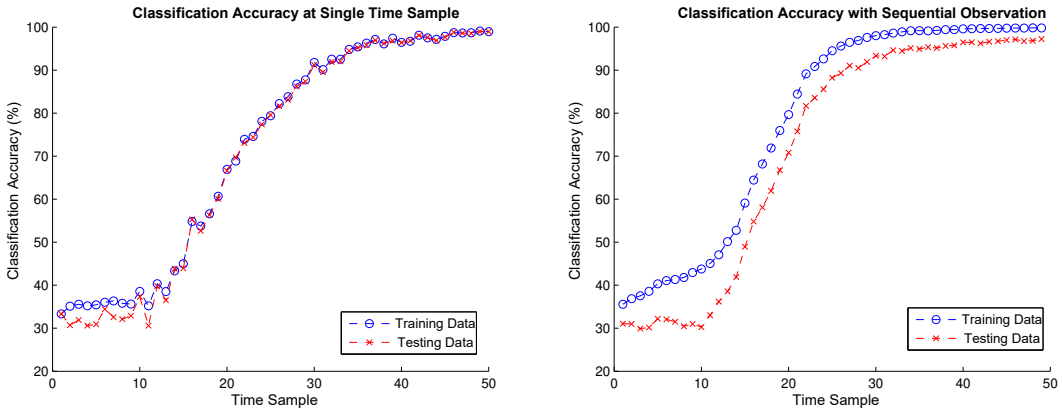
Figure 3.1: A stronger correlation is achieved between locally observed robot density and the environment when more robots are initially distributed. In addition, the robots need a sufficient amount of time to interact with each other and the environment. For example, 90% of the environments were classified correctly when 500 robots moved about 30 times.

Adding more robots to the environment generally improved the classification accuracy of the neural network as expected because more interactions occur. However, there are diminishing re-

turns in the classification accuracy with respect to increased robot count, suggesting the system can become saturated. Exploration time is also a key factor in evaluating the effectiveness of correlating environmental features to observations of the local robot density. A minimum of 9 moves are required for a robot to reach a potential wall boundary and return to one of the central observation bins so it is not surprising that the classification accuracy was essentially random, independent of robot count, for a time sample of 10. Observing the central robot density after robots moved 20 times significantly increased the classification accuracy but it required 30 moves before the classification accuracy was reliably over 90% for the robot counts considered.

Based on the results in Fig. 3.1, a robot count of 500 was used for the remaining analysis in 1D as 500 robots appeared to provide a sufficient number of interactions without saturating the environment. A closer look at the classification accuracy of the neural network with 500 robots is shown in Fig. 3.2(a). The trained network classified the 420 training samples and 180 previously unseen test samples with over 90% accuracy after 30 time steps. In our hypothetical hallway scenario, these results indicate that a person could allow the robots in a swarm to move 30 times and then look at the distribution immediately around them to determine which direction is unobstructed. More realistically, a person would watch the evolution of the robot density around them, which means the neural network should consider the density in bins 6 and 7 at the current time step and all previous time steps. We define this approach as a *sequential* observation. Using the sequential bin density did decrease the time required to reach a desired classification accuracy as shown in Fig. 3.2(b). It required observing the number of robots in bins 6 and 7 for about 25 robot moves before training environments could be classified with 90% accuracy.

As expected, the initial classification accuracy in both approaches of Fig. 3.2 was approximately 33%, equivalent to randomly guessing one of the three potential environment classes. As was noted earlier, the 1D scenario requires a minimum of 9 moves for a robot to reach a potential wall boundary and return to one of the central observation bins. This reflection property is affirmed in the results of Fig. 3.2, where the accuracies begin to steadily climb after time step 9. Fig. 3.2(b) indicates a slight overtraining of the neural network as the training data was consistently classified



(a) **(b)**

Figure 3.2: By observing the number of robots in the center of a simulated 10-bin hallway, a simple neural network could correlate the density with the environment being explored and predict the location of exits with 90% accuracy after 500 robots move just 30 times whether considering a single time step (a) or a sequential observation (b).

with higher accuracy than the testing data. Nonetheless, using a sequential observation of robots in bins 6 and 7, which better model a human observer, did reduce the number of robot moves required to predict the environment class for a desired accuracy as anticipated.

3.5.3 Performance in 2D Environment

A similar analysis was performed using the 2D simulations. The 2D environment represents an office-type scenario where a person is attempting to identify which wall contains the single, 5-bin doorway in a square, 10×10-bin room by observing the local distribution of robots. An *observation center* was placed at bin $(7, 7)$, near the middle of the environment to represent our hypothetical office employee who is equally distant from all four defining boundaries. A person can deploy robots immediately around them, which corresponds to the simulated robots being initially distributed equally in the eight bins surrounding the *observation center*.

With four potential environment classes and 200 simulations per class, a *data set* with 800 rows was created for each scenario. Each generated *data set* has eight columns, each corresponding to a single observation bin around the *observation center*. As in the 1D scenario, the classification

accuracy of the neural network was averaged across 50 trials to reduce the impact of statistical variation in the results.

Once again, an initial simulation was performed to determine an appropriate number of robots for the 2D environment. The number of robots distributed in the eight bins surrounding the *observation center* was systematically increased from just 100 up to 15000 in increments of 100. Fig. 3.3 summarizes the results of this exploration. As anticipated, increasing the size of the environment–from 10 bins in the 1D scenario to 100 bins in the 2D scenario–also increased the number of individual robots and robot moves to explore the environment.

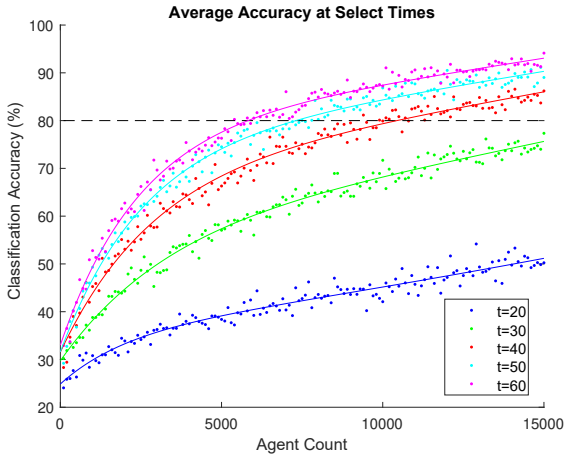


Figure 3.3: Increasing the environment size meant more robots were needed to perform the classification and each robot needed more moves to encode features. For a 10×10 environment, it took $10,000$ robots and approximately 40 moves before enough interactions had occured for the neural network to accurately identify 80% of the environments.

Fig. 3.4 shows the relative distribution of $10,000$ robots after they have moved 40 times in the 2D environment. The environment pictured is a sample from the North Class. Robots can only exit through the north doorway and no robots will re-enter the environment through a doorway, so the adjacent bins have a noticeably lower density of robots. However, the decrease is less distinct in the bins surrounding the *observation center* at bin $(7, 7)$ where the hypothetical office worker is located.

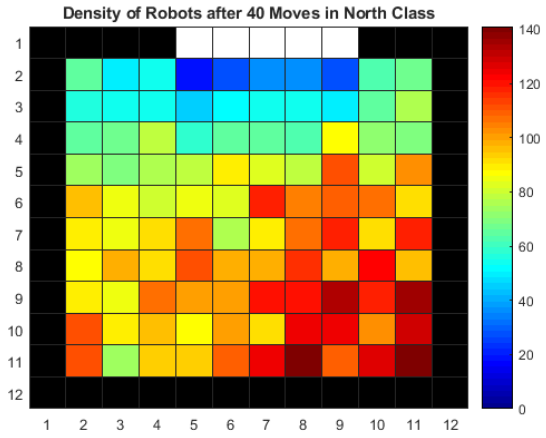
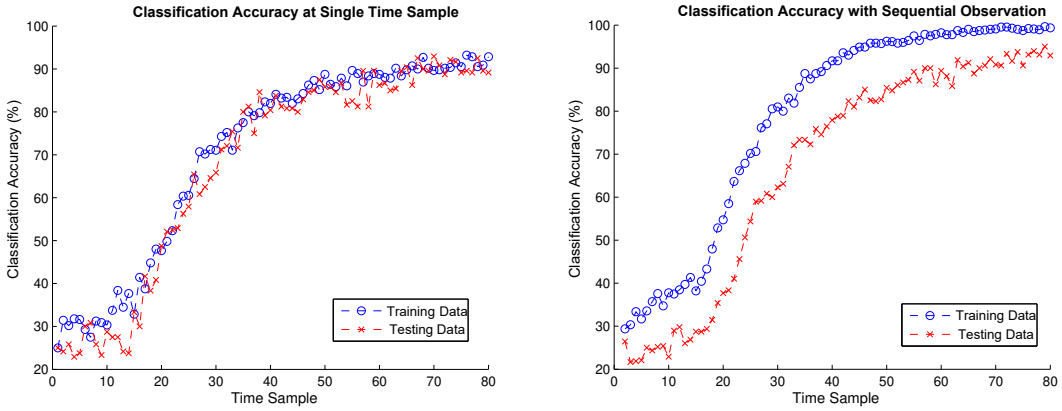


Figure 3.4: A heat map of the robot density for $10,000$ robots after 40 time steps in a 2D environment with a north doorway shows a lower density in the northern bins as expected. The difference is less distinct in the bins surrounding the *observation center* at (7,7).

If the hypothetical office worker attempted to classify the environment based on the least dense bin in the *observation center*, they would obtain only a 21% accuracy after 40 robot moves. By contrast, using the density in all 8 bins of the *observation center*, the neural network classified more than 80% of the environments correctly after approximately 40 time steps as reconfirmed in Fig. 3.5(a), despite the subtle variation in robot distribution. Using sequential observations around the *observation center* improved the classification accuracy slightly. Fig. 3.5(b) shows the test data still required approximately 40 robot moves to reliably reach over 80% classification accuracy. In our hypothetical office scenario, a person near the center of the environment could therefore predict in which wall a doorway was located with 80% accuracy by observing the number of robots around them over the course of about 40 robot moves.

Fig. 3.5(a) shows that for about the first 15 moves, the classification accuracy was equivalent to a random guess, similar to the 1D simulations. This is not surprising as at a minimum, a robot initially placed in bin $(8, 8)$ (the lower right initialization bin) would need 7 moves to reach either the east or south wall and return to the observed area. A robot in the upper left requires a minimum of 9 moves to return after encountering a wall in the north or west. If a robot moves one bin 'left' or 'right' during its minimal trajectory, the robot may miss a door and return to falsely inflate the number of robots in an observed bin. This is one reason why the classification accuracy increased



(a) **(b)**

Figure 3.5: Locating a doorway in the 2D environment using eight central bin densities required approximately 40 time steps to reach 80% accuracy if using a single time observation (a) or using observed robot densities at all previous times (b).

much more slowly in the 2D scenarios. The increased number of potential bins to explore also increased the time required for classification.

The same general behavior extends to the sequential observations shown in Fig. 3.5(b), though there are signs of overtraining as the training data had a consistently higher classification accuracy than the testing data. Even with these results, the local observations of a robot swarm can be correlated to global environmental features. Indeed, Fig. 3.5(b) shows that the number of robots around a central bin can be used to form an educated prediction about which wall contains a doorway—even with a simple single-layer neural network and minimalist robots.

3.5.4 Robustness of the Classification Process

The robustness of the trained network in Fig. 3.5(b) at time 40 was explored with respect to variations in the environment and a decrease in robot count. For the first investigation, test data was gathered from simulated environments where the doorway had been shifted with respect to the training environments. The doorway width was maintained at five bins and was incrementally moved from the far left (a shift of -3) to the far right (a shift of +2). A total of 60 simulations were conducted per environment variation to ensure a comparable test data size of 240 samples per doorway location.

The resulting classification accuracy for each new doorway position is summarized in Table 3.1. As expected, the highest classification accuracies of 97% occured in environments most similar to the training environment. Shifting the doorway three bins left resulted in the lowest classification accuracy of 77% because this corner is the furthest from the original training doorway. These results affirm one major advantage of the neural network when compared to a PDE approach—the ability to avoid explicitly describing environmental scenarios. Indeed, the trained neural network still predicted the location of a doorway with 77% accuracy–significantly better than random–even when the doorway was shifted to the far side of a wall and only partially overlapped the original doorway position.

Table 3.1: Classification Accuracy for Shifted Doorway

Door Shift	Accuracy
-3	77%
-2	88%
-1	95%
0	97%
1	97%
2	95%

The trained neural network can also account for a large loss of robots. For the next robustness investigation, the number of robots in a test enviornment was systematically reduced from 10000 to 1000 to simulate potential robot failures. Sixty simulations were run for each environment class, so once again, 240 data samples were used to evaluate the neural network for each reduced robot count. The classification accuracy was averaged over 10 separate trials to reduce the impact of statistical variation. Fig. 3.6 shows that the classification accuracy decreased as the number of robots was reduced, as expected, but remains at nearly 94% as accurate as the training scenario when only 5000 robots are present. This means that half of the robots can fail, but the worker in our hypothetical building collapse can still predict the environment and be 94% as accurate as if all the robots were still functional. Further, the neural network classified the environment with over 64% of the original accuracy when only 1000 robots are present. Nine out of ten robots can

fail, but the network is still able to predict which wall contains the single doorway with better than random accuracy.

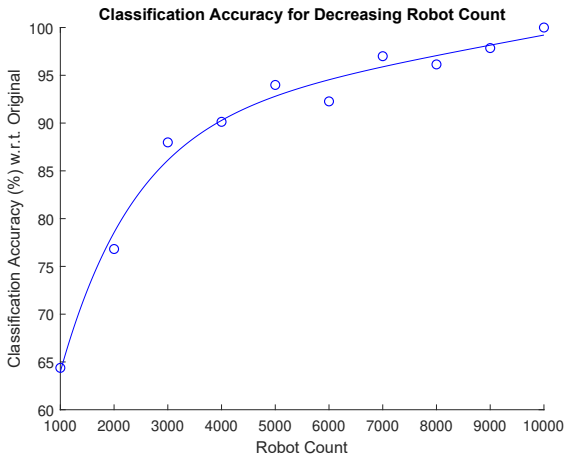


Figure 3.6: A network trained on 10000 robots can still reasonably identify a 2D environment when the system undergoes large-scale robot failure.

3.6 Extending the Neural Network Results

Using just one-tenth of the original number of robots, the neural network is ruling out certain environment classes based purely on observations of the local robot distribution. Table 3.2 summarizes how the neural network classified the different environments for a single run in a confusion matrix. Sixty of the test samples contained a doorway in the north wall (N), and 36 of those samples were correctly classified; however, 9 of the samples were mistakenly classified as having a door in the west (W). Looking at samples which contained a doorway in the west wall, 38 of the 60 samples were correctly classified while 8 were misclassified as having a door in the north and 14 classified as having a door in the south (S). Zero were classified as the 'opposite' where a door was placed in the east (E). Overall, the confusion matrix indicates environments were most rarely confused with their opposite direction which is encouraging.

Ruling out the least-likely environment by observing robot distributions makes intuitive sense. When the bottom row of observation bins has a low robot count, it is likely that the door is in the

Table 3.2: A confusion matrix of the test data classification for the 2D, sequential observation scenario with just 1000 robots shows that the neural network rarely classifed an environment as its cardinal opposite when pre-trained with 10 times as many robots.

		Predicted Class				
		N	W	S	E	
Actual Class	N	36	9	2	13	
	W	8	38	14	0	
	S	0	7	43	10	
	E	18	5	4	33	
						62.5%

south wall but robots may still be escaping east or west with regular frequency so reaching full classification confidence remains difficult. The classification process can be further understood by comparing the relative weight values of the neural network for each environment class. For each class, the highest weight is associated with the corner furthest away from the doorway while the associated row or column also contains generally higher weights. Hence, having a large number of robots in three of the observed bins greatly reduces the likelihood of the opposite wall from containing a doorway. Determining which specific wall contains the doorway is more challenging because now the discriminating weights are much more similar and, as can be seen in Fig. 3.4, the variation in robot density is less clear.

Still referencing Fig. 3.4, the distribution of robots does become more distinct in bins closer to the doorway. This observation led to an update scheme that demonstrates how a person can further leverage observations of the emergent swarm behavior in differing environments to locate viable exits. Specifically, the demonstrated strategy moves the *observation center* one bin away from the least likely doorway location, analogous to the office worker moving away from the most crowded area.

The single-layer neural network was again trained using the derived update procedure from (3.8). Training data came from the scenario shown in Fig. 3.5(b) with 10000 robots. New test data was generated with only 1000 robots exploring the same four environment classes, a failure rate of 90%, to show what information can still be obtained about the environment. Sixty simulations were run for each class to generate 240 test samples for consistency with previous experiments.

During the training process, five separate sets of weights were generated. The center set was trained using density data from the eight bins surrounding the original *observation center* at bin (7,7) where the office worker is initially standing. These trained center weights were then used to perform an initial classification. If the classification results indicated that the doorway was least likely to be located in the south wall, the office worker moves one bin in the opposite direction so the *observation center* is now one bin north at $(6, 7)$. A new observation of the robot distribution is then taken and used to predict an updated doorway location using a second set of trained weights. The second set of weights is pre-generated using training data from the eight bins surrounding a north *observation center*. Similar weights are generated for a potential move either east, south, or west.

Fig. 3.7 summarizes how the dynamic *observation center* significantly improved the classification accuracy even when the swarm experienced a drastic 90% failure rate. The test environments were initially classified randomly with an accuracy of about 25% but improved to 40% when considering the number of robots in the surrounding bins for 40 robot moves as shown by the blue line. Moving the observation center one bin opposite the least likely environment class and reclassifying the environment consistently increased the accuracy as shown by the red line in Fig. 3.7. At time 40, the dynamic observation produced a classification accuracy of 51% and the improvement continued throughout the simulated time.

In a disaster scenario, it is very likely the terrain will cause some degree of failure in exploratory robots. Our simulation assumed a 90% failure rate, which left just 1000 robots to explore an unknown domain. A person could still observe the local distribution of this swarm for 40 moves to predict in which direction a doorway is located and they would be about 40% correct. However, if they move once and re-evaluate, their prediction will now be 51% correct. Waiting longer improves both results. In short, a person can regularly update their prediction by moving in a more promising direction and re-evaluating the local robot distribution. Fig. 3.7 shows that moving just once will consistently improve the person's ability to accurately predict where a doorway is located even after mass failure of the swarm.

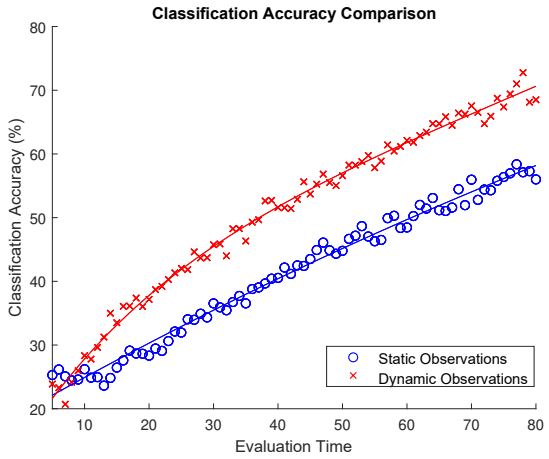


Figure 3.7: Moving the *observation center* one bin opposite the least-likely environment increases the classification accuracy.

3.7 Concluding Remarks

Our focus in this work was to exploit the correlation between individual robot behaviors, environmental features, and locally observed robot distributions to reliably predict global environmental features. Using simulated robots equipped with minimum sensing and no communication, we found that the local distribution of robots could be used to accurately infer information about the environment being explored. A simple, single-layer neural network was sufficient for correlating observations of the robot density in a central part of the environment with the location of openings in the environment. The approach was robust with respect to variations in the environment as well as large-scale swarm failure. We demonstrated how trapped office workers could use a simple microprocessor and observations of the local swarm distribution around them to navigate toward unobstructed openings in hallways or office rooms even after 9 out of 10 robots fail.

This work is a preliminary step in designing swarms of simple, inexpensive robots to explore harsh environments where communication and sensing are unreliable. While there is much to be done to improve the mobility of physical swarms - especially for harsh environments - our work focuses on achieving reliable feature inference given minimal sensing and computational abilities. Our future work will continue building on the general premise of using local observations of emergent swarm behavior to infer environmental features. While our long-term focus is toward

increasing the richness of environmental features that can be predicted, we will next focus on implementing a simulated test platform to better quantify the relationship between swarm size, environment size, and identification accuracy for varying swarm behaviors. This platform will also be used to compare the effectiveness of swarms with respect to smaller teams of robots equipped with more sophisticated sensors.

Chapter 4

Quantifying Swarm Resilience with Simulated Exploration of Maze-Like Environments [3]

4.1 Summary

Artificial swarms have the potential to provide robust, efficient solutions for a broad range of applications from assisting search and rescue operations to exploring remote planets. However, many fundamental obstacles still need to be overcome to bridge the gap between theory and application. In this characterization work, we demonstrate how a minimalist human rescuer can leverage local observations of emergent swarm behavior to locate a lone survivor in challenging environments. The simulated robots and rescuer have limited sensing and no communication capabilities to model a worst-case scenario. We then explore the impact of fundamental properties at the individual robot level on the utility of the emergent behavior to direct design choices. We further demonstrate the relative resilience of the simulated robotic swarm by quantifying how reasonable probabilistic failure affects the rescue time in a complex environment. These results are compared to the theoretical performance of a single wall-following robot to further demonstrate the potential benefits of utilizing robotic swarms for rescue operations.

4.2 Introduction

Swarm robotics is a relatively new domain of research that takes inspiration from cooperative biological systems such as flocking birds, ant colonies, and schools of fish. Like their biologic counterparts, robots in a swarm are governed by local rules but frequent interactions with other robots and the environment generate a more complex, emergent behavior. Sophisticated foraging strategies and complex colony construction by ants reveal just a few of the potential advantages

[3] Submitted in *Robotics and Automation Letters*, 2021

of emergent behavior because these actions are accomplished in a distributed and robust manner. Similar emergent behaviors in artificial systems will be extremely useful in many exploration tasks, particularly in harsh environments with a high probability of robot failure. Disaster scenarios are an especially relevant application domain with an increasing rise in occurance and economic impact [39] and there are currently no viable robotic solutions.

Despite the many important applications, research in robotic swarms has not yet matured to the point of reliable, real-world deployment. Two fundamental hurdles between the potential and reality of swarms are (1) determining what collective behavior will emerge from the swarm and (2) identifying the influence of local parameters. These challenges are not decoupled. Each robot within the swarm may be equipped with a variety of sensors and control strategies which can be considered as parameters. Swarms are necessarily composed of a large number of agents [6], [7] so the choice of parameters is immediately scaled by the size of the swarm which imposes some minor difficulties. The true difficulty comes from the collective behavior that emerges after the robots interact with each other as well as the environment. Robot interactions propagate aspects of the local parameters but there is no closed-form method for determining how individual behaviors will affect the emergent behavior.

Physical implementations would best reveal the full emergent behavior. Swarm test platforms are surfacing and offer improved research development but current realizations are restricted to table-top environments, like the kilobot [40] or Zooid [41], and constrained by the physical number of robots. For example, Georgia Tech's impressive Robotarium offers a remotely accessible testbed for swarm algorithms but only includes 25 physical robots [42]. Beyond environment and size limitations, the initial choice of robot physicality already constrains the range of potential emergent behaviors the swarm can exhibit [43]. Simulation therefore remains a very necessary design step because it can be used to investigate the general impact of local parameters on collective properties. The investigation can inform choices for the base physicality of the swarm and is not as constrained by swarm size nor environmental constraints.

Arguably the most fundamental local parameters dictate how a robot moves and interacts with the environment. Specifically, a robot's speed, motion ability, and sensing range control how the robot will move through an environment. The effect of these parameters will be amplified in a swarm and impact properties of the emergent behavior. A human observing the swarm will likely not be able to identify subtle changes in the emergent behavior but group studies where human participants are asked to classify swarm behaviors by observing simulated results have shown people can recognize general patterns [44], [45], [46]. Robot density, velocity, and relative cohesion were all notable properties that helped human participants classify the simulated emergent behavior.

In this foundational study, we consider a minimalist human rescuer using local observations of the emergent swarm velocity to navigate a simulated disaster environment in an attempt to locate a lone survivor. Robots within the swarm have no communication or localization ability. The influence of fundamental robot features related to motion and sensing can be explored. Variations in local swarm parameters are evaluated qualitatively by observing the resulting motion and area coverage as well as quantitatively by the impact of the parameter variation on the rescuer's ability to successfully locate the lone survivor. From this work, we begin to establish important swarm characterization. The robustness of swarms in terms of parameter variation, environmental features, and robot failure is quantified. We also demonstrate the value of employing swarms in disaster scenarios.

We first place our work in the grander scope of swarm robotics by presenting related work in Section 4.3 before presenting the simulation framework in Section 4.4. Section 4.5 presents a summary of results from our investigation of emergent behavior. We also discuss important observations from our research in Section 4.6 before summarizing our work in Section 4.7.

4.3 Related Work

Our work is motivated by the absence of robot solutions for disaster scenarios including mine rescue [47] and building collapse [48]. Ongoing DARPA challenges and reports [49] indicate a

need for more reliable physical robot implementations. There is promising progress in this domain for single robots [50], [51] and improved path planning to mitigate risk [52]. Swarms can potentially leverage these advancements and further reliability efforts by utilizing a large number of robots for increased robustness. Experiments have shown that emergent swarm patterns can serve as an information storage mechanism [53], [54] which supports the robustness paradigm. Yet, to the best of our knowledge, no work has actually quantified the resilience of a swarm.

In our investigation, we explore a minimalist swarm to establish a baseline performance to which additional functionality can then be compared. Our approach considers the interests of potential swarm users as indictated in an important study conducted by Carrillo-Zapta et al. where target users, including firefighters, identified areas where swarms can support current activities [55]. Study participants considered swarms beneficial for gathering information and supporting communication but emphasized that an actual person should remain 'in the loop' for decision making.

Decision-making by the human can be informed by observations of the emergent swarm behavior. As demonstrated by works in swarm expressivity, human participants were able to classify swarm behavior by 'unfocusing' and instead looking at collective behaviors such as robot spacing, velocity, and relative cohesion [44], [45], [46]. We maintain our baseline approach by only utilizing the swarm velocity for our simulated rescuer.

The results from our work focus on maintaining a minimal approach which can represent a worst-case scenario in robot functionality but also serve as a baseline. More sophisticated algorithms can then be compared to these results for design and application purposes. For example, the potential benefits of improved odometry [56], velocity control [57], shape formation [43], and local communication for risk mitigation [58] can be evaluated.

4.4 Simulation Framework

4.4.1 Swarm Behavior Model

We model a minimalist robot moving in an unknown, two-dimensional environment. The robot is unable to communicate, has no means for localization, and relies on limited sensing. This model allows us to characterize baseline performance for the swarm but also represents a worst-case scenario for physical robot deployments.

The swarm is composed of M such robots where M is a user-defined parameter allowing the significance of swarm size to be explored. Robots are initially distributed randomly within a user-defined radius around the *start_center*. The position of robot i at iteration k is denoted as $\mathbf{x}(i,k)$. Using discrete-time steps, robots attempt to move in a straight line with general desired velocity

$$\mathbf{v}_d(k) = s_s[\cos(\alpha_k), \sin(\alpha_k)], \tag{4.1}$$

with s_s representing the maximum robot speed, while also striving to avoid collisions. Each robot is initialized with a uniformly distributed starting angle α_1. Subsequent trajectory angles are a combination of the previous n trajectory angles and a Gaussian noise calculated as

$$\alpha_k = \begin{cases} \frac{1}{n}\sum_{a=k-n}^{k-1} \alpha_a + \gamma & \text{if } n < k \\ \alpha_1 & \text{otherwise} \end{cases} \tag{4.2}$$

where γ is a normally distributed random variable centered at zero with user-defined standard deviation, σ. We explore the impact of heading control on swarm resiliency by varying σ. The size of the moving window, n, is also varied to determine the importance of *memory* on the emergent swarm behavior.

Once a desired velocity is determined, we ensure the robot does not collide with any objects in its sensing radius, r_s, by using a generalized forcing model. The actual velocity for robot i at time step k is calculated as

$$\mathbf{v}_s(i,k) = \mathbf{v}_d(i,k) + \sum_{w=1}^{W} f_w(w) + \sum_{j=1, j\neq i}^{M} f_s(i,j) \tag{4.3}$$

where f_w and f_s denote the effect of the environment and other robots on robot i, respectively. We next define the repulsive wall force for each of the W walls creating our environment as

$$f_w(w) = \begin{cases} A_s e^{(r_s - d_w)/B_s} \hat{\mathbf{n}} & \text{if } d_w \leq r_s \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

with $\hat{\mathbf{n}}$ as a unit vector pointing into the environment, perpendicular to the wall, and d_w as the minimum distance to the wall segment. The choice of coefficients A_s and B_s dictates how strongly robots will be repelled from obstacles. Robots in the swarm also need to avoid collisions with each other so f_s is a similarly repulsive force imposed on robot i by all other robots. Specifically, we define

$$f_s(i,j) = \begin{cases} C_s e^{(r_s - d(i,j))/F_s} \hat{\mathbf{k}} & \text{if } d(i,j) \leq r_s \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

where $d(i,j)$ is now the scalar distance between robot i and robot j calculated as

$$d(i,j) = \|\mathbf{x}(i,k) - \mathbf{x}(j,k)\|. \tag{4.6}$$

We maintain C_s and F_s as exploration parameters and $\hat{\mathbf{k}}$ as a unit vector in the direction of the line of impact for the collision. Here we note (4.5) also serves as a dispersion model to ensure robots are generally directed further into new regions of the environment.

4.4.2 Rescuer Model

Maintaining our base assumptions of a disaster scenario, the human rescuer likewise has a limited visibility range and cannot modify the behavior of the robots. The rescuer can travel at a maximum speed of s_{max}, has an observation radius of r_h, and position $\mathbf{x}_h(k)$ at time step k. During

time step k, the rescuer calculates their desired velocity as

$$\mathbf{v}_{d,h}(k) = v_{obs}(k) + \sum_{w=1}^{W} f_{hw}(w) \tag{4.7}$$

where $v_{obs}(k)$ is a velocity inferred from local observations of the swarm behavior. The second term in (4.7) prevents the rescuer from colliding with the wall, analogous to (4.4) but with defined scalars

$$A_h = 0.1 s_{max} \tag{4.8}$$

$$B_h = 2 s_{max} / \log \frac{2 s_{max}}{A_h}. \tag{4.9}$$

By the formulation of (4.7), the rescuer's desired velocity is primarily determined using observations of the swarm. We implement a simple flow model for the rescuer to explore how changes in the emergent swarm behavior influence the utility of observable swarm properties. More specifically,

$$v_{obs}(k) = s_{max} \frac{\mathbf{v}_e(k)}{\|\mathbf{v}_e(k)\|} \tag{4.10}$$

where

$$\mathbf{v}_e(k) = \sum_{i=1}^{M} \mathbf{v}_s(i,k) \; \forall i \in M \text{ s.t.}$$

$$\|\mathbf{x}(i,k) - \mathbf{x}_h(k)\| \leq r_h. \tag{4.11}$$

Finally, the rescuer cannot exceed their maximum speed, s_{max}, nor will they act if the inferred velocity from swarm observation is below some threshold magnitude, s_{min}, so the actual implemented velocity at iteration k, denoted as $v_h(k)$, becomes

$$v_h(k) = \begin{cases} s_{max} \frac{\mathbf{v}_{d,h}(k)}{\|\mathbf{v}_{d,h}(k)\|} & \text{if } s_{max} < \|\mathbf{v}_{d,h}(k)\| \\ 0 & \|\mathbf{v}_{d,h}(k)\| < s_{min} \\ \mathbf{v}_{d,h}(k) & \text{otherwise} \end{cases} \tag{4.12}$$

with an additional limit of $\pi/4$ on the maximum angle change the rescuer will experience between time steps unless they are near a wall boundary.

Equation (4.11) leverages the pattern recognition ability of people and demonstrated in studies like those by Walker et al. [44]. It also introduces an important but challenging aspect of autonomous exploration which is determining the 'optimal' amount of time to wait for information before acting.

Swarm interactions encode important environmental information like the presence of openings [54]. The encoding process changes properties of the emergent behavior but takes an amount of time depending on the robot density and distance to obstacles. We introduced two simple parameters to explore when the rescuer should begin moving in the environment using local observations of the swarm. The s_{min} parameter specifies a minimum magnitude of observed velocity the rescuer must observe before acting and represents a coarse consensus within the swarm about a desired direction. The second parameter, PT, is a specified pause time wherein the rescuer allows the robots time to interact before observing the emergent velocity.

4.4.3 Environment Description

Our goal in this work is to initiate a baseline quantification of swarm parameters so we defined a starting environment, explored the simulated performance, then systematically added features to gain insight into how different parameters influenced the emergent behavior. Figure 4.1 presents the four main environments discussed in this work. All of the environments are built around a central, square room that is four units wide. The size of the environment serves as a scaling factor which directed the choice of other exploratory parameters like maximum speed and sensor radius. The simulated robot swarm and rescuer are initially positioned centrally in this room and cannot

pass through environment boundaries. Robots continue to move about the environment according to (4.3) unless they experience a failure or are within a sensor radius of the survivor's location at which point the robot is removed from the simulation.

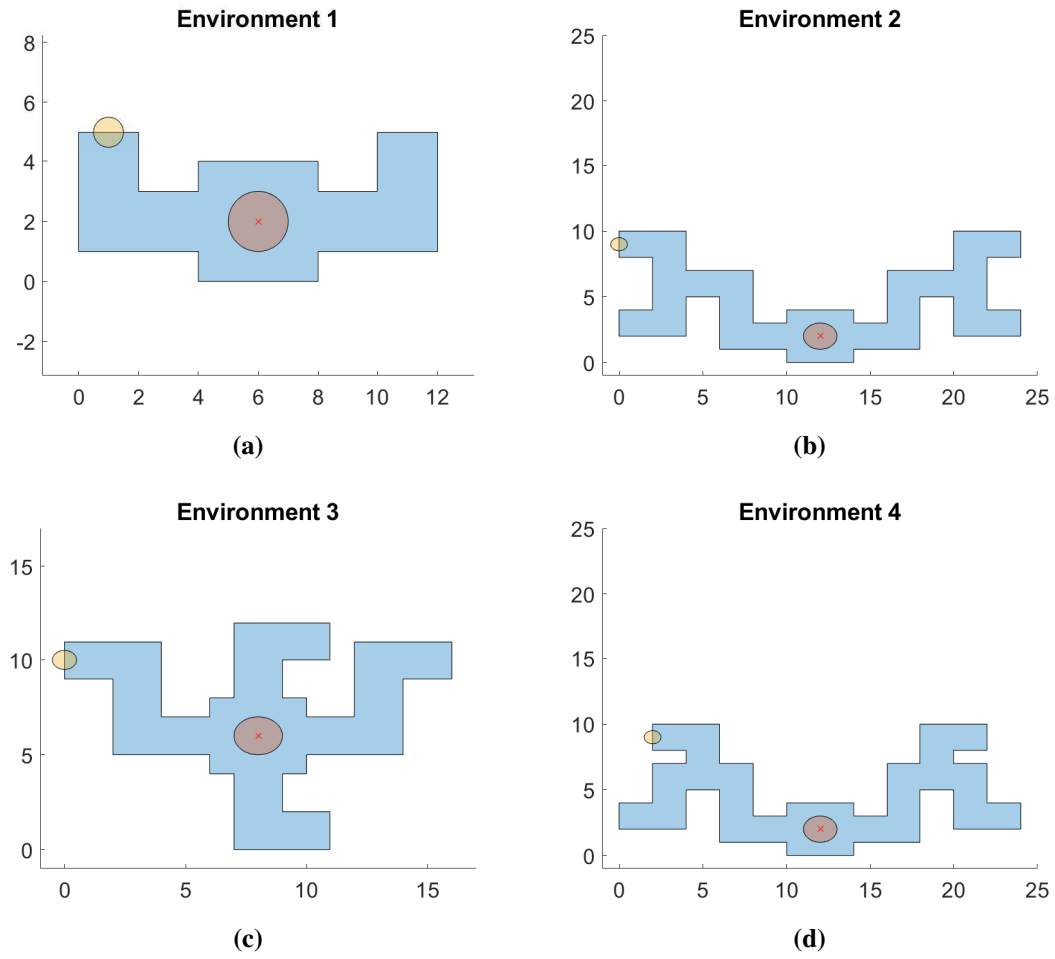


Figure 4.1: A lone survivor is placed at the end of the top, left-most hallway for each of the four simulated disaster environments, denoted as the yellow circle. The rescuer begins in the center of the environment but has limited sensing as indicated by the red circle.

4.5 Results

4.5.1 Preliminary Parameter Characterization

For the initial swarm parameter evaluation, we focused on qualitatively assessing the reasonableness of the emergent behavior by animating the swarm exploration process and quantitatively recording the percentage of area explored in a fixed time. We initially simulated the robots in *Environment* 1. As indicated in Fig. 4.1, the survivor is placed at the end of the west hallway. Robots stop when they are in the vicinity of the survivor. After identifying a reasonable initial operating range for all parameters, we proceeded to systematically vary one parameter at a time and evaluate the resulting swarm performance. The experiments confirmed several intuitive correlations:

- Increasing the number of robots increased the area explored but with diminishing rate of return so a very large number of robots was needed to generate even a small increase in exploration area
- Increased robot speed similarly increased area explored but again with diminishing returns
  - Too large a speed resulted in unnatural motion, disproportionate to environment size
- The number of past movements stored had a negligible effect on the area explored when there were robot interactions
- More area was explored when the robots' motion was less random
  - To represent realistic motion, a small value of σ should be incorporated into the robot motion
  - The presence of other robots acted to restrict sporadic motion due to collision avoidance
- Larger sensor radius generally improved performance.

From this preliminary set of experiments, we defined base values for the swarm parameters, summarized in Table 4.1, that created a relatively fluid and efficient dispersion in the environment.

Table 4.1: Summary of Simulation Parameters

Parameter	Significance	Base Value
M	Number of robots in swarm	300
s_s	Maximum robot speed	0.4
n	Number of past movements stored	1
σ	Std. dev. for Gaussian noise in motion	0.1
r_o	Robot's sensing radius	0.8
A_s, B_s	Force coefficient for obstacles	1, 0.5
C_s, F_s	Force coefficient for neighboring robots	1, 0.5

4.5.2 Resilience to Rescuer Variation

Using the baseline swarm parameter values summarized in Table 4.1, we ran 100 different swarm dispersion scenarios for the environments in Fig. 4.1. A maximum exploration time of 3000 steps was imposed on all scenarios to ensure the simulation time remained tractable. We then simulated a rescuer who navigates the unknown environment according to (4.12) in an effort to reach the lone survivor. Each rescuer parameter - threshold speed (s_{min}), pause time (PT), sensor radius (r_h), and maximum speed (s_{max}) - was systematically varied to explore the resiliency of the swarm and resulting rescue strategy with respect to variations in the fundamental rescuer behavior. Our priorities in evaluating the rescuer performance align with real-world disaster scenarios: ensuring the rescuer successfully locates the survivor as reliably as possible, minimizing the danger to the rescuer by reducing the number of steps they take in a potentially dangerous environment, and reducing the time it takes for the rescuer to locate the survivor.

Systematically varying each rescuer parameter revealed fundamental relationships between the properties of a successful rescue and the local emergent behavior. Variation in s_{min} had negligible impact on our evaluation criteria but the rescuer often required fewer steps to locate the lone survivor when the PT was first increased. The median rescue times for each environment and a range of PTs are plotted in Figure 4.2 along with a fitted curve to help visualize the overall trend. Pause times around 300 were generally most beneficial for reducing the number of steps the rescuer needed to locate the survivor, independent of environment. Further increases to the PT did not notably reduce the number of steps and instead only increased the rescue time.

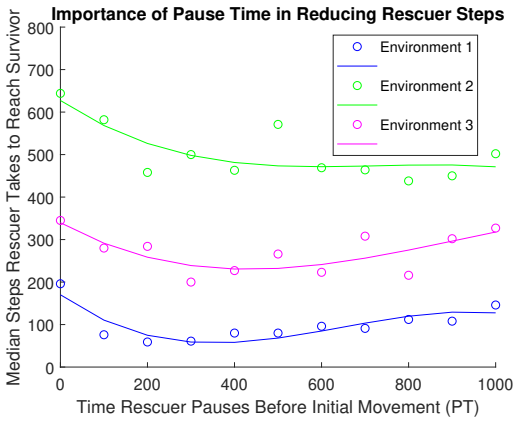


Figure 4.2: The rescuer requires fewer steps to locate the lone survivor if they first pause and allow the robots to do some preliminary exploration. The median number of steps taken by the rescuer over 100 simulations shows a PT of 300 was generally the most beneficial but there is a range of comparable values indicating resilience to rescuer PT.

Decreasing the number of rescuer steps is partially correlated with reduced danger to the rescuer, an important priority in disaster scenarios, but waiting for the robots to do preliminary exploration also adds time to the overall rescue which is undesirable. The trade-off between acting early or waiting for more information is not new but disaster scenarios add the challenge of a potentially dynamic environment where uncertainty is never fully resolved. Fortunately, rescuers are highly trained to make decisions based on the specific situation so the results of Fig. 4.2 can be used to primarily inform real-world rescue strategies. The informative role of the swarm exploration also aligns with the preferred swarm interaction expressed by participants in the survey by Carrillo-Zapata et al. [55].

Speed is a similarly important parameter to ensure the rescuer fully leverages information from the swarm. Figure 4.3 shows the median time required to locate the survivor as a function of rescuer speed for the first three environments from Fig. 4.1. The survivor is located a specific distance away from the center of the environment where the rescuer is initially positioned so, theoretically, a faster speed would result in decreasing the lower bound on rescue time. We see the benefits of travelling faster are initially present in Fig. 4.3 but speeds faster than the swarm speed of 0.4 do not significantly improve performance in any of the environments.

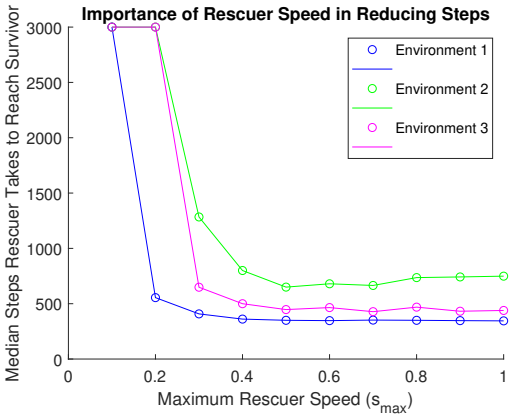


Figure 4.3: The rescuer often requires the fewest number of steps to locate the survivor when they have the same maximum speed as the robots. Here the median rescuer step count is shown when robots have a maximum speed of 0.4. Travelling faster than the swarm does not offer improved performance but indicates a resiliency to variations in rescuer speed.

While the utility of the swarm was resilient to variations in rescuer parameters, the rescuer generally required fewer steps to locate the survivor if they implemented a sufficient pause time (PT) and a reasonable speed (s_{max}). Another important parameter is the number of robots present in the swarm to ensure an informative behavior emerges. Figure 4.4 demonstrates the effect of increasing the number of robots in *Environment* 4 on the average rescue time. A sufficient number of robots are clearly needed - the rescuer never located the survivor when only one robot was present and required more than 2400 time steps on average with 50 robots - but *sufficient* is difficult to define in advance, particularly when the environment itself is unknown as is the case for most disaster scenarios. Fortunately, the rescue performance for our minimalist scenario is relatively robust with respect to variations in swarm size. Swarm sizes of $200 - 600$ robots all resulted in very similar average rescue times as summarized in Fig. 4.4 for *Environment* 4.

4.5.3 Resilience to Environment Hazards

The harsh terrain of many disaster scenarios results in a high probability of failure for autonomous systems. We have shown that the utility of the emergent behavior is robust with respect to initial swarm size but, alternatively, these results can be extended to scenarios where a large number of robots fail during the exploration task. To more fully characterize the impact of robot

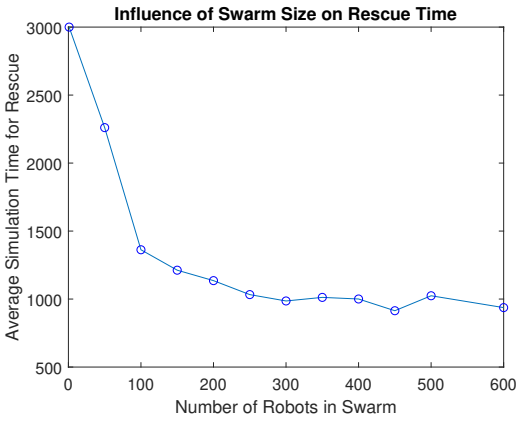


Figure 4.4: The rescuer required less time on average once more than 300 robots were initially distributed because a sufficient number of robot interactions occurred for informative behaviors to emerge. Using more than 300 robots did not significantly reduce the rescue time as demonstrated by the average rescue times for *Environment* 4.

failure on our hypothetical disaster rescue scenario, we introduce a failure parameter, p_f, to our simulation model. Prior to moving, a robot will be assigned a uniformly generated random number R. The robot is removed from the simulation if $R \leq p_f$ to simulate a realistic, immobilizing failure.

The lower curve in Figure 4.5 shows the median time required to locate the survivor in *Environment* 4 as p_f was systematically increased. The top curve indicates how many of the 100 scenarios resulted in the rescuer successfully locating the survivor. For all 100 scenarios, 300 robots are initially dispersed, travelling at 0.4 units, and a rescuer PT of 300 time steps.

It is instructive to compare the performance of the swarm in the presence of catastrophic failure from Fig. 4.5 to the results for varying the initial swarm size in Fig. 4.4. A p_f of 0.0014 resulted in the rescuer taking a median of 639 steps. The swarm experienced a 75% failure rate over the course of the simulation so 225 of the original 300 robots failed by the time the rescuer successfully located the survivor. Although there were only 75 robots operating in the environment near the end of the simulation, the surviving robot behaviors had still been influenced by interactions with other robots pre-failure. Information about the environment was distributed through the swarm by these interactions so, even with mass failure, the rescuer could still leverage the swarm knowledge to locate the survivor. Interpolating the values from Fig. 4.4, the median time for the rescuer would

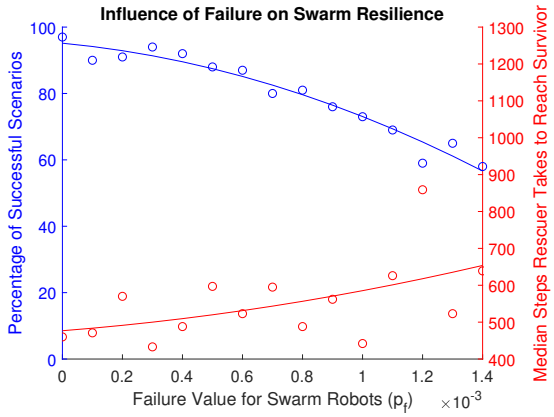


Figure 4.5: The median rescue time in *Environment* 4 (lower red curve) increased as robots experienced larger failure rates but the rescuer still successfully located the survivor in the majority of scenarios (top blue curve) even when as few as 12 robots remained functional.

have been about 2000 steps if the swarm was initially composed of only 75 robots and there were no failures. Amazingly, the rescuer successfully reached the survivor as long as at least 12 of the original 300 robots were still present in the environment which occurred in 58 of the 100 scenarios.

To further put the swarm resilience into context, we consider the theoretical performance of a single robot that primarily operates using a wall-following strategy. *Environment* 4 has a perimeter of 104 units so a wall-following robot would have an expected path length of 52 units or, enforcing the same speed, we expect the wall-follower to need 130 steps on average to locate the lone survivor. The wall-following robot is also a singular entity so the rescuer will only successfully locate the survivor if the robot itself does not fail but each move in a harsh environment adds a chance for failure. We model the probability, p_n, of the wall-following robot successfully moving n steps by using the exponential decay model

$$p_n = e^{-cn} \tag{4.13}$$

where c is the probability of failure per unit step. Evaluating (4.13) with $n = 130$, the wall-following robot would theoretically need to achieve a 99.58% reliability per step to locate the lone survivor in *Environment* 4 in as many trials as our minimalist swarm where 75% of the robots failed and the survivor was still successfully located.

Even with 100% step reliability, a wall-following strategy may simply be ineffective for the particular environment of interest due to a physical lack of boundaries or extremely reduced visibility which even limits the abilities of human rescuers [55]. We further explore the swarm's resilience to environmental variations by adding a room, or cavern, to one end of *Environment* 4 and placing the lone survivor at the center of this room. The room dimensions are such that neither the robots nor the rescuer can sense the survivor without losing contact with a wall. Figure 4.6 shows the new environment as well as the resulting rescuer trajectory for one illustrative simulation. The trajectory shows the rescuer sometimes circles back on their previous path which adds steps to the total that would not occur with a wall-following algorithm; however, the trajectory also shows the rescuer avoided entering several unoccupied regions of the environment, instead spending extra time in areas that must necessarily be entered in order to reach the survivor.

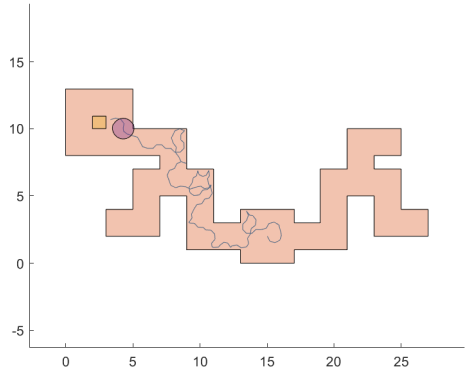


Figure 4.6: In the cavern environment, the survivor (yellow square) is positioned in the middle of a large room where limited sensing prevents both the rescuer and robots from seeing the survivor while maintaining contact with the wall. The survivor would therefore not be successfully located if the rescuer was alone, or reliant on a wall-following robot with comparable sensing radius, but the dispersion of robots in the swarm directs the rescuer away from the wall and to the survivor. The rescuer also avoids entering unnecessary regions of the environment in this scenario.

Our minimalist rescuer model only considers the velocities of robots within the rescuer's sensing radius but this simple algorithm leverages the emergent swarm behavior to reduce the likelihood of entering unnecessary regions of the environment. Consider the rescuer's decision making process when they approach the final hallway junction before entering the large room where the

survivor is located. Figure 4.7 zooms in on this region of the environment and shows the rescuer's position as a red 'x' with the sensing area illustrated with the purple circle. At this point, the rescuer has been unable to detect the upper hallway. A wall-following strategy would lead the rescuer into the dead-end hallway but several robots have already explored this region and are moving back up, creating a small repulsive wave that directs other robots as well as the rescuer away from the dead end in general. The velocity for each robot is shown as a green arrow. The repulsive force of the wall combined with the observable swarm motion at the single time step direct the rescuer away from a potentially hazardous region of the environment and up into the correct hallway.

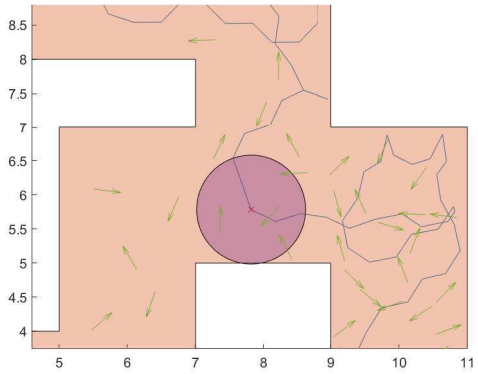


Figure 4.7: Some robots have already entered the left-most branch of the cavern environment and encountered a dead end. Zooming in on the final junction of the cavern environment and showing the robot velocities as green arrows, the returning robots exert a pressure that helps direct the rescuer, who can only detect objects located within the large purple circle, into the correct hallway and away from the unnecessary environment segment.

More sophisticated path-planning algorithms would reduce the amount of rescuer back-tracking but our focus in this work was on quantifying the resiliency of the emergent swarm behavior to perturbations. Even with the minimalist model used, the rescuer successfully located the lone survivor in our cavern environment in 83 out of 100 simulations and required a median of 676 steps. The rescuer also frequently avoided entering unnecessary regions of the environment, thereby further reducing the potential hazards encountered in real disaster scenarios. No change in the swarm be-

havior was needed to account for a significant change in the environment which further illustrates the swarm's resiliency, this time with respect to environmental features.

4.6 Discusison

While we present specific parameter values in Section 4.5, we are by no means proposing these as ideal scalars for swarm exploration. Nor are we advocating a simplistic interaction between swarm and simulated rescuer. The real contribution of this work is establishing a baseline quantification of swarm resilience, demonstrating important relationships between robots and a minimalist human rescuer, and illustrating the feasible benefits of leveraging emergent properties for important applications like locating survivors in disaster environments.

Interactions are the primary force driving emergent swarm behavior so a sufficient number of robots are necessary to ensure collective properties evolve. While this statement is partially intuitive and also supported by Fig. 4.4, the influence of swarm size on individual robot properties is potentially less clear but an extremely important design consideration. We used values from Table 4.1 to govern the dispersion algorithm. The choice of coefficients was impressively resilient to variations but testing minimum swarm sizes highlights the value of interactions for producing more complex behaviors.

For example, during our preliminary investigations, we found that an individual robot did not need to store past steps or implement sophisticated heading control for efficient area coverage if other robots were in the sensing area. Neighboring robots in the swarm essentially serve as a motion memory and enforce more direct trajectories. When one robot moved, other robots could fill in the space to prevent the robot from undoing its move which is the traditional role of 'memory'. Similarly, an individual robot may have very random motion (large value of σ in our simulation) but the presence of other robots combined with a simple collision avoidance mechanism reduced the robot's erratic trajectory compared to the robot's trajectory in isolation. The collective behavior generally resulted in more efficient local behaviors without any change to the individual robot parameters. Figure 4.8 qualitatively demonstrates the improved trajectory for a robot in a swarm

with $M = 300$ robots (red line) compared to a robot operating on its own (blue line) with the same values from Table 4.1.

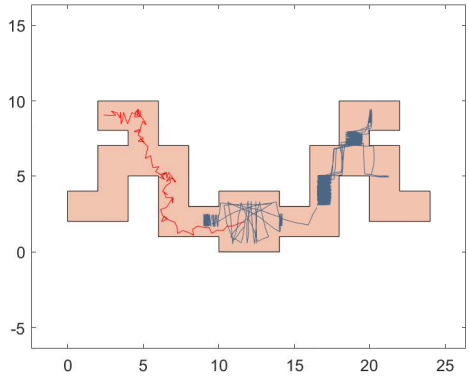


Figure 4.8: A robot operating with the parameters of Table 4.1 with swarm size $M = 1$ ineffectively navigates the environment (blue line) but increasing the swarm size improves exploration trajectories without any change to the local rules. The red line is an illustrative robot trajectory when $M = 300$.

Robot interactions also contribute to the desired robustness frequently associated with swarms because information is stored implicitly in the behaviors rather than explicitly with any single robot. Figure 4.5 demonsrates that the swarm could experience catastrophic failure but still retain sufficient information for the rescuer to leverage and successfully locate the lone survivor. Of the 58 scenarios with $p_f = 0.0014$ where the rescuer successfully reached the survivor, the fewest number of robots still functioning in the environment was just 12 but the motion of those 12 robots had benefitted from the initial distribution of 300 robots.

The surviving robots were generally guided along a more direct path to the survivor by other robots that had entered dead end hallways and were attempting to return before failing. The rescuer similarly benefitted from the exploration of failed robots before they were immobilized. One challenge from our implementation is that the rescuer *required* a minimum observed velocity from the swarm before moving. Thus, in the event that all robots within the rescuer's sensing radius failed, the rescuer was unable to move. In a real-world disaster scenario, the swarm would most likely only inform the rescuer's search strategy so even in the extreme cases where no robots are in the rescuer's sensing radius, the rescuer can still navigate the environment.

4.7 Conclusion

The distributed nature of swarms offers an intuitive promise of robustness and efficiency which is especially desirable for exploration-based tasks in harsh environments like locating disaster survivors. In this work, we establish a baseline quantification of swarm resilience by first evaluating the impact of fundamental robot properties on the utility of the emergent behavior. A simulated human rescuer relies on the locally observable swarm velocity to navigate an unknown disaster environment while attempting to locate a lone survivor. We affirmed that a minimalist swarm governed by a simple collision avoidance algorithm had sufficiently complex interactions to encode important environmental information that the rescuer successfully leveraged. The utility of the swarm was robust with respect to variations in fundamental rescuer parameters and significant changes in the environment. More impressively, the rescuer still found the survivor when the swarm underwent catastrophic failure and lost up to 288 of the original 300 robots.

Although simple, the robust success of a simulated rescuer locating a lone survivor affirms the benefits of robot swarms. Local observations of the emergent swarm behavior often reduced the number of regions the rescuer entered and even reliably directed the rescuer into empty spaces that would be inaccessible if relying on limited-sensing strategies like wall-following. Properties of the swarm do not need to be optimized for the swarm to still be effective in a variety of environments. While our simulation established an important baseline, we believe more information can be extracted by a human observing the emergent behavior and the additional information will further improve the swarm performance in all interactive applications.

Chapter 5

Concluding Remarks and Future Work

From the earliest conceptions, robots have often taken human forms with a predilection for individual sophistication. Humanoid robot designs partially turned these preferences into reality with impressive demonstrations, including Boston Dynamics' Atlas robot doing parkour, yet a noticeable lack of robotic presence remains in disaster rescue scenarios despite technological advancements. The increase in occurrence and economic impact of disaster scenarios [39] demands solutions and robots can still be a part of that solution but perhaps in a form more analogous to ants than humans!

Robotic swarms have many potential benefits in exploration-based tasks, including aiding in disaster rescue scenarios, as demonstrated in nature by cooperative systems including insect colonies, schools of fish, and flocking birds. In all of these systems, local interactions governed by individual rules contribute to a more complex, cooperative behavior. The emergent behavior can lead to efficient foraging strategies or complex constructions which exceed the abilities of any single swarm member. Though evidence of desirable emergent behaviors exists, the local mechanisms which lead to those emergent behaviors is generally unintuitive, leading to a very fundamental challenge in the design of robotic swarms. No closed-form mapping exists between local and emergent behaviors within a swarm.

Some researchers have attempted to resolve the correlation between local and emergent behaviors using a top-down approach wherein a desired emergent behavior is described mathematically and then the coefficients for individual robot controllers are iteratively optimized to reproduce the desired emergent behavior [8], [11], [12]. The top-down approach can be effective in controlled settings but undermines the overall potential of swarms. First, the resulting behaviors are no longer robust because changes in the swarm composition or environment fundamentally alter the interactions and hence change the emergent behavior. The top-down strategies also fail to leverage the interaction element of swarms and instead impose more sophisticated controllers at the local level

which is similar to the humanoid robot strategies which thus far have been ineffective in harsh environments.

This dissertation takes a fundamentally different approach by establishing bottom-up swarm strategies with minimalist swarms. In Chapter 2, we use a continuum limit methodology to map discrete-time robot actions to a partial differential equation (PDE) model which describes the distribution of robots in the environment as a function of time. The robots represent a worst-case model because they were reduced to random motion with no communication or localization abilities. Nonetheless, the minimalist swarm encoded enough information about the environment for a centrally located observer to determine the location of openings in a $1D$ hallway or $2D$ square room. The observer would compare the locally observed distribution of robots to the PDE model for each potential environment and classify the environment using a least-square error strategy.

Explicitly modelling each potential environment is not practical in real-world disaster scenarios where the environment has been altered; however, the mathematically-based PDE formulation established an important correlation between local robot behaviors, environmental features, and resulting emergent swarm behavior which we then expanded on in Chapter 3. A simple neural network with a soft max activation function was trained to correlate locally observed robot distributions with the environment being explored by the swarm. Once again, this bottom-up swarm approach used simple robots and leveraged the emergent behavior to classify environment features. The trained neural network reliably predicted the location of single openings in the environment and was demonstrably robust. The network could be trained using centrally located doorways in a square $2D$ office but then accurately identify which wall contained the doorway when the simulated environment actually shifted the door to the far side. The minimalist swarm could also experience catastrophic failure, losing 90% of the original robots, and still be used to predict the doorway location with better than random accuracy. We demonstrate the value of this robustness in Chapter 3 by introducing a theoretical person in the center of the office who uses the observed swarm density to locate the single doorway with better than random accuracy even with 90% robot failure.

Chapter 4 further characterizes the robustness of bottom-up swarm strategies for disaster environments and extends the swarm-human interaction scenarios. First, we affirm the robustness of the swarm with respect to variations in local robot characteristics. Robots within the swarm consistently dispersed throughout the environment despite large changes in individual speed, heading control, and swarm size. Building on studies where human participants are asked to classify simulated emergent swarm behaviors from a list of established behaviors [44], [45], [46], we introduced a simulated human rescuer who was only able to discern the local robot velocities. Using this minimalist information, the rescuer was able to reliably locate a survivor in the environments. The rescuer could utilize a variety of speeds themself and change how long they allowed the robots to explore before acting without significantly changing the amount of time needed to reach the survivor. Further, the swarm consistently helped direct the rescuer away from dead end passages which is advantageous in real-world environments where conditions are unstable. No change in strategy was needed even when the environment was fundamentally changed to include a large open room which would preclude the use of wall-following strategies. Once again, the swarm was quantifiably robust to individual robot failure. The survivor was located when as few as 12 of the original 300 robots remained.

The results of the foundational work presented in this dissertation validate the potential for implementing swarms to aid in the exploration of harsh environments. By leveraging fundamental interactions between robots and the environment rather than imposing top-down rules, we demonstrated that even minimalist robots can encode key features of unknown environments. PDE models quickly and accurately described the emergent behavior for a simple environment. Neural networks expanded the variety of environmental features which could be classified. More than classifying a full environment, we demonstrated how rescuers can use local observations of the emergent swarm behavior to aid in locating disaster survivors and navigating unknown environments.

We focused on minimalist robots throughout this work. The simple robots serve as a worst-case model but also establish a baseline performance to which more sophisticated sensor additions and algorithms can be compared. Future work building on this foundation should include physical

implementations and potentially additional complexity at the local robot level. Before adding functionality to the robots, however, properties of the emergent behavior should be investigated further. We only utilized density or velocity properties from the swarm in our work because the focus was on the swarm itself but this methodology affirmed that beneficial information was encoded in the emergent behavior. All of the available information from the swarm should be extracted prior to increasing the complexity of robot interactions so the most informative emergent properties can be identified for the desired tasks.

Robots have advanced significantly since their inception but are still unable to reliably navigate harsh terrain like collapsed buildings or mines. Perhaps applications like interplanetary exploration and especially disaster rescues would benefit from a new approach inspired by nature's cooperative systems. Robotic swarms leverage interactions between relatively simple robots and the environment to generate more complex emergent behaviors. As asserted in this dissertation, the emergent behavior can encode important environmental features to aid in navigation and the swarm itself is robust to changes in its fundamental properties, the environment, and can even withstand catastrophic failures. Swarms may not look like the automatic assistants described by Karel Čapek in 1920 but they may be the key advancement to support rescuers in the harsh terrain of a disaster.

Bibliography

[1] Robot. https://en.wikipedia.org/wiki/Robot#cite_note-6. Accessed: 2020-03-28.

[2] CBC Radio. Robots are piling up inside fukushima's robot graveyard. https://www.cbc.ca/radio/quirks/one-minute-of-exercise-fukushima-robots-fail-dna-stores-everything-but-the-kitchen-sink-1.4019376/robots-are-piling-up-inside-fukushima-s-robot-graveyard-1.4019409, March 2017. Accessed: 2020-03-28.

[3] Defense Advanced Research Projects Agency. Darpa robotics challenge (drc). https://www.darpa.mil/program/darpa-robotics-challenge. Accessed: 2020-03-28.

[4] Iñaki Navarro and Fernando Matía. An introduction to swarm robotics. *ISRN Robotics*, 2013, 2012.

[5] Brian P. Gerkey and Maja J. Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–954, 2004.

[6] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: A review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, March 2013.

[7] Erol Şahin. Swarm robotics: From sources of inspiration to domains of application. In *International Workshop on Swarm Robotics*, pages 10–20. Springer, 2004.

[8] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson. Distributed event-triggered control for multi-agent systems. *IEEE Transactions on Automatic Control*, 57(5):1291–1297, May 2012.

[9] S. Berman, V. Kumar, and R. Nagpal. Design of control policies for spatially inhomogeneous robot swarms with application to commercial pollination. In *2011 IEEE International Conference on Robotics and Automation*, pages 378–385, May 2011.

[10] S. Berman, A. Halasz, M. A. Hsieh, and V. Kumar. Optimized stochastic policies for task allocation in swarms of robots. *IEEE Transactions on Robotics*, 25(4):927–937, Aug 2009.

[11] L. E. Barnes, M. A. Fields, and K. P. Valavanis. Swarm formation control utilizing elliptical surfaces and limiting functions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(6):1434–1445, Dec 2009.

[12] K. Elamvazhuthi and S. Berman. Optimal control of stochastic coverage strategies for robotic swarms. In *2015 IEEE International Conference on Robotics and Automation*, pages 1822–1829, May 2015.

[13] M. Berger, L. M. Seversky, and D. S. Brown. Classifying swarm behavior via compressive subspace learning. In *2016 IEEE International Conference on Robotics and Automation*, pages 5328–5335, May 2016.

[14] Y. Zhang, E. K. P. Chong, J. Hannig, and D. Estep. Approximating extremely large networks via continuum limits. *IEEE Access*, 1:577–595, 2013.

[15] Roderick Matheson Logan and RE Stickney. Simple classical model for the scattering of gas atoms from a solid surface. *The Journal of Chemical Physics*, 44(1):195–201, 1966.

[16] R. L. Sturdivant and E. K. P. Chong. The necessary and sufficient conditions for emergence in systems applied to symbol emergence in robots. *IEEE Transactions on Cognitive and Developmental Systems*, 10(4):1035–1042, Dec 2018.

[17] Megan Emmons, Anthony A. Maciejewski, and Edwin K. P. Chong. Modelling emergent swarm behavior using continuum limits for environmental mapping. In *Proc. IEEE International Conference on Control and Automation*, pages 86–93, June 2018.

[18] C. A. C. Parker and H. Zhang. Cooperative decision-making in decentralized multiple-robot systems: The best-of-n problem. *IEEE/ASME Transactions on Mechatronics*, 14(2):240–251, April 2009.

[19] Julia T. Ebert, Melvin Gauci, and Radhika Nagpal. Multi feature collective decision making in robot swarms. In *Proc. 17th International Conference on Autonomous Agents and Multiagent Systems*, July 2018.

[20] M. Schneider-Fontan and M. J. Mataric. Territorial multi-robot task division. *IEEE Transactions on Robotics and Automation*, 14(5):815–822, Oct 1998.

[21] Brian P. Gerkey and Maja J. Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–953, Sept. 2004.

[22] Maja J. Matarić, Gaurav S. Sukhatme, and Esben H. Østergaard. Multi-robot task allocation in uncertain environments. *Autonomous Robots*, 14(2):255–263, March 2003.

[23] Dieter Fox, Wolfram Burgard, Hannes Kruppa, and Sebastian Thrun. A probabilistic approach to collaborative multi-robot localization. *Autonomous Robots*, 8(3):325–344, June 2000.

[24] Amanda Prorok, Alexander Bahr, and Alcherio Martinoli. Low-cost collaborative localization for large-scale multi-robot systems. In *Proc. IEEE International Conference on Robotics and Automation*, pages 4236–4241, May 2012.

[25] Jie Chen, Kian Hsiang Low, Yujian Yao, and Patrick Jaillet. Gaussian process decentralized data fusion and active sensing for spatiotemporal traffic modeling and prediction in mobility-on-demand systems. *IEEE Transactions on Automation Science and Engineering*, 12(3):901–921, July 2015.

[26] Alessandro Giusti, Jawad Nagi, Luca Gambardella, and Gianni A. Di Caro. Cooperative sensing and recognition by a swarm of mobile robots. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 551–558, Oct 2012.

[27] Kian Hsiang Low, Wee Kheng Leow, and Jr. Marcelo H. Ang. Autonomic mobile sensor network with self-coordinated task allocation and execution. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 36(3):315–327, May 2006.

[28] Saptarshi Bandyopadhyay, Soon-Jo Chung, and Fred Y. Hadaegh. Probabilistic and distributed control of a large-scale swarm of autonomous agents. *IEEE Transactions on Robotics*, 33(5):1103–1123, Oct 2017.

[29] A. Dirafzoon and E. Lobaton. Topological mapping of unknown environments using an unlocalized robotic swarm, Nov 2013.

[30] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436, 2018.

[31] Julian Morelli, Pingping Zhu, Bryce Doerr, Richard Linares, and Silvia Ferrari. Integrated mapping and path planning for very large-scale robotic (vlsr) systems. In *2019 International Conference on Robotics and Automation*, pages 3356–3362, May 2019.

[32] L. Hou, F. Fan, J. Fu, and J. Wang. Time-varying algorithm for swarm robotics. *IEEE/CAA Journal of Automatica Sinica*, 5(1):217–222, Jan 2018.

[33] K. Elamvazhuthi, H. Kuiper, and S. Berman. Pde-based optimization for stochastic mapping and coverage strategies using robotic ensembles. *Automatica*, 95:356–367, Sept. 2018.

[34] Francesco Mondada, Michael Bonani, Xavier Raemy, James Pugh, Christopher Cianci, Adam Klaptocz, Stephane Magnenat, Jean-Christophe Zufferey, Dario Floreano, and Alcherio Martinoli. The e-puck, a robot designed for education in engineering. *Proc. 9th Conference on Autonomous Robot Systems and Competitions*, 1(1):59–65, 2009.

[35] Francesco Mondada, Edoardo Franzi, and Paolo Ienne. Mobile robot miniaturisation: A tool for investigation in control algorithms. *Experimental Robotics III*, 200:501–513, 2009.

[36] Michael Rubenstein, Christian Ahler, and Radhika Nagpal. Kilobot: A low cost scalable robot system for collective behaviors. In *Proc. IEEE International Conference on Robotics and Automation*, pages 3293–3298, May 2012.

[37] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2013.

[38] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Proc. International Symposium on Computational Intelligence in Robotics and Automation*, pages 146–151, July 1997.

[39] Matteo Coronese, Francesco Lamperti, Klaus Keller, Francesca Chiaromonte, and Andrea Roventini. Evidence for sharp increase in the economic damages of extreme natural disasters. *Proceedings of the National Academy of Sciences*, 116(43):21450–21455, 2019.

[40] Gabriele Valentini, Anthony Antoun, Marco Trabattoni, Bernat Wiandt, Yasumasa Tamura, Etienne Hacquard, Vito Trianni, and Marco Dorigo. Kilogrid: A novel experimental environment for the kilobot robot. *Swarm Intelligence*, 12:245–266, 2018.

[41] Mathieu Le Goc, Lawrence H. Kim, Ali Parsaei, Jean-Daniel Fekete, Pierre Dragicevic, and Sean Follmer. Zooids: Building blocks for swarm user interfaces. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, page 97–109. Association for Computing Machinery, 2016.

[42] Daniel Pickem, Paul Glotfelter, Li Wang, Mark Mote, Aaron Ames, Eric Feron, and Magnus Egerstedt. The robotarium: A remotely accessible swarm robotics research testbed. In *IEEE International Conference on Robotics and Automation*, pages 1699–1706, 2017.

[43] Ian Buckley and Magnus Egerstedt. Infinitesimal shape-similarity for characterization and control of bearing-only multirobot formations. *IEEE Transactions on Robotics*, pages 1–15, 2021.

[44] Phillip Walker, Michael Lewis, and Katia Sycara. Characterizing human perception of emergent swarm behaviors. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC) 2016*, pages 002436–002441, 2016.

[45] David St-Onge, Florent Levillain, Elisabetta Zibetti, and Giovanni Beltrame. Collective expression: How robotic swarms convey information with group motion. *Paladyn, Journal of Behavioral Robotics*, 10(1):418–435, 2019.

[46] Maria Santos and Magnus Egerstedt. From motions to emotions: Can the fundamental emotions be expressed in a robot swarm. *International Journal of Social Robotics*, 2020.

[47] Robin R. Murphy, Jeffery Kravitz, Samuel L. Stover, and Rahmat Shoureshi. Mobile robots in mine rescue and recovery. *IEEE Robotics Automation Magazine*, 16(2):91–103, 2009.

[48] Evan Ackerman. Why robots can't be counted on to find survivors in the florida building collapse, Jul 2021.

[49] Jennifer Carlson, Robin R. Murphy, and Andrew Nelson. Follow-up analysis of mobile robot failures. In *IEEE International Conference on Robotics and Automation*, volume 5, pages 4987–4994 Vol.5, 2004.

[50] Amanda Bouman, Muhammad Fadhil Ginting, Nikhilesh Alatur, Matteo Palieri, David D. Fan, Thomas Touma, Torkom Pailevanian, Sung-Kyun Kim, Kyohei Otsu, Joel Burdick, and Ali akbar Agha-Mohammadi. Autonomous spot: Long-range autonomous exploration of extreme environments with legged locomotion. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2518–2525, 2020.

[51] Eri Takane, Kenjiro Tadakuma, Masahiro Watanabe, Masashi Konyo, and Satoshi Tadokoro. Design and control method of a planar omnidirectional crawler mechanism. *Journal of Mechanical Design*, 144(1), 07 2021.

[52] Xuesu Xiao, Jan Dufek, and Robin R. Murphy. Robot risk-awareness by formal risk reasoning and planning. *IEEE Robotics and Automation Letters*, 5(2):2856–2863, 2020.

[53] M.D.A. Terry Jack, S.A. Khuman, and K. Owa. Spatio-temporal patterns act as computational mechanisms governing emergent behavior in robotic swarms. *International Journal of Swarm Intelligence and Evolutionary Computation*, 8(1), 2019.

[54] Megan Emmons, Anthony A. Maciejewski, Charles Anderson, and Edwin K. P. Chong. Classifying environmental features from local observations of emergent swarm behavior. *IEEE/CAA Journal of Automatica Sinica*, 7(3):674–682, 2020.

[55] Daniel Carrillo-Zapata, Emma Milner, Julian Hird, Georgios Tzoumas, Paul J. Vardanega, Mahesh Sooriyabandara, Manuel Giuliani, Alan F. T. Winfield, and Sabine Hauert. Mutual shaping in swarm robotics: User studies in fire and rescue, storage organization, and bridge inspection. *Frontiers in Robotics and AI*, 7:53, 2020.

[56] Matteo Palieri, Benjamin Morrell, Abhishek Thakur, Kamak Ebadi, Jeremy Nash, Arghya Chatterjee, Christoforos Kanellakis, Luca Carlone, Cataldo Guaragnella, and Ali akbar Aghamohammadi. Locus: A multi-sensor lidar-centric solution for high-precision odometry and 3d mapping in real-time. *IEEE Robotics and Automation Letters*, 6(2):421–428, 2021.

[57] Sandeep A. Kumar, J. Vanualailai, B. Sharma, and A. Prasad. Velocity controllers for a swarm of unmanned aerial vehicles. *Journal of Industrial Information Integration*, 22:100198, 2021.

[58] Edmund R. Hunt, Conor B. Cullen, and Sabine Hauert. Value at Risk Strategies for Robot Swarms in Hazardous Environments. In Hoa G. Nguyen, Paul L. Muench, and Brian K. Skibba, editors, *Unmanned Systems Technology XXIII*, volume 11758, pages 158 – 177. International Society for Optics and Photonics, SPIE, 2021.