Classifying Environmental Features From Local Observations of Emergent Swarm Behavior

Megan Emmons, *Student Member, IEEE*, Anthony A. Maciejewski, *Fellow, IEEE*, Charles Anderson, *Senior Member, IEEE*, and Edwin K. P. Chong, *Fellow, IEEE*

***Abstract*—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.**

***Index Terms*—Biologically inspired robotics, environment exploration, multi-agent systems, swarm robotics.**

I. 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, i.e., 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 [1] 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., [2]) 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 [3]. 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

Manuscript received January 29, 2020; revised February 19, 2020; accepted March 6, 2020. Recommended by Associate Editor MengChu Zhou. *(Corresponding author: Megan Emmons.)*

Citation: M. Emmons, A. A. Maciejewski, C. Anderson, and E. K. P. Chong, "Classifying environmental features from local observations of emergent swarm behavior," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 3, pp. 674–682, May 2020.

M. Emmons, A. A. Maciejewski, and E. K. P. Chong are with the Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523 USA (e-mail: memmons14k@gmail.com; aam@colostate.edu; edwin.chong@colostate.edu).

C. Anderson is with the Department of Computer Science, Colorado State University, Fort Collins, CO 80523 USA (e-mail: chuck.anderson@colostate.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JAS.2020.1003129

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 II. We then describe the simulation test platform and introduce the neural network used to correlate observed robot distributions with training environments in Section III. Section IV evaluates the performance of the implemented network. We extend the simulations in Section V 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 VI.

II. 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 [4], [5], task allocation [6]–[8], and localization [9], [10] 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 [11], [12], or [13] 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 [14], are computationally expensive like [15] or [16], or imbue the robots with sophisticated sensory knowledge to construct individual maps as in [17]. 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., [18]–[21]). These approaches do produce the desired behaviors but therefore require first knowing what behavior is desired. 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 [22] and e-puck [23] 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 [24] is one promising research platform because the robots can be managed collectively.

III. Formulation of Test Environment

A. 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 [3]. 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, $\boldsymbol{x}$. Mathematically, the desired output from the trained neural network for a specific simulation run n can be formulated as

$$g_k(\boldsymbol{x}_n) = p(C = k|\boldsymbol{x}_n) \tag{1}$$

where the value of $g_k(\boldsymbol{x}_n)$ 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, $\boldsymbol{w}$, that can be trained to maximize the likelihood of the data distinguishing between all K potential environment classes. Introducing a function $f(\boldsymbol{x}_n, \boldsymbol{w}_k)$ and defining

$$g_k(\boldsymbol{x}_n) = \frac{f(\boldsymbol{x}_n, \boldsymbol{w}_k)}{\sum_{m=1}^{K} f(\boldsymbol{x}_n, \boldsymbol{w}_m)} \tag{2}$$

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

$$f(\boldsymbol{x}_n, \boldsymbol{w}_k) = \exp[\boldsymbol{w}_k^T \boldsymbol{x}_n] \tag{3}$$

to create a general softmax formulation [25]. 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

$$t_{n,k} = \begin{cases} 1, & \text{if } \boldsymbol{x}_n \text{ is from class } k \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

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, $L(\boldsymbol{w})$, is expressed as

$$L(\boldsymbol{w}) = \prod_{n=1}^{N} \prod_{k=1}^{K} g_k(\boldsymbol{x}_n)^{t_{n,k}}. \tag{5}$$

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 (5), but for computational efficiency we instead maximize the natural logarithm of the likelihood

$$\log L(\boldsymbol{w}) = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{n,k} \log g_k(\boldsymbol{x}_n). \tag{6}$$

Substituting (2) back into (6) and taking the gradient with respect to the weights of a single output neuron, j, we get

$$\begin{aligned} \nabla_{\boldsymbol{w}_j} \log L(\boldsymbol{w}) &= \sum_{n=1}^{N} \sum_{k=1}^{K} \frac{t_{n,k}}{g_k(\boldsymbol{x}_n)} \nabla_{\boldsymbol{w}_j} g_k(\boldsymbol{x}_n) \\ &= \sum_{n=1}^{N} (t_{n,j} - g_j(\boldsymbol{x}_n)) \boldsymbol{x}_n. \end{aligned} \tag{7}$$

Equation (7) is still nonlinear with respect to $\boldsymbol{x}$, so an iterative update process is needed to find which weights maximize the conditional probability in (2). This leads to a gradient ascent update rule for the weights of the form

$$\boldsymbol{w}_j(i+1) = \boldsymbol{w}_j(i) + \alpha \sum_{n=1}^{N} (t_{n,j} - g_j(\boldsymbol{x}_n)) \boldsymbol{x}_n \tag{8}$$

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.

B. 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 one-dimensional (1D) and two-dimensional (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 1D hallway and 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 [26].

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 on 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.

IV. Evaluation of Swarm Methodology

A. 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.

B. 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 (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. 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.

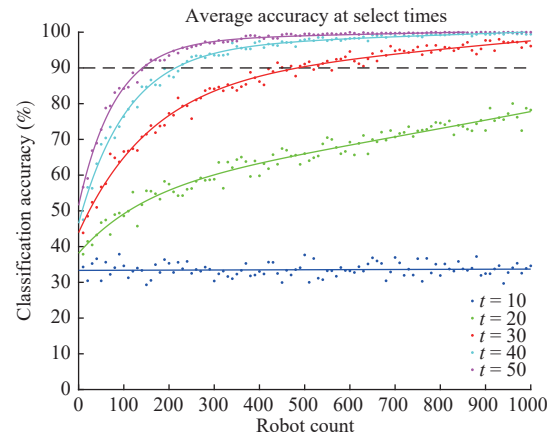


Fig. 1. A stronger correlation is achieved between locally observed robot density and the environment when more robots are initially distributed. In addition, the robots need a sufficient amount of time to interact with each other and the environment. For example, 90% of the environments were classified 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 returns 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. 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. 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. 2(b). It required observing the number of robots in bins 6 and 7 for about 25

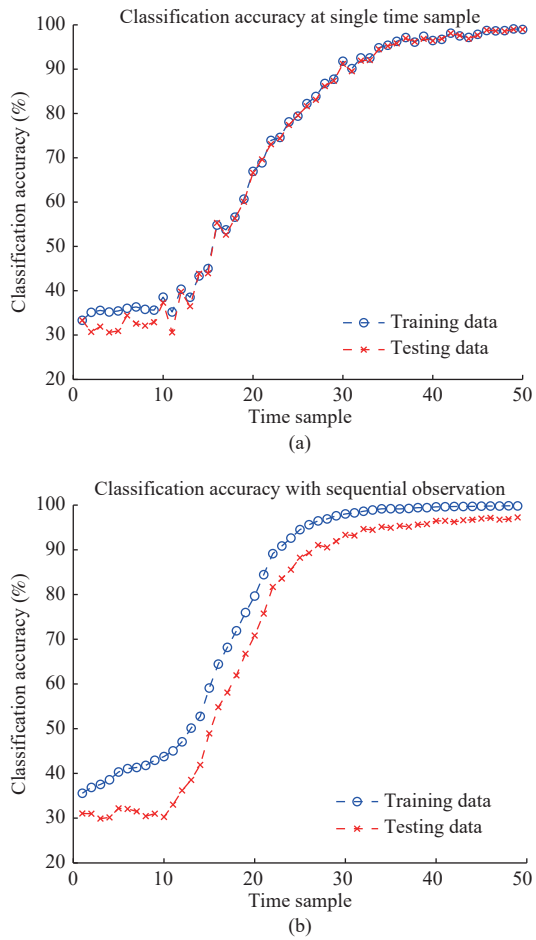


Fig. 2. By observing the number of robots in the center of a simulated 10-bin hallway, a simple neural network could correlate the density with the environment being explored and predict the location of exits with 90% accuracy after 500 robots move just 30 times whether considering (a) a single time step; or (b) a sequential observation.

robot moves before training environments could be classified with 90% accuracy.

As expected, the initial classification accuracy in both approaches of Fig. 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. 2, where the accuracies begin to steadily climb after time step 9. Fig. 2(b) indicates a slight overtraining of the neural network as the training data was consistently classified 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.

C. 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 15 000 in increments of 100. Fig. 3 summarizes the results of this exploration. As anticipated, increasing the size of the environment (i.e., 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.

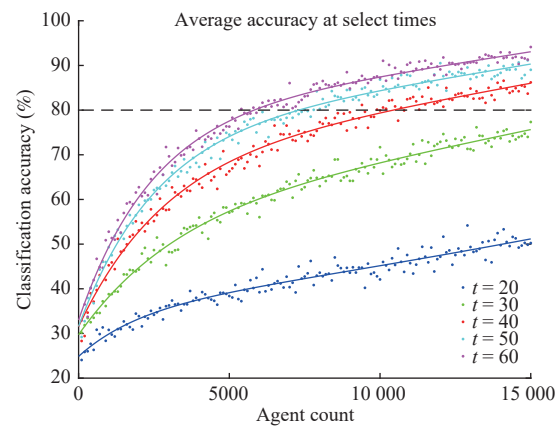


Fig. 3. Increasing the environment size meant more robots were needed to perform the classification and each robot needed more moves to encode features. For a 10×10 environment, it took 10 000 robots and approximately 40 moves before enough interactions had occured for the neural network to accurately identify 80% of the environments.

Fig. 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.

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

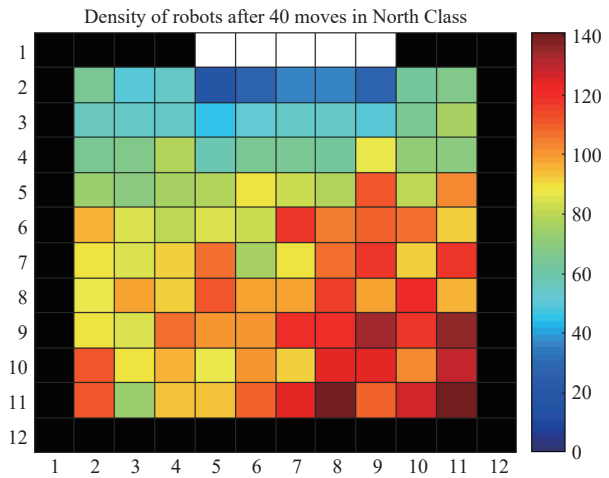


Fig. 4. A heat map of the robot density for 10 000 robots after 40 time steps in a 2D environment with a north doorway shows a lower density in the northern bins as expected. The difference is less distinct in the bins surrounding the observation center at (7,7).

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. 5(a), despite the subtle variation in robot distribution. Using sequential observations around the observation center improved the classification accuracy slightly. Fig. 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. 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 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. 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. 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.

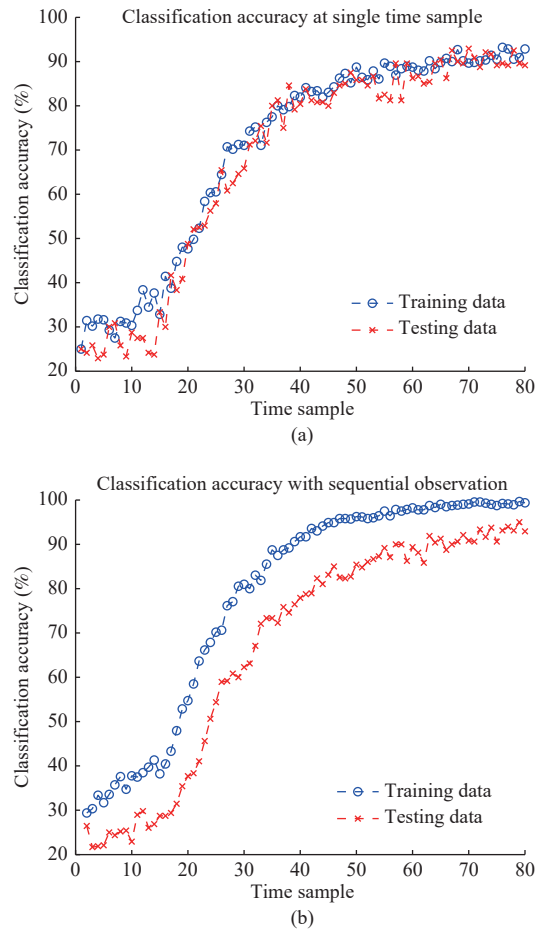


Fig. 5. Locating a doorway in the 2D environment using 8 central bin densities required approximately 40 time steps to reach 80% accuracy if using (a) a single time observation; or using (b) observed robot densities at all previous times.

D. Robustness of the Classification Process

The robustness of the trained network in Fig. 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 I. 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

TABLE I
CLASSIFICATION ACCURACY FOR SHIFTED DOORWAY

Door shift	Accuracy
–3	77%
–2	88%
–1	95%
0	97%
1	97%
2	95%

a wall and only partially overlapped the original doorway position.

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 environment was systematically reduced from 10 000 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. 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.

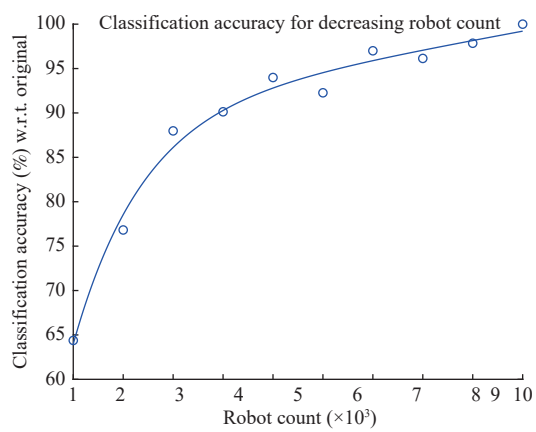


Fig. 6. A network trained on 10 000 robots can still reasonably identify a 2D environment when the system undergoes largescale robot failure.

V. 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 II 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.

TABLE II
CONFUSION MATRIX OF THE TEST DATA CLASSIFICATION FOR THE 2D, SEQUENTIAL OBSERVATION SCENARIO WITH JUST 1000 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%

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 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. 4, the variation in robot density is less clear.

Still referencing Fig. 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 (8). Training data came from the scenario shown in Fig. 5(b) with 10 000 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. 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. 7. At time 40, the dynamic observation produced a classification accuracy of 51% and the improvement continued throughout the simulated time.

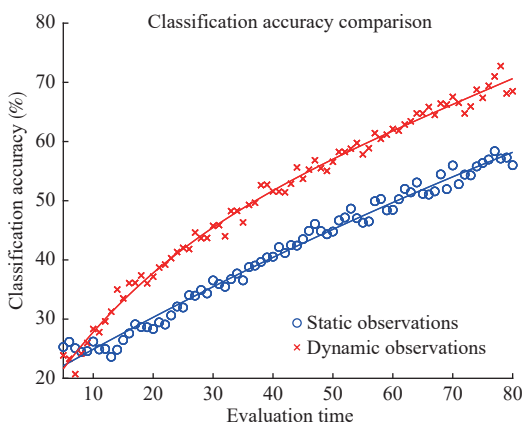


Fig. 7. Moving the observation center one bin opposite the least-likely environment increases the classification accuracy.

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. 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.

VI. 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.

References

[1] I. Navarro and F. Matía, "An introduction to swarm robotics," *Int. Scholarly Research Notices Robotics*, vol. 2013, 2012.

[2] R. L. Sturdivant and E. K. P. Chong, "The necessary and sufficient conditions for emergence in systems applied to symbol emergence in robots," *IEEE Trans. Cognitive and Developmental Systems*, vol. 10, no. 4, pp. 1035–1042, Dec. 2018.

[3] M. Emmons, A. A. Maciejewski, and E. K. P. Chong, "Modelling emergent swarm behavior using continuum limits for environmental mapping," in *Proc. IEEE Int. Conf. Control and Automation*, Jun. 2018, pp. 86–93.

[4] C. A. C. Parker and H. Zhang, "Cooperative decision-making in decentralized multiple-robot systems: the best-of-n problem," *IEEE/ASME Trans. Mechatronics*, vol. 14, no. 2, pp. 240–251, Apr. 2009.

[5] J. T. Ebert, M. Gauci, and R. Nagpal, "Multi feature collective decision making in robot swarms," in *Proc. 17th Int. Conf. Autonomous Agents and Multiagent Systems*, Jul. 2018.

[6] M. Schneider-Fontan and M. J. Mataric, "Territorial multi-robot task division," *IEEE Trans. Robotics and Automation*, vol. 14, no. 5, pp. 815–822, Oct. 1998.

[7] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The Int. J. Robotics Research*,

vol. 23, no. 9, pp. 939–953, 2004.

[8] M. J. Matarić, G. S. Sukhatme, and E. H. Østergaard, "Multi-robot task allocation in uncertain environments," *Autonomous Robots*, vol. 14, no. 2, pp. 255–263, Mar. 2003.

[9] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "A probabilistic approach to collaborative multi-robot localization," *Autonomous Robots*, vol. 8, no. 3, pp. 325–344, Jun. 2000.

[10] A. Prorok, A. Bahr, and A. Martinoli, "Low-cost collaborative localization for large-scale multi-robot systems," in *Proc. IEEE Int. Conf. Robotics and Automation*, May 2012, pp. 4236–4241.

[11] J. Chen, K. H. Low, Y. Yao, and P. Jaillet, "Gaussian process decentralized data fusion and active sensing for spatiotemporal traffic modeling and prediction in mobility-on-demand systems," *IEEE Trans. Automation Science and Engineering*, vol. 12, no. 3, pp. 901–921, Jul. 2015.

[12] A. Giusti, J. Nagi, L. Gambardella, and G. A. D. Caro, "Cooperative sensing and recognition by a swarm of mobile robots," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Oct. 2012, pp. 551–558.

[13] K. H. Low, W. K. Leow, and J. Marcelo H. Ang, "Autonomic mobile sensor network with self-coordinated task allocation and execution," *IEEE Trans. Systems*, *Man*, *and Cybernetics*, *Part C* (*Applications and Reviews*), vol. 36, no. 3, pp. 315–327, May 2006.

[14] S. Bandyopadhyay, S.-J. Chung, and F. Y. Hadaegh, "Probabilistic and distributed control of a large-scale swarm of autonomous agents," *IEEE Trans. Robotics*, vol. 33, no. 5, pp. 1103–1123, Oct. 2017.

[15] A. Dirafzoon and E. Lobaton, "Topological mapping of unknown environments using an unlocalized robotic swarm," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 5545–5551, Nov. 2013.

[16] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and largescale data collection," *The Int. J. Robotics Research*, vol. 34, no. 4–5, pp. 421–436, 2018.

[17] J. Morelli, P. Zhu, B. Doerr, R. Linares, and S. Ferrari, "Integrated mapping and path planning for very large-scale robotic (vlsr) systems," in *Proc. Int. Conf. Robotics and Automation*, May 2019, pp. 3356–3362.

[18] L. Hou, F. Fan, J. Fu, and J. Wang, "Time-varying algorithm for swarm robotics," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 1, pp. 217–222, Jan. 2018.

[19] S. Berman, V. Kumar, and R. Nagpal, "Design of control policies for spatially inhomogeneous robot swarms with application to commercial pollination," in *Proc. IEEE Int. Conf. Robotics and Automation*, May 2011, pp. 378–385.

[20] K. Elamvazhuthi, H. Kuiper, and S. Berman, "PDE-based optimization for stochastic mapping and coverage strategies using robotic ensembles," *Automatica*, vol. 95, pp. 356–367, Sept. 2018.

[21] L. E. Barnes, M. A. Fields, and K. P. Valavanis, "Swarm formation control utilizing elliptical surfaces and limiting functions," *IEEE Trans. Systems*, *Man*, *and Cybernetics*, *Part B*, vol. 39, no. 6, pp. 1434–1445, Dec. 2009.

[22] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli, "The e-puck, a robot designed for education in engineering," in *Proc. 9th Conf. Autonomous Robot Systems and Competitions*, vol. 1, no. 1, pp. 59–65, 2009. [Online]. Available: http://infoscience.epfl.ch/record/135236

[23] F. Mondada, E. Franzi, and P. Ienne, "Mobile robot miniaturisation: a tool for investigation in control algorithms," *Experimental Robotics III*, vol. 200, pp. 501–513, 2009.

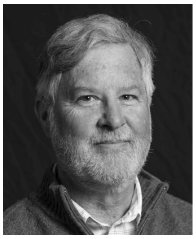
[24] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: a low cost scalable robot system for collective behaviors," in *Proc. IEEE Int. Conf. Robotics and Automation*, May 2012, pp. 3293–3298.

[25] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2013.

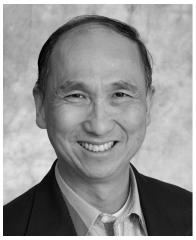
[26] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. Int. Symp. Computational Intelligence in Robotics and Automation*, Jul. 1997, pp. 146–151.

Megan Emmons (S'08) is currently a Ph.D. candidate at Colorado State University, USA. She received the B.S. degree from the Colorado School of Mines, USA, in 2010 and M.S. degree from Utah State University, USA, in 2013. Her research interests include control systems and swarm robotics. She is an elected, at-large Member of the IEEE Robotics and Automation Society. Her research interests include robotics and biologically-inspired systems.

Anthony A. Maciejewski (F'05) received the B.S., M.S., and the Ph.D. degrees in electrical engineering from The Ohio State University, USA, in 1982, 1984, and 1987, respectively. From 1988 to 2001, he was a Professor of electrical and computer engineering with Purdue University, USA. He is currently a Professor and Department Head of Electrical and Computer Engineering with Colorado State University, USA. His research interests include robotics, high-performance computing, and STEM education. A complete vita is available at: http://www.engr.colostate.edu/~aam.



Charles Anderson (SM'15) received the B.S. degree in computer science from the University of Nebraska, USA, in 1978 and the Ph.D. degree in computer science from the University of Massachusetts-Amherst, USA, in 1986. After working as a machine learning Researcher at GTE Labs, he joined the Faculty of the Department of Computer Science at Colorado State University in Fort Collins, USA. He has been a Professor there for 28 years where he holds joint appointment in the Molecular, Cellular, and Integrative Neuroscience Program, the School of Biomedical Engineering, the Graduate Degree Program in Ecology, and the Online Systems Engineering Graduate Program. He teaches and conducts research in machine learning with a focus on deep reinforcement learning. He teaches and conducts research in machine learning algorithms for classification, prediction and control with a focus on reinforcement learning, and brain-computer interfaces.



Edwin K. P. Chong (F'04) received the B.E. degree (First Class Honors) from the University of Adelaide, South Australia, in 1987; and the M.A. and Ph.D. degrees in 1989 and 1991, respectively, both from Princeton, where he held an IBM Fellowship. He joined the School of Electrical and Computer Engineering at Purdue University, USA, in 1991, where he was named a University Faculty Scholar in 1999. Since August 2001, he has been a Professor of electrical and computer engineering and of mathematics at Colorado State University, USA. He coauthored the best-selling book, *An Introduction to Optimization* (4th Edition, Wiley-Interscience, 2013). He is an IEEE Fellow and was President of the IEEE Control Systems Society in 2017. His research interests include optimization, control, estimation, and stochastic systems.