IJRR Submission
000(00):1–13
©The Author(s) 2010
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI:doi number
http://mms.sagepub.com

Adapting to Sensing and Actuation Variations in Multi-Robot Coverage

Alyssa Pierson[1], Lucas C. Figueiredo[2], Luciano C. A. Pimenta[2], and Mac Schwager[1,3]*

[1]*Department of Mechanical Engineering, Boston University, Boston, United States*

[2]*Department of Electronic Engineering, Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, Brazil*

[3]*Division of Systems Engineering, Boston University, Boston, United States*

Abstract

This paper considers the problem of multi-robot coverage control, where a group of robots has to spread out over an environment to provide coverage. We propose a new approach for a group of robots carrying out this collaborative task that will adapt online to performance variations among the robots. Two types of performance variations are considered: variations in sensing performance (e.g., differences in sensor types, calibration, or noise), and variations in actuation (e.g., differences in terrain, vehicle types, or lossy motors). The robots have no prior knowledge of the relative strengths of their performance compared to the others in the team. We present an algorithm that learns the relative performance variations among the robots online, in a distributed fashion, and automatically compensates by assigning the weak robots a smaller portion of the environment and the strong robots a larger portion. Using a Lyapunov-type proof, we show that the robots converge to a locally optimal coverage configuration. The algorithm is also demonstrated in both Matlab simulations and experiments with Pololu m3pi ground robots.

Keywords

Adaptive control, Distributed Robot Systems, Networked Robots

1. Introduction

We are interested in decentralized control strategies for groups of robots that can adapt to individual deficiencies and performance variations within the group. Our work considers a team of robots carrying out a coverage control task, wherein the robots must spread out across the environment while covering areas of high importance. Once deployed, the robots may be given a variety of tasks, such as sensing, surveillance, or servicing events within the environment. The group is heterogeneous in that some robots may perform better than others. The difference in performance can be assessed by various parameters of the robot and will depend on the given task. We assume the robots are unaware of their relative performance with respect to the group, as in the case of a real-world implementation. We propose an algorithm that incorporates online learning and adaptive control into Voronoi-based coverage control. The robots will learn "performance weightings" indicating their performance metric relative to the other robots in the group. This is calculated using the robot's sensing

* e-mail: pierson@bu.edu, lucascoelhof@gmail.com, lucpim@cpdee.ufmg.br, schwager@bu.edu

and actuation errors and local communication with nearby neighbors. If a robot is determined to have low performance weightings, this will shrink the size of the robot's dedicated coverage area, while high performance weightings correspond to a robot taking charge of a larger portion of the environment.

In this paper, we divide coverage tasks into two categories: sensing-based and actuation-based. For sensing-based tasks, the robots move to the centroid of their weighed Voronoi cell, and use sensors to monitor the environment. Here, performance variations can occur due to different sensors on each robot, differences in quality or degradation of the sensors, and sensor creep in long-term deployments. External factors, such as dust or fog, can also affect the quality of each robot's camera and our algorithm adjusts for these variations online. For actuation-based tasks, the Voronoi cells provide regions over which a robot responds to events, requiring the robot to move around its cell as events occur. Both speed and accuracy of the robot are important to the successful completion of their tasks. We combine speed and accuracy into a single metric that we call the performance weighting. Variations in actuation can occur for several reasons. The robots may have different hardware components, or the terrain over which the robots are moving can be varied. For example, if the environment contains a mix of paved roads, forested areas, or dirt paths, the terrain will affect the robots' actuation performance. Overall, internal and external variations lower the efficiency of the entire group. Our algorithm accounts for these variations and adapts the performance weights accordingly.

Applications of sensing-based or actuation-based tasks can be illustrated with several examples. For a sensing-based task, consider a group of robots that has been deployed to take pictures of a region following some disaster, such as an earthquake or building collapse. Here, the robots' Voronoi cells dictate the part of the region for each robot to photograph. Due to varying levels of dust and debris, each robot's picture quality can vary. Using our adaptive weighting algorithm, these lower-performing robots can be compensated for and given a smaller region in the environment. One example of an actuation-based task is a group of agricultural robots that are deployed across some field to water crops in the environment. Once deployed, the robots must move around their cells to water crops as needed. Errors in their speed and precision affect how successful they deliver water, which can affect the overall health of the crops. Another application to consider is the problem of illegal fishing. Some experts estimate this accounts for approximately 1 in 5 fish caught in the wild, which may account for up to $23.5 billion worth of fish in the world market (Trusts, 2015). Here, Voronoi-based coverage control could be used to distribute a robotic network across some area of interest. As boats enter a robots' region, the robots may need to move closer or track patterns of the boats to identify illegal activity. Errors in actuation can compromise how quickly or effectively a robot could track a suspicious boat.

1.1. Related Work

Our work builds upon Voronoi-based coverage control, a common problem in multi-robot systems. A decentralized, multi-robot algorithm was first proposed by Cortés et. al, wherein the robots continuously move towards the centroid of their Voronoi cell (Cortés, 2010; Cortes et al., 2004). This is commonly referred to as the move-to-centroid algorithm. Through this online algorithm, the robots can move through the environment to some locally optimal configuration. The algorithm is decentralized, meaning it scales with the size of the group, and does not require communication with some central agent. Voronoi-based coverage control builds upon previous work in locational optimization, such as the optimal placement of retail facilities, as well as algorithms for data compression (i.e. "vector quantization") (Drezner, 1995). However, a limitation of this original algorithm is that all agents are assumed to have equal characteristics, while in practice a group of agents may have a wide range of performance and sensing capabilities. Other extensions of the Voronoi-based coverage control algorithm use a weighted Voronoi diagram, also called a power diagram. These weightings account for heterogeneity among the robots. Pavone et. al showed that different cell weights allow the agents to take on varying sensing responsibility by modifying the accountable area for each robot (Pavone et al., 2009). Here, a lower relative weight resulted in a smaller cell area for each robot. When the performance or health of the robots is known globally, the weightings can be used to

adjust accountability within the group. To link the cell weightings to performance, Pimenta et. al used the sensing radius of the robot as its weighting (Pimenta et al., 2008). The sensing radius allows the robot to take on a relatively larger cell if it can sense points in its region, and conversely shrink its cell if it has a smaller sensing radius. Another approach used the weightings as an energy-efficiency metric (Kwok & Martinez, 2007). Here, a lower efficiency corresponds to a lower weighting, allowing the high power robots to compensate for the low power robots in the Voronoi tessellation. Marier et. al quantified sensor health with the Voronoi weights, assigning low-performing robots smaller areas of coverage (Marier et al., 2011, 2012). In their implementation, the health was incorporated as a multiplicative factor in a relevant cost function. In Mahboubi et. al's work, the coverage radius of each agent is utilized to minimize the holes within their Voronoi polygons (Mahboubi et al., 2014b). By minimizing the coverage holes individually, the group is able to achieve an improved global coverage. An extension to this work employs multiplicatively-weighted Voronoi cells, which have curved boundaries similar to a circular sensor footprint (Mahboubi et al., 2014a). When the features of the environment are changing, Lee et. al present a strategy that tracks the time-varying information density function within the Voronoi-based coverage configuration (Lee et al., 2015).

Within the existing research utilizing weighted Voronoi cells, all to our knowledge assume the correct weightings are known beforehand. In contrast, our approach learns the performance weightings online using only information about the robot's performance and the data from its neighbors. Preliminary versions of some of the results in this paper appeared in conference versions, which incorporated sensing performance (Pierson & Schwager, 2013) and actuation variation (Pierson & Schwager, 2015) into an adaptive trust weighting. This journal version presents a unified, streamlined theoretical approach to both cases, as well as new simulation results and hardware experiments. This adaptation occurs in parallel to the Voronoi-based coverage control algorithm. We first present an adaptation law in the context of sensing-based tasks, where the robot knows its performance parameters and then adapts the performance weight relative to the performance of the robot's neighbors. Next, we use an actuation-based task to illustrate the case where a robot does not know its performance parameters, and must estimate them online. Using a Lyapunov-style analysis, we show the robots converge to a locally optimal coverage configuration, while the weightings converge to a set of values defined by the performance error. Our algorithm is demonstrated through Matlab simulations and hardware experiments using Pololu m3pi robots.

The rest of the paper is organized as follows: Section 2 outlines the main assumptions, definitions, and problem formulation. Section 3 presents the proof of convergence to a locally optimal configuration when no parameter estimation is needed. Section 4 incorporates a parameter estimator for an actuation-based task and illustrates how the proof of convergence can be written for this added uncertainty. Results of Matlab simulations are presented in Section 5, experimental results are given in Section 6, and conclusions in Section 7.

2. Problem Set-Up

Consider a group of n robots in a bounded, convex environment $Q \subset \Re^2$. Points in Q are denoted q, and positions of individual agents are denoted $p_i \in Q$. Prior coverage control algorithms use the standard Voronoi partition, and for our work, we will use the weighted Voronoi partition. Let $\{V_1, \ldots, V_n\}$ be the Voronoi cells of Q, where each cell is defined as

$$V_i = \{q \in Q \mid \|q - p_i\| \leq \|q - p_j\|, \quad \forall j \neq i\}.$$

The weighted Voronoi partition, also known as the Power Diagram (Aurenhammer, 1987), introduces a weight for each of the cells. Consider w_i as the cell weight for each robot i, and let $\{W_1, \ldots, W_n\}$ be the weighted Voronoi cells in Q, with each cell defined as

$$W_i = \{q \in Q \mid ||q - p_i||^2 - w_i \leq ||q - p_j||^2 - w_j, \quad \forall j \neq i\}. \tag{1}$$

Figure 1 illustrates the differences between a regular and weighted Voronoi diagram. In the figure, the blue lines denote the regular Voronoi diagram, and the green lines are the weighted Voronoi diagram. Here, Robot 2 has been assigned a lower weight relative to its neighbors, and we see that its weighted Voronoi cell is smaller. Similarly, Robot 6 has been assigned a higher relative weight, and its weighted cell is larger. Each robot uses its performance weightings as the Voronoi cell weighting, calculated using information about its individual performance health and the performance of its neighbors. We map these performance parameters to a scalar h_i, which indicates the relative "health" of the robot. In the event that the robot cannot directly measure its performance parameters, we also introduce an online estimator.

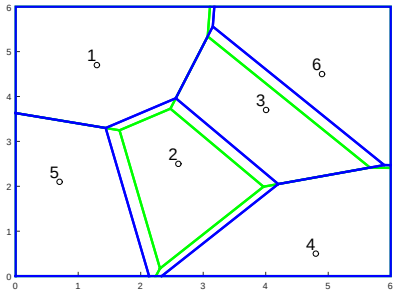


Fig. 1. Here, the regular Voronoi partition is shown in blue, and the weighted partition is shown in green. Robot 2 has a lower weight, which gives it a smaller cell. Similarly, Robot 6 has a higher relative weight, and therefore has a larger cell.

For our region Q, we also define an integrable function $\phi : Q \to \Re_{>0}$ to represent the areas of importance in the environment. Large values of $\phi(q)$ correspond to areas of more importance than small values of $\phi(q)$. All robots are assumed to have knowledge of this function. When the robots do not know this function, techniques have been developed to learn the function online (Schwager et al., 2009; Martínez, 2010).

2.1. Locational Optimization

Before introducing our problem formulation, we state some basic nomenclature and results from Voronoi-based coverage control. A cost function (Cortes et al., 2004; Pavone et al., 2009) for the robot network over the region Q is formulated as

$$\mathcal{H}_V(p_1, ..., p_n) = \sum_{i=1}^{n} \int_{V_i} \frac{1}{2} \|q - p_i\|^2 \phi(q) dq. \tag{2}$$

Intuitively, a low value of $\mathcal{H}_V$ indicates a good coverage configuration of the robots across the environment. For our work, we use the weighted Voronoi cell, also known as the Power Diagram, given in (1). We formulate a similar cost function from the weighted cells that incorporates the robots' performance in the cost. Consider a scalar-valued "health" h_i indicative of the robot's individual performance. We incorporate this into the cost function as

$$\mathcal{H}_W(p_1, ..., p_n) = \sum_{i=1}^{n} \int_{W_i} \frac{1}{2} \left(\|q - p_i\|^2 - h_i \right) \phi(q) dq. \tag{3}$$

We also define M_{W_i} and C_{W_i} of the weighted Voronoi cell as

$$M_{W_i} = \int_{W_i} \phi(q) dq \quad \text{and} \quad C_{W_i} = \frac{1}{M_{W_i}} \int_{W_i} q \phi(q) dq.$$

By definition, $\phi(q)$ is strictly positive, thus M_{W_i} is analogous to the phyiscal mass of the weighted Voronoi cell, and C_{W_i} is analogous to the centroid. While there exists a complex dependency between the position of the robots and the geometry

of the Voronoi cells, a surprising result from locational optimization (Drezner, 1995) is that

$$\frac{\partial \mathcal{H}_W}{\partial p_i} = -\int_{W_i} (q - p_i)\phi(q)dq = -M_{W_i}(C_{W_i} - p_i), \tag{4}$$

which implies the critical points of $\mathcal{H}_W$ will correspond to the robots positioned at the centroids of their weighted Voronoi cells (Pavone et al., 2009; Marier et al., 2011), or $p_i = C_{W_i}$ for all i. Critical points can correspond to either local minima, local maxima, or saddle points. Cortés introduced a gradient-based controller that drives the robots to critical points corresponding to local minima of (2) (Cortes et al., 2004). Using (4) we will introduce a similar controller that only drives the robots towards the local minima of (3). The global optimization of (3) is known to be NP-hard, thus we only consider local minima of H_W. When referring to optimal coverage configurations, we mean locally optimal configurations, which are known to be satisfactory in practice. Variations on the control law that attempt to find the global minima of (2) via exploration are discussed in (Salapaka et al., 2003; Schwager et al., 2008).

2.2. Sensor Model

We introduce the sensing function $\gamma_i(\cdot)$ that relates the sensor health and the data sensed by the robot. Here, $\gamma_i(\cdot)$ is the value measured by the robot's sensor, such as pixel brightness for a camera. The value of $\gamma_i(\cdot)$ is influenced by the position of the robot, the point the robot is sensing, and the health of the sensor. While in practice it is not necessary to know the exact form, for our convergence proofs we assume that $\gamma_i(\cdot)$ is approximated by (Pierson & Schwager, 2013)

$$\gamma_i(p_i, q, h_i) = -\alpha \left(\|q - p_i\|^2 - h_i \right), \tag{5}$$

where h_i is the sensor performance health for robot i and α is a scaling factor. Note this equation for $\gamma_i(p_i, q, h_i)$ shares a similar structure with the weighted Voronoi cell definition (1). We also see that $\gamma_i(p_i, q, h_i)$ can take on different values by different robots looking at the same point q. For example, if γ_i represents the color at point q, variations in camera sensors may produce a different value for a robot i located at p_i versus a robot j located at p_j.

It is not necessary for the robots to know h_i or α directly, so long as $\gamma_i(\cdot)$ can be measured from the robot's sensor. The variables α and h_i shape the approximation of how the sensing quality of some point q decreases as the sensor p_i moves further away from q. Although we use the example of cameras in this paper, $\gamma_i(\cdot)$ can also model other 2D sensors.

2.3. Robot Model

This section describes our models for the dynamics of the robots. Let the robots have integrator dynamics. We can equivalently assume that low-level controllers are in place to cancel existing dynamics and enforce the desired control input. The robots are also given some additive actuation error, denoted by Δ_i and calculated from the robot's performance parameters. The dynamics of the robot can then be written

$$\dot{p}_i = k_p u_i + \Delta_i.$$

Here, u_i is the control input, k_p is a proportional gain, and Δ_i is the actuation error. For this paper, we assume that Δ_i has the form

$$\Delta_i = \begin{bmatrix} \Delta_{i,1} & \Delta_{i,2} \\ \Delta_{i,3} & \Delta_{i,4} \end{bmatrix} u_i, \tag{6}$$

which leads to

$$\dot{p}_i = K_i u_i, \quad K_i = \begin{bmatrix} k_p + \Delta_{i,1} & \Delta_{i,2} \\ \Delta_{i,3} & k_p + \Delta_{i,4} \end{bmatrix}. \tag{7}$$

We assume that K_i remains a positive definite matrix. Practically speaking, this implies that the robot drives within 90^o of its intended direction, as shown in Figure 2. If there is no actuation error, then K_i reduces to the proportional control gain k_p. In practice, we may not know the value of these actuation parameters, in which case we propose an estimator to find $\hat{K}_i$, derived from known quantities and discussed in Section 4.

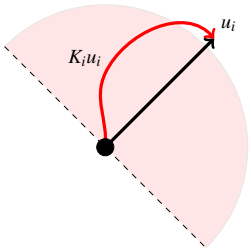


Fig. 2. For some vector u_i, the possible directions of $K_i u_i$ are shown in the shaded region, and one sample path is illustrated in red.

We define the communication network as an undirected graph in which two robots share an edge if they share Voronoi cell boundaries. This is also known as a Delaunay graph. The set of neighbors for any robot i can then be written as $\mathcal{N}_i := \{j | V_i \cap V_j \neq 0\}$. We assume that the robots are able to communicate with their neighbors and share information, such as their position, weight, and sensing data. Additionally, we assume robots are able to compute their weighted Voronoi cell, as defined by (1), which can be computed with well-known algorithms (Cortes et al., 2004; Marier et al., 2011; Salapaka et al., 2003).

Using these models for the sensor and actuation errors, we can now present the problem to be addressed in this work:

Problem 1. ***(Adapting to Sensing and Actuation Variations in Coverage Control)*** *Given a group of robots with positions p_i and performance weights w_i, find the control laws for u_i that locally minimize the coverage cost function given in* (3) *and adaptation laws for $\dot{w}_i$ that compensate for variations in either sensor health modeled in* (5) *or variations in actuation errors modeled in* (6)*.*

3. Adapting to Sensing Variations

In this section we propose a controller and adaptation law that drives the robots to an optimal configuration while adjusting weightings to account for variations in sensing performance. To account for these performance variations, we will define a move-to-centroid control law, and an adaptation law to change the weightings of the robots based on sensor data. We will also assume that the sensor values can be measured directly. We will then prove that the control law drives the robots to converge asymptotically to a stable equilibrium configuration corresponding to a local minimum of the sensing cost function. For this section, we assume there is no actuation error in the robots.

We propose to use the control law

$$\dot{p}_i = u_i = k_p(C_{W_i} - p_i). \tag{8}$$

For the weightings adaptation law, using the sensing function described in (5), we propose

$$\dot{w}_i = \frac{k_w}{2M_{W_i}} \sum_{j \in \mathcal{N}_i} \left(\int_{b_{ij}} [\gamma_i(p_i, q, h_i) - \gamma_j(p_j, q, h_j)]\, dq \right) \tag{9}$$

where k_w is a positive proportional gain constant, and b_{ij} is the shared cell boundary line between neighboring agents i and j. Essentially, this compares the values of the sensing data between two neighbors over shared points along their boundaries, which is illustrated in Figure 3.

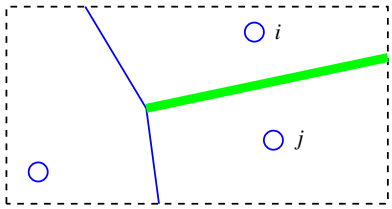


Fig. 3. For neighbors i and j the green line highlights their shared Voronoi cell boundary. The weightings adaptation law compares sensing data along points in this boundary.

To simplify this expression, we notice from (5) the integrand becomes

$$\gamma_i(p_i, q, h_i) - \gamma_j(p_j, q, h_j) = -\alpha \left(\|q - p_i\|^2 - h_i - \|q - p_j\|^2 + h_j \right).$$

However, we are evaluating the point q along the cell boundary, so we know it satisfies (1)

$$\|q - p_i\|^2 - w_i = \|q - p_j\|^2 - w_j.$$

Combining these expressions, we find that for points q along the cell boundary,

$$\gamma_i(p_i, q, h_i) - \gamma(p_j, q, h_j) = -\alpha \left(w_i - w_j - h_i + h_j \right).$$

This yields another form of the weightings adaptation law as

$$\dot{w}_i = \frac{-\alpha k_w}{2M_{W_i}} \sum_{j \in \mathcal{N}_i} \left[(w_i - h_i) - (w_j - h_j) \right] d_{ij}. \tag{10}$$

where d_{ij} is the length of the shared boundary b_{ij}. Note that (9) and (10) are mathematically equivalent, however, the robots can only calculate (9) from sensing data. We will use (10) in the proof of our following theorem.

Theorem 1. *Using the control law* (8) *and the weightings adaptation law* (9)*, the robots converge to the centroids of their weighted Voronoi cells,*

$$\|p_i - C_{W_i}\| \to 0 \ \ \forall \ \ i \in n. \tag{11}$$

Furthermore, the weightings satisfy

$$(w_i - w_j) \to (h_i - h_j) \ \ \forall \ \ i, j. \tag{12}$$

Proof. To prove (11) and (12), we will invoke a global version of LaSalle's Invariance Principle (Bullo et al. (2009), Theorem 1.20). First, we will introduce a continuously differentiable Lyapunov-like function $\mathcal{V}$ that is similar in form to our coverage cost function (3). We use this to show that all trajectories of the system are bounded, and the function is non-increasing, thus $\dot{\mathcal{V}} \leq 0$. Then we will use LaSalle's Principle to prove the claims of the theorem. Consider the function

$$\mathcal{V} = \sum_{i=1}^{n} \int_{W_i} \frac{1}{2} \left(\|q - p_i\|^2 - w_i \right) \phi(q) dq. \tag{13}$$

Its time derivative is

$$\dot{\mathcal{V}} = \sum_{i=1}^{n} \int_{W_i} (q - p_i)^T \phi(q) dq \dot{p}_i - \sum_{i=1}^{n} \int_{W_i} \phi(q) dq \dot{w}_i.$$

The derivative can then be written in two parts as

$$\dot{\mathcal{V}}_1 = \sum_{i=1}^{n} \int_{W_i} (q - p_i)^T \phi(q) dq \dot{p}_i, \quad \dot{\mathcal{V}}_2 = \sum_{i=1}^{n} \frac{-M_{W_i}}{2} \dot{w}_i, \quad \text{where } \dot{\mathcal{V}} = \dot{\mathcal{V}}_1 + \dot{\mathcal{V}}_2 \tag{14}$$

Substituting our controller (8) into $\dot{\mathcal{V}}_1$ yields

$$\begin{aligned} \dot{\mathcal{V}}_1 &= \sum_{i=1}^{n} \int_{W_i} (q - p_i)^T \phi(q) dq \left[k_p (C_{W_i} - p_i) \right] \\ &= \sum_{i=1}^{n} -k_p M_{W_i} \| C_{W_i} - p_i \|^2 \leq 0. \end{aligned}$$

We can also see that plugging in our adaptation law (9) into $\dot{\mathcal{V}}_2$, this simplifies as

$$\begin{aligned} \dot{\mathcal{V}}_2 &= \sum_{i=1}^{n} \frac{-k_w}{4} \sum_{j \in \mathcal{N}_i} \left(\int_{b_{ij}} \left[\gamma_i(p_i, q) - \gamma_j(p_j, q) \right] dq \right) \\ &= 0. \end{aligned}$$

Thus, we have $\dot{\mathcal{V}} \leq 0$.

Given that the derivative $\dot{\mathcal{V}} \leq 0$, we can infer that the trajectories of the robots $p_i(t)$ are bounded. To determine whether the weightings are bounded, consider the vector form of $\dot{w}$, formulated from (10). First, we define

$$w(t) = \begin{bmatrix} w_1(t) \\ \vdots \\ w_n(t) \end{bmatrix}, \quad h = \begin{bmatrix} h_1 \\ \vdots \\ h_n \end{bmatrix}, \quad M^{-1} = \begin{bmatrix} \frac{1}{M_{W_1}} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{1}{M_{W_n}} \end{bmatrix}, \text{ and}$$

$$L = \begin{bmatrix} \ddots & & L_{ij} \\ & \sum_{j \in \mathcal{N}_i} d_{ij} & \\ L_{ij} & & \ddots \end{bmatrix}, \quad L_{ij} = \begin{cases} -d_{ij} & \text{for } j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases}.$$

Here, M^{-1} is a diagonal matrix of positive entries and L is the weighted Laplacian of the neighbor graph, which is known to be positive semi-definite (Godsil & Royle, 2001; Horn & Johnson, 1990), and it can be shown that the product $M^{-1}L$ is positive semi-definite. Hence we can write the derivative in vector form as

$$\dot{w}(t) = -k_w M^{-1} L w(t) + k_w M^{-1} L h. \tag{15}$$

We see that $w(t)$ is the state of a marginally stable filter defined by (15). Note that since the health is static, h is bounded and we have the input to the filter defining $w(t)$ is bounded. From input-to-state properties of marginally stable filters we know that for the weights $w(t)$ to go unbounded the driving signal $k_w M^{-1} L h$ must lie in the null space of the dynamics matrix $-k_w M^{-1} L$ of the filter. However, since the input h is itself multiplied by $k_w M^{-1} L$, the driving signal $k_w M^{-1} L h$ can have no component in the null space of $-k_w M^{-1} L$. Therefore, $w(t)$ remains bounded, and this shows that all trajectories of the system ($p_i(t)$ and $w_i(t)$ for all i) remain bounded.

By the properties of stable linear filters, we know that as the input approaches a limit, the state will also approach a limit (Khalil, 2002), which satisfies the steady-state equation

$$w \to \{w_\infty | 0 = -k_w M^{-1} L(h - w_\infty)\}.$$

Solving for w_∞, we find

$$k_w M^{-1} L w_\infty = k_w M^{-1} L h$$
$$L w_\infty = L h.$$

Since L is the Graph Laplacian, this is equivalent to

$$w_{i,\infty} - w_{j,\infty} = h_i - h_j, \tag{16}$$

proving (12) from Theorem 1.

Given that we have shown that $\dot{\mathcal{V}} \leq 0$, to complete the proof we must find the largest invariant set within the set defined by $\dot{\mathcal{V}} = 0$. We can see that $\dot{\mathcal{V}} = 0$ occurs when $p_i = C_{W_i}$. From our control law (8), this itself is an invariant set. Therefore, by LaSalle's Invariance Principle, we have that

$$p_i(t) \to C_{W_i}(t) \text{ as } t \to \infty,$$

proving (11) from Theorem 1.

□

Remark 1. *This proof shows that using the weightings adaptation law* (9)*, our weightings converge to a set of values relating the sensing performance among agents. Overall, the convergence of the weightings implies they will reach static values, which in conjunction with the move-to-centroid controller* (8) *means the robots will find final locations in the environment. While changing the weightings causes a change in the cell boundaries, thus a change in the centroids, the weightings eventually converge to an invariant set, which means the positions of the robots will eventually reach their centroids.*

Remark 2. *Theorem 1 guarantees convergence of the difference between the weightings to the difference between the corresponding health factors, but it does not guarantee the convergence of each weighting to its corresponding health factor. This is expected, as the robots only have relative sensor measurements to compare. However, weighted Voronoi cell boundaries are calculated by a relative difference* (1)*, so any constant offset is canceled from both sides.*

Remark 3. *The proof structure was chosen to illustrate the parallels with the proof in Section 4 for variations in actuation. The proof can also be written with a simpler structure that only uses a Lyapunov-like function and LaSalle's Invariance Principle, and is presented in the authors' preliminary conference version (Pierson & Schwager, 2013).*

If the weightings are initially assigned the correct values, it implies all robots will agree in the compared sensing data values. In the final result of Theorem 1, the final positions of the robots in the environment are as good as if the correct performance weightings were known beforehand. Using this result, we can show the positional control law (11) and weightings adaptation law (12) provide a solution to Problem 1. Corollary 1 formalizes this by demonstrating that when the weightings converge to a static set of values, our Lyapunov-like function shares the same minima as our coverage cost function.

Corollary 1. *Given the convergence of the performance weightings to the set described by* (12)*, the local minima of our Lyapunov-like function* (13) *are equal to the local minima of our coverage cost function* (3)*.*

Proof. For some constant c, (12) implies $w_{i,\infty} = h_i - c$ for all i. Substituting this into (13), we write

$$\mathcal{V} = \sum_{i=1}^{n} \int_{W_i} \frac{1}{2} \left(\|q - p_i\|^2 - h_i - c \right) \phi(q) dq = H_W + c \int_Q \phi(q) dq.$$

Since $c \int_Q \phi(q) dq$ is constant, the set of positions $(p_1, ..., p_n)$ that comprise a local minima of (13) will also comprise a local minima of (3). □

We can simplify the computational complexity of the weightings adaptation law by comparing sensing values only at a single point, instead of across the entire boundary b_{ij}. The motivation to compare sensing functions at fewer points, as illustrated in Figure 4, is that it may be faster and computationally easier than the boundary calculation, albeit less robust. Corollary 2 shows that a simplification to a single point still maintains the convergence of the weightings to an invariant set, as well as convergence of the location of the robots to their centroids.

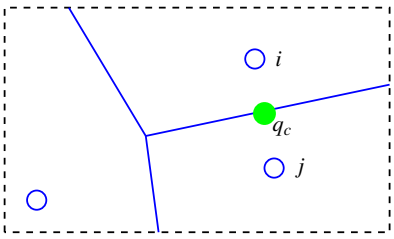


Fig. 4. For neighboring robots i and j, the weighted midpoint q_c (green) lies along the shared Voronoi boundary (blue).

Corollary 2. *The claims of Theorem 1 also hold true for the adaptation law*

$$\dot{w}_i = \frac{k_w}{2M_{W_i}} \sum_{j \in \mathcal{N}_i} \left(\gamma_i(p_i, q_c, h_i) - \gamma_j(p_j, q_c, h_j) \right), \tag{17}$$

where q_c is any point in b_{ij}.

Proof. Using (17) in place of the previous weightings adaptation law (9), and noting that the weighted graph Laplacian becomes the normal graph Laplacian (Godsil & Royle, 2001), the same proof and arguments hold from Theorem 1. □

4. Adapting to Actuation Variations

In our sensing task example, we proposed a control law to drive robots to the centroids of their Voronoi cells, as well as a weightings adaptation law to compare sensor data to compensate for low-performing agents. This section will examine an actuation-based task. In contrast to the previous case, it is necessary to use a parameter estimator in determining the performance health of a robot. We use our estimated parameters in a weightings adaptation law similar to the previous case to adjust the robots' weightings. We then prove that the robots still converge asymptotically to a stable equilibrium configuration corresponding to the local minimum of our cost function (3). We also prove our parameter estimator converges to the true parameter value.

The robots' dynamics are composed of two parts: the desired input u_i and some actuation error Δ_i defined in (6). Our desired input will be the move-to-centroid controller introduced in the last section, written

$$u_i = k_p(C_{W_i} - p_i).$$

Substituting this controller into the dynamics defined in (7), we write

$$\dot{p}_i = u_i + \Delta_i = K_i(C_{W_i} - p_i), \tag{18}$$

where C_{W_i} is the centroid of the robot's weighted Voronoi cell and $K_i > 0$ is the matrix representing control gain and actuation error from (7). In the unlikely case that a Voronoi cell is empty, we evaluate (18) using the integral form of the gradient-descent based controller and let $u_i = \Delta_i = 0$.

We define a function mapping the matrix K_i to a scalar-valued health,

$$h_i = g(K_i),$$

where h_i is the actuation performance "health," and $g(K_i)$ is a function of the properties of the matrix K_i. We require that $g(K_i)$ is bounded when K_i is bounded and continuous, however, the choice of $g(K_i)$ is subjective to the desired performance metrics. Some common choices for $g(K_i)$ include the matrix norm, determinate, trace, or eigenvalues. Here, the robots do not know K_i, so we find an estimate of the matrix, denoted $\hat{K}_i$. This mapping of the actuation parameters to a health can then be used in our weightings adaptation law. By changing the robot's cell weights, we can adjust the size of their Voronoi cell corresponding to their relative performance. Similar to (9), we write the weightings adaptation law as

$$\dot{w}_i = \frac{-k_w}{M_{W_i}} \sum_{j \in \mathcal{N}_i} \Big((w_i - g(\hat{K}_i)) - (w_j - g(\hat{K}_j)) \Big) \tag{19}$$

where k_w is a positive proportional gain constant. In the unlikely case of an empty cell, where M_{W_i} goes to zero, we let $\dot{w}_i = 0$.

4.1. Estimating $\hat{K}_i$

To compute the estimated matrix $\hat{K}_i$, we propose the following online estimator:

$$\begin{aligned} \dot{\hat{K}}_i &= \dot{\lambda}_i - \hat{K}_i \dot{\Lambda}_i \\ \dot{\lambda}_i &= \dot{p}_i (C_{W_i} - p_i)^T \\ \dot{\Lambda}_i &= (C_{W_i} - p_i)(C_{W_i} - p_i)^T. \end{aligned} \tag{20}$$

We further simplify this expression as

$$\begin{aligned} \dot{\hat{K}}_i &= \int_0^t \dot{\lambda}_i(\tau) d\tau - \hat{K}_i \int_0^t \dot{\Lambda}_i(\tau) d\tau \\ &= \int_0^t \dot{p}_i(\tau)(C_{W_i}(\tau) - p_i(\tau))^T d\tau - \hat{K}_i \int_0^t (C_{W_i}(\tau) - p_i(\tau))(C_{W_i}(\tau) - p_i(\tau))^T d\tau \\ &= (K_i - \hat{K}_i) \int_0^t (C_{W_i}(\tau) - p_i(\tau))(C_{W_i}(\tau) - p_i(\tau))^T d\tau \\ &= -\tilde{K}_i \Lambda_i(t) \end{aligned} \tag{21}$$

where

$$\tilde{K}_i = (\hat{K}_i - K_i).$$

Note that although (20) and (21) are mathematically equivalent, the robots can only directly compute (20) because they do not have knowledge of the true error $\tilde{K}_i$. However, the form in (21) is useful for analysis. The behavior of our system compensating for actuation-based variations can be formalized in the following theorem.

Theorem 2. *Using the control law* (18)*, weightings adaptation law* (19)*, and estimator for* $\hat{K}_i$ (20)*, the robots converge to the centroids of their weighted Voronoi cells,*

$$\lim_{t\to\infty} \|p_i(t) - C_{W_i}(t)\| = 0 \ \ \forall \ \ i \in \{1, ..., n\}. \tag{22}$$

Furthermore, the control gain matrix estimation error converges to the null space of $\Lambda_i(t)$*,*

$$\lim_{t\to\infty} \tilde{K}_i(t)\Lambda_i(t) = 0 \ \ \forall \ \ i \in \{1, ..., n\}. \tag{23}$$

Proof. The proof of (22) and (23) will invoke a global version of LaSalle's Invariance Principle (Bullo et al. (2009), Theorem 1.20), similar to the proof of Theorem 1. We will introduce a continuously differentiable Lyapunov-like function $\mathcal{V}$ which will include an additional term to account for the parameter estimation. This will be used to show that all trajectories of the system are bounded, and the function is non-increasing, $\dot{\mathcal{V}} \leq 0$. We then use LaSalle's Principle to prove the claim of the theorem. Consider the function

$$\mathcal{V} = \sum_{i=1}^{n} \int_{W_i} \frac{1}{2} \left(\|q - p_i\|^2 - w_i \right) \phi(q) dq + \sum_{i=1}^{n} \frac{1}{2} Tr[\tilde{K}_i \tilde{K}_i^T]. \tag{24}$$

The time derivative of this function is

$$\dot{\mathcal{V}} = \sum_{i=1}^{n} \int_{W_i} (q - p_i)^T \phi(q) dq \dot{p}_i - \sum_{i=1}^{n} \int_{W_i} \frac{1}{2} \phi(q) dq \dot{w}_i + \sum_{i=1}^{n} Tr[\dot{\hat{K}}_i \tilde{K}_i^T].$$

We can break this into three parts

$$\dot{\mathcal{V}}_1 = \sum_{i=1}^{n} \int_{W_i} (q - p_i)^T \phi(q) dq \dot{p}_i, \quad \dot{\mathcal{V}}_2 = -\sum_{i=1}^{n} \frac{1}{2} M_{W_i} \dot{w}_i, \quad \dot{\mathcal{V}}_3 = \sum_{i=1}^{n} Tr[\dot{\hat{K}}_i \tilde{K}_i^T].$$

Substituting our controller (18) for $\dot{p}_i$, the time derivative $\dot{\mathcal{V}}_1$ becomes

$$\begin{aligned} \dot{\mathcal{V}}_1 &= \sum_{i=1}^{n} \int_{W_i} (q - p_i)^T \phi(q) dq \left[K_i (C_{W_i} - p_i) \right] \\ &= \sum_{i=1}^{n} -M_{W_i} (C_{W_i} - p_i)^T K_i (C_{W_i} - p_i). \end{aligned}$$

Since M_{W_i} is positive and K_i is positive definite, we know $\dot{\mathcal{V}}_1 \leq 0$. For $\dot{\mathcal{V}}_2$, plugging in our adaptation law (19) for $\dot{w}_i$ yields

$$\begin{aligned} \dot{\mathcal{V}}_2 &= \sum_{i=1}^{n} \frac{k_w}{2} \sum_{j \in \mathcal{N}_i} \Big((w_i - g(\hat{K}_i)) - (w_j - g(\hat{K}_j)) \Big) \\ &= 0. \end{aligned}$$

For $\dot{\mathcal{V}}_3$, we plug in our estimator (21) for $\dot{\hat{K}}_i$, thus

$$\begin{aligned}\dot{\mathcal{V}}_3 &= \sum_{i=1}^{n} \text{Tr}\left[-\tilde{K}_i \Lambda_i(t) \tilde{K}_i^T\right] \\ &\leq 0.\end{aligned}$$

Which leads to $\dot{\mathcal{V}} \leq 0$. Since the trajectories of both $p_i(t)$ and $\tilde{K}_i(t)$ are bounded. From this, we see $\hat{K}_i$ is also bounded. To determine whether the weightings are bounded, consider the vector $\dot{w}$. To do so, we define

$$w = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}, \; M^{-1} = \begin{bmatrix} \frac{1}{M_{W_1}} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{1}{M_{W_n}} \end{bmatrix}, \; g_K(t) = \begin{bmatrix} g(\hat{K}_1) \\ \vdots \\ g(\hat{K}_n) \end{bmatrix}, \text{ and}$$

$$L = \begin{bmatrix} \ddots & & L_{ij} \\ & \sum_{j \in \mathcal{N}_i} -1 & \\ L_{ij} & & \ddots \end{bmatrix}, \; L_{ij} = \begin{cases} -1 & \text{for } j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases}$$

Hence we can write the adaptation law in vector form as

$$\dot{w} = -k_w M^{-1} L w + k_w M^{-1} L g_K(t) \tag{25}$$

where M^{-1} is a diagonal matrix of positive entries, L is the Laplacian of Delaunay graph. By using the same arguments given in the proof of Theorem 1 based on the input-to-state properties of marginally stable filters and the fact that $g_K(t)$ is bounded and continuous, we can say that $w(t)$ remains bounded. Thus, all trajectories of the system ($p_i(t)$, $\tilde{K}_i(t)$, and $w_i(t)$ for all i remain bounded.

Since we have already shown that $\dot{\mathcal{V}} \leq 0$, to complete the proof we must find the largest invariant set within the set defined by $\dot{\mathcal{V}} = 0$. We can see that $\dot{\mathcal{V}} = 0$ occurs when $p_i = C_{W_i}$ and $\tilde{K}_i \Lambda_i = 0$. From our control law (18) and estimator (21), this itself is an invariant set. Therefore, by LaSalle's Invariance Principle, we can say that the positions of the robots obey

$$p_i(t) \to C_{W_i}(t) \text{ as } t \to \infty$$

and

$$\tilde{K}_i(t) \Lambda_i(t) \to 0 \text{ as } t \to \infty,$$

proving (22) and (23) from Theorem 2. □

Corollary 3. *If $\Lambda_i(t)$ achieves full rank for all $i \in \{1, ..., n\}$ and any $t > 0$, then*

$$\lim_{t \to \infty} \hat{K}_i(t) = K_i \;\; \forall \, i \in \{1, ..., n\}. \tag{26}$$

Furthermore,

$$\lim_{t \to \infty} (w_i(t) - w_j(t)) = g(K_i) - g(K_j) \tag{27}$$

for all $i, j \in \{1, ..., n\}$.

Proof. From Theorem 2, we see that $\tilde{K}_i(t)$ converges to the null space of $\Lambda_i(t)$. It can be shown that the rank of $\Lambda_i(t)$ is nondecreasing in time. Thus, if at some $\tau > 0$ we find $\Lambda_i(\tau)$ has full rank, then $\Lambda_i(t)$ has full rank for all $t > \tau$, and the

null space of $\Lambda_i(t)$ is the set only containing the zero vector. Therefore for $\tilde{K}_i = (\hat{K}_i - K_i)$, we can write

$$\lim_{t\to\infty} \tilde{K}_i \Lambda_i = 0$$
$$\Rightarrow \lim_{t\to\infty} \hat{K}_i = K_i,$$

proving (26). Furthermore, our weightings adaptation law (19) can be written in vector form, as shown in (25). By the properties of stable linear filters, we know that as the input approaches a limit, the state will also approach a limit (Khalil, 2002), which satisfies the steady-state equation

$$w \to \{w_\infty | 0 = -k_w M^{-1} L(g_{K\infty} - w_\infty)\}$$

where $g_{K\infty}$ is the limit of $g_K(t)$, written as

$$g_{K\infty} = \begin{bmatrix} g(K_i) \\ \vdots \\ g(K_n) \end{bmatrix}.$$

Solving for w_∞ we find

$$k_w M^{-1} L w_\infty = k_w M^{-1} L g_{K\infty}$$
$$L w_\infty = L g_{K\infty}.$$

Given that L is the Graph Laplacian, this is equivalent to

$$w_{i,\infty} - w_{j,\infty} = g(K_i) - g(K_j),$$

proving (27) from Corollary 3. □

Remark 4. *In all of our simulations and experiments, we see that $\Lambda_i(t)$ quickly achieves full rank for all i. To see why, notice that for $\Lambda_i(t)$ not to achieve full rank, the robot i must move in a precisely straight line throughout its entire trajectory, which is unlikely given the nonlinear nature of the system. However, this fact is difficult to prove rigorously.*

Remark 5. *Using arguments similar to Corollary 1, we can show that the local minima of our Lyapunov-like function* (24) *are also local minima of our coverage cost function* (3)*. Using Theorem 2 and Corollary 3, our controller* (18)*, weightings adaptation law* (19)*, and $\hat{K}_i$ estimator* (20) *provide a solution to Problem 1.*

5. Simulations

To demonstrate our move-to-centroid controller (18), weightings adaptation laws (19) and (9), and actuation performance estimator (20), we conducted a series of simulations in Matlab. We present three different simulations: the first simulation is a sensing-based task, where all robots are initialized with equal weights, but one robot has a lower relative health. The second simulation is also for a sensing-based task, but the sensor healths have been randomized. The third simulation shows an actuation-based task, where the initial weights have been randomized and there is one higher-performing robot and one lower-performing robot.

For the actuation-based scenarios, we chose $g(\hat{K}_i) = \|\hat{K}_i\|$ to measure our actuation performance.

5.1. Sensing Example Simulation

We use Matlab to simulate the sensing-based weightings adaptation law within our coverage control algorithm for $n = 8$ robots. A uniform information density function $\phi(q)$ is used. Weightings are initialized to $w_i = 1$ for all i, and the sensing health is $h_i = 1$ for all i, except for robot 2, which has a health of $h_2 = 0.1$. Figure 5 compares the initial and final configurations of the robots.

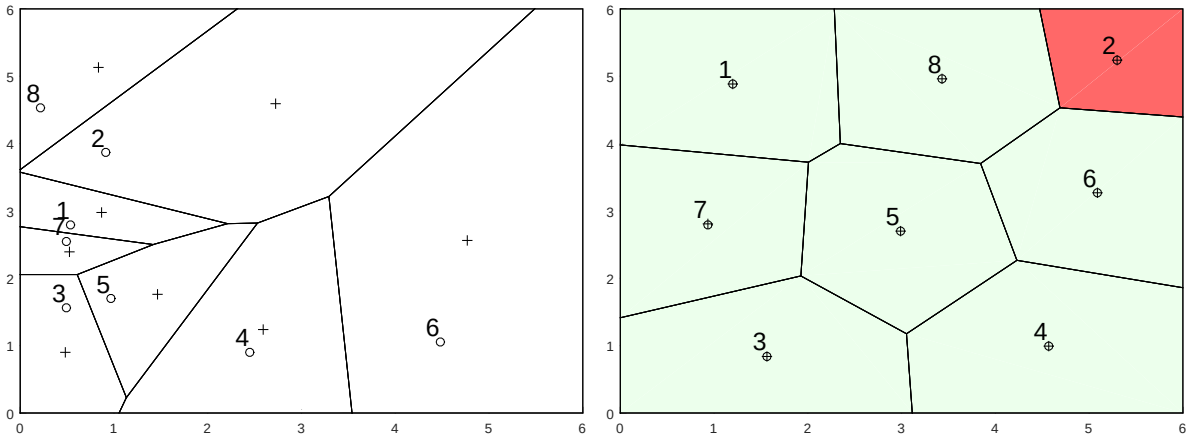


Fig. 5. The (a) initial and (b) final configurations of the robots during a sensing-based task. Here, the cell shading corresponds to the relative performance weight. In the initial configuration, all robots have equal weights, but by the final configuration, robot 2 has the lowest relative weight, as expected.

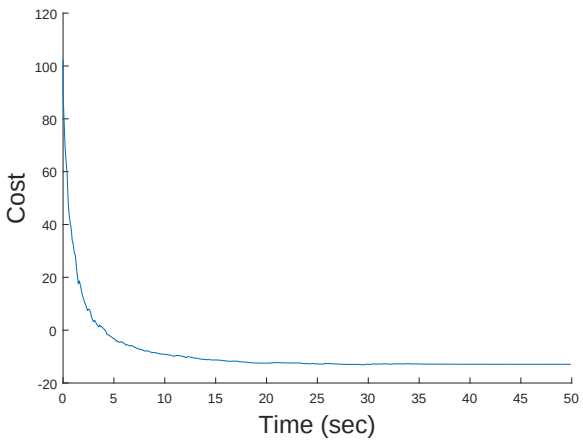
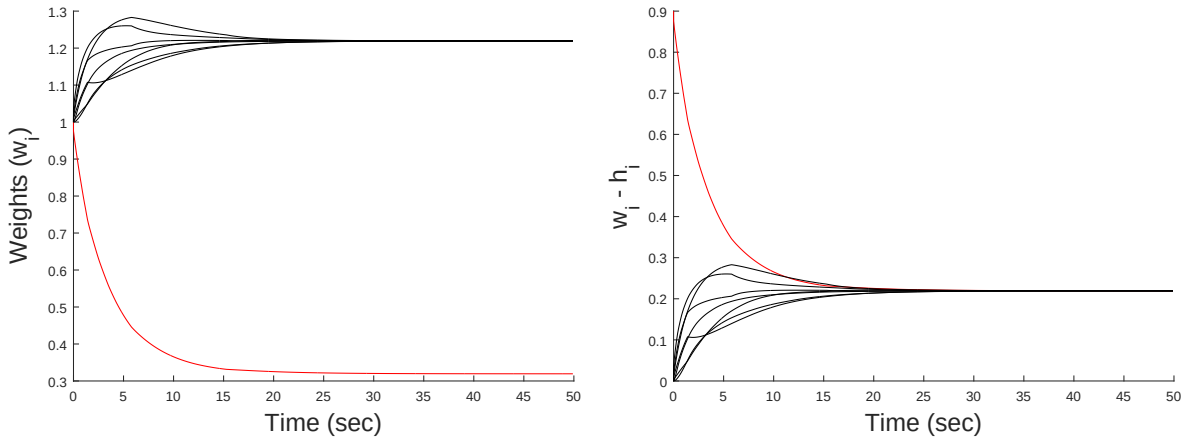


Fig. 6. Sensing cost (3) over time. The cost decreases to a minimum value, indicating the group reached a locally optimal configuration.

Figure 6 shows the cost function (3) over time. We see that the cost decreases until it reaches a minimum value, which corresponds to the group reaching the centroidal Voronoi configuration.

Figure 7 shows the true value of the performance weights over time, and the convergence of the relative difference $(w_i - h_i)$ to a common value, with robot 2 shown in red. We see that robot 2's weight decreases over time, which is expected given its lower sensor health. However, the difference $(w_i - h_i)$ reaches a common value across all agents, as predicted by Theorem 1.

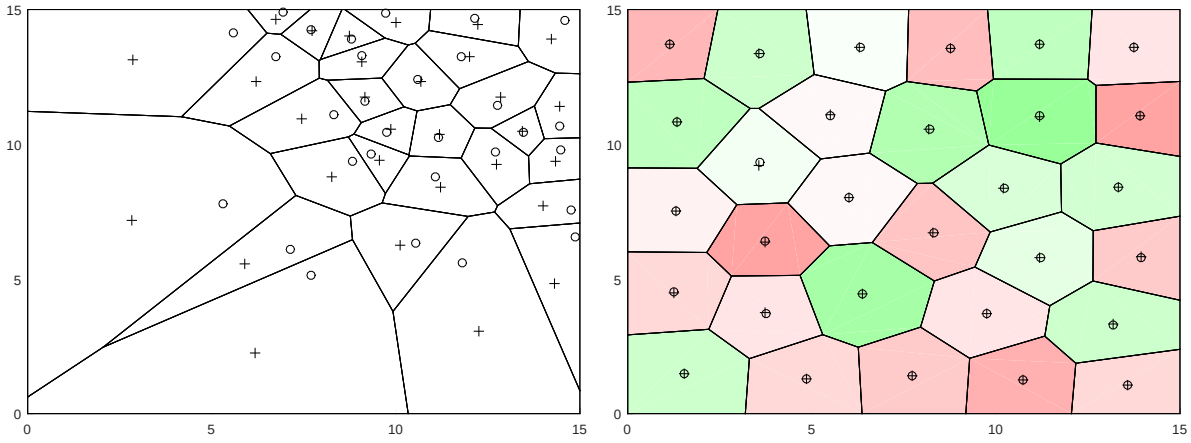


(a) (b)

Fig. 7. (a) Values of the performance weights over time, with robot 2 in red. As predicted, robot 2 attains a lower relative weight than the rest of the group. (b) The difference $(w_i - h_i)$ over time, with robot 2 in red. For the group, this difference converges to a common value as predicted by Theorem 1.

5.2. Sensing Example with Randomized Health

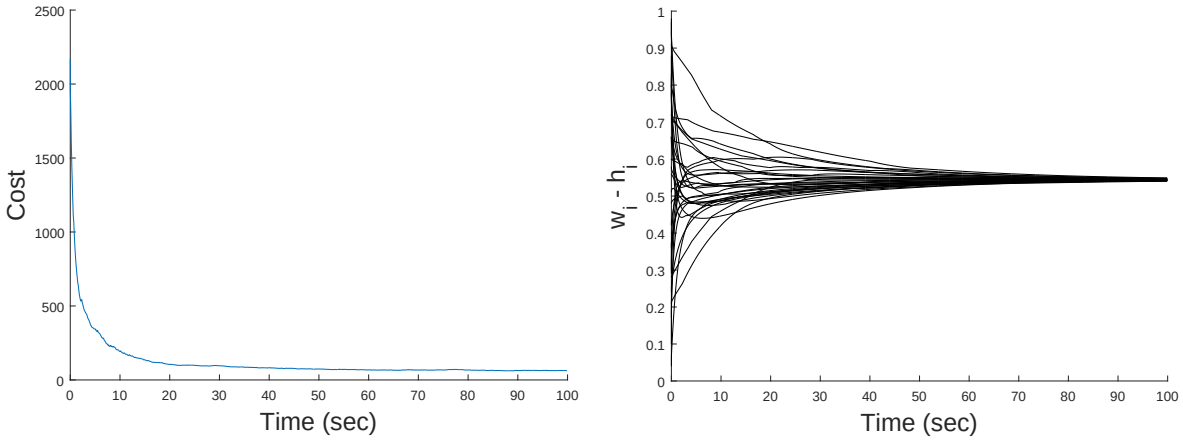
This sensing-based task simulation was also performed in Matlab to demonstrate our controller performance in a randomized configuration. We use a uniform information density function $\phi(q)$ and $n = 30$ agents. In this simulation, the weightings were initialized to be $w_i = 1$ for all i, and sensing health factors were initialized as random numbers drawn from the uniform distribution over $[0, 1]$. Figure 8 compares the initial and final configurations of the robots.



(a) (b)

Fig. 8. The (a) initial and (b) final configurations. The initial configuration begins with all robots at randomized locations, with equal initial weights, shown by the lack of cell shading. By the final configuration, the robots have reached a more balanced configuration, and the weights have adapted to the relative health differences. Lower weights are shown in red, and higher weights are shown in green.

Figure 9 shows the cost function (3) as well as the difference $(w_i - h_i)$ over time. From Figure 8, we see the algorithm reaches a centroidal Voronoi configuration from randomized initial positions. Note that in the initial configuration, all weights are equal, so the shading is the same (white) for all cells. However, by the end of the simulation, the shading reflects the various differences in performance weights to match the variations in health. Robots with the highest relative health have the strongest shade of green, while the lowest relative health is the strongest red. We also observe in Figure 9 that despite the weights diverging to unique values, the weightings converge to the invariant set of $(w_i - h_i) = (w_j - h_j)$.



(a) (b)

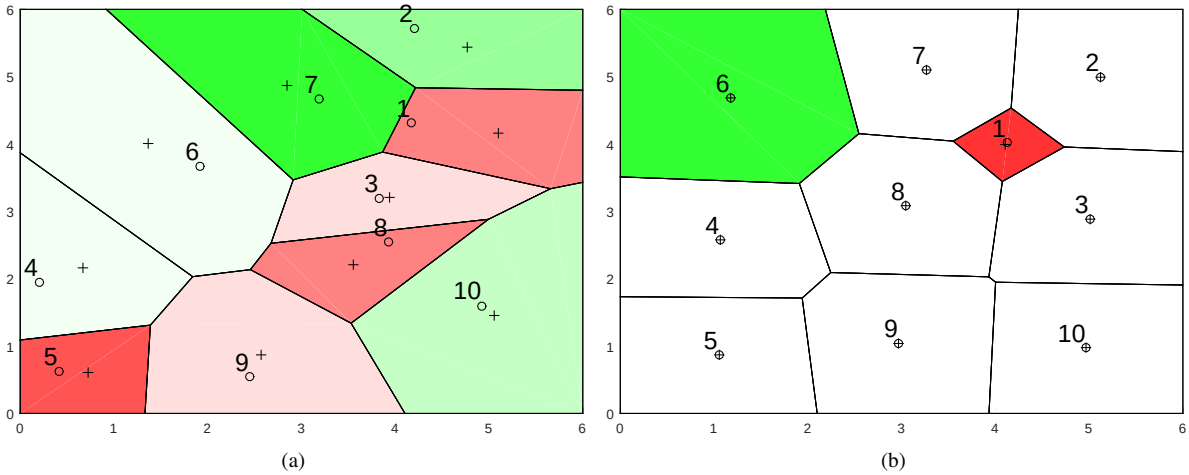
Fig. 9. (a) Cost (3) over time. As the robots spread out over the environment, they minimize the sensing cost before settling into a final, locally optimal configuration. (b) The difference $(w_i - h_i)$ for each robot. As expected, this relative difference converges to a common value across the group.

5.3. Actuation Example Simulation

To demonstrate our adaptation law and parameter estimator, we simulated an actuation example in Matlab. We use a uniform information density function and $n = 10$ agents with randomized initial weighting values drawn from the uniform distribution over $[0, 1]$, as noted below. Robots were assigned $K_i = I$, except robots 1 and 6, whose values are given below.

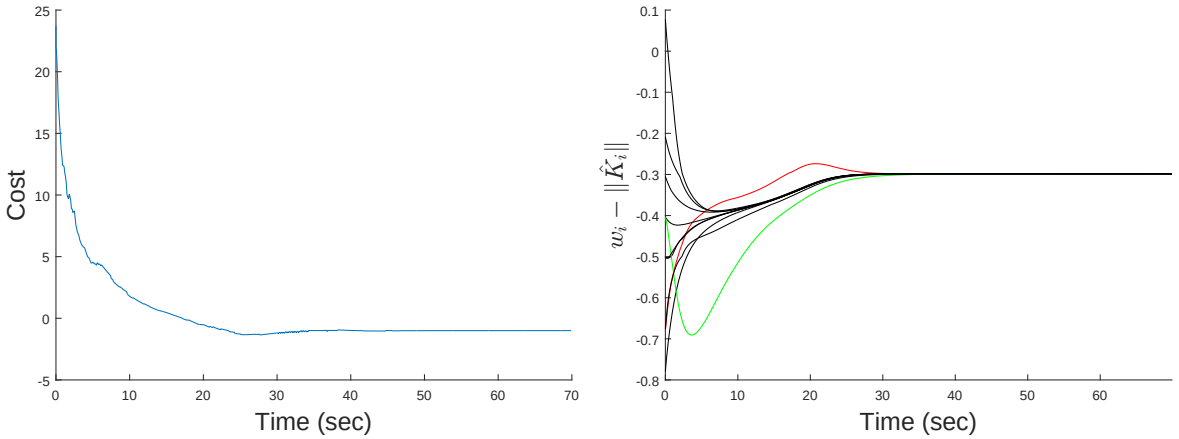
$$w(0) = [0.2,\ 0.7,\ 0.4,\ 0.5,\ 0.1,\ 0.5\ 1.0,\ 0.2,\ 0.4,\ 0.6]$$
$$K_1 = \begin{bmatrix} 0.05 & 0 \\ 0 & 0.05 \end{bmatrix}, \quad K_6 = \begin{bmatrix} 1.65 & 0 \\ 0 & 1.65 \end{bmatrix}.$$

Figure 10 shows the initial and final configurations of the agents, with their relative performance weights given by the cell shading. Initially, the robots are assigned random weights, but by the end of the simulation, robot 1 has the lowest weight, and robot 6 has the highest weight, as expected. Figure 11 shows the global cost and convergence of $w_i - \|\hat{K}_i\|$.



(a) (b)

Fig. 10. The initial (a) and final (b) configurations, with the cell shading corresponding to the relative performance weight. The initial configuration has the robots starting at randomized locations with varying initial performance weights. By the final configuration, the highest performing robot 6 has the largest cell, and the lowest performing robot 1 has the smallest cell, with its relative weight indicated by the shading.



(a) (b)

Fig. 11. (a) Cost (3) over time. We see the cost decreases to a minimum, demonstrating the group reaches a locally optimal configuration. (b) The relative difference $(w_i - \|\hat{K}_i\|)$, with Robot 1 shown in red, and Robot 6 in green. As predicted, the group reaches a common value.

We see that the cost over time decreases to a minimum, corresponding to the the group finding a locally optimal centroidal configuration. As expected, we see that the values of $w_i - \|\hat{K}_i\|$ are equal across all agents. Figure 12 shows the values of our $\hat{K}_i$ estimator over time. Although robots 1 and 6 do not initially know their K_i matrix, using our online

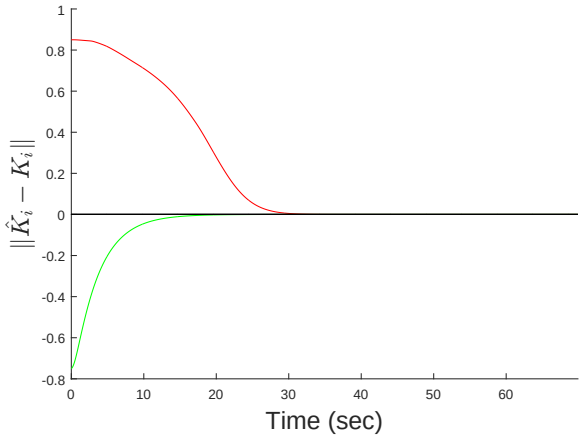


Fig. 12. Convergence of $\|\hat{K}_i\|$ to $\|K_i\|$, with Robot 1 shown in red, and Robot 6 in green. Initially, the estimate for these robots is incorrect, but over time, the robots learn their true value of K_i as they move throughout the environment.

estimator they are able to successfully determine the correct values. Since their estimate of performance converges, the conditions of Corollary 3 are satisfied, and we know that the performance weightings will converge to $(w_i - \|K_i\|)$.

6. Experiments

In order to verify the behavior of our controller, we implemented our algorithm using m3pi[1] robots equipped with XBee[2] radios. Pose data was calculated using an Optitrack[3] system. The experiments were run to parallel the scenarios presented

[1] Pololu's m3pi: www.pololu.com/product/2151
[2] Digi's XBee: www.digi.com/xbee/
[3] Natural Point OptiTrack: www.naturalpoint.com/optitrack/

in our Simulation section. Videos of the experiments are included with this paper and can also be found on the Multi-robot Systems Lab website [4]. For our weightings adaptation law, all experiments use the performance function $g(\hat{K}_i) = \|\hat{K}_i\|$.

The m3pi robot is a small differential drive robot from Pololu Robotics, shown in Figure 13. It utilizes an onboard mbed microcontroller to handle actuation and communication. The mbed controls the motors on the robot, and we send velocity data via XBee radios to the robots based on the Voronoi calculations in Matlab. To localize our robots, we used NaturalPoint's OptiTrack system with sixteen IR cameras. Short-throw projectors were used to display the centroid and Voronoi boundaries on the floor mats during the experiments, also shown in Figure 13. Given the m3pi robots are non-

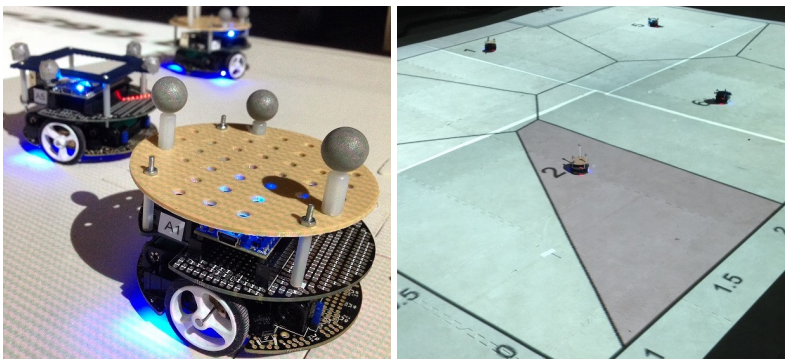


Fig. 13. The m3pi robots (left) used in the experiments. Each robot is equipped with an mbed processor to handle the low-level control. Communication with Matlab is done via an XBee radio, and the silver reflective markers are used for identification in our Optitrack system. The Voronoi cells are projected onto the floor during each experiment (right).

holonomic vehicles, we also incorporated a low-level point-offset controller (Michael & Kumar, 2009) to account for the dynamics. Here, instead of driving the robot to the centroid, we will drive the point-offset of the robot to the centroid. In the experiment video stills, the point-offset is plotted with a circle, and the centroid of the Voronoi cells are plotted with a '+' symbol.

6.1. Sensing Example

This experiment demonstrates the performance of the controllers presented in Section 3. Here, our environment has a uniform information density function $\phi(q)$. Seven robots were initialized with different weight and health values, assigned as

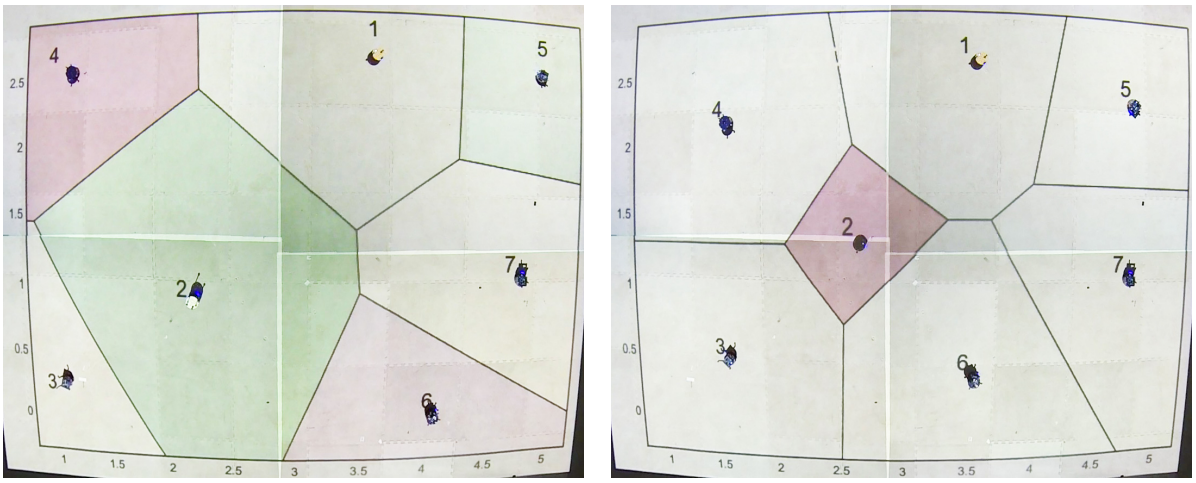
$$h = [1.0 \;\; 0.3 \;\; 1.0 \;\; 1.0 \;\; 1.0 \;\; 1.0 \;\; 1.0],$$
$$w(0) = [0.6 \;\; 0.8 \;\; 0.4 \;\; 0.1 \;\; 0.7 \;\; 0.3 \;\; 0.5].$$

Figure 14 shows the initial and final configurations of the agents over the course of the experiment. Each cell is shaded to indicate its relative weight, with green indicating a higher relative weight, and red indicating a lower relative weight.

To assess the performance, we can also examine a plot of the cost and $(w_i - h_i)$ given in Figure 15.

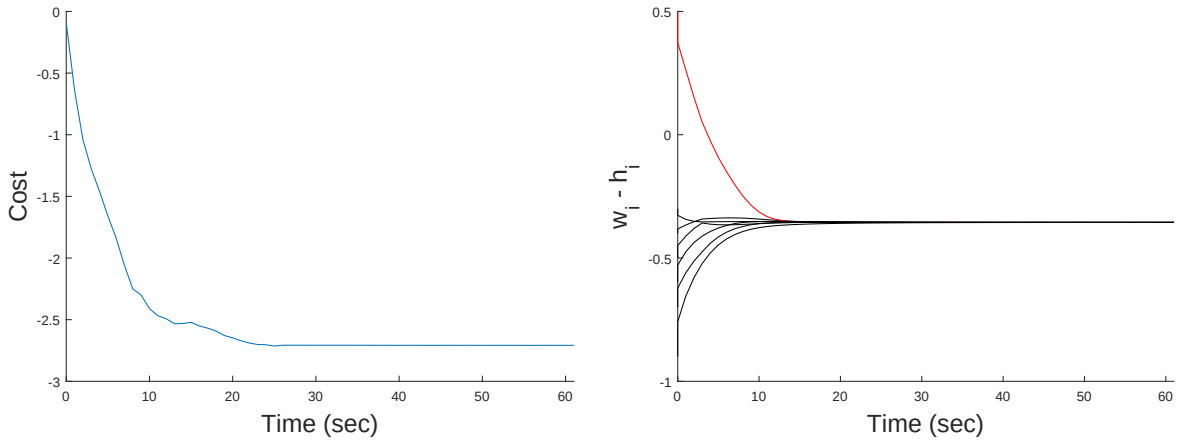
Similar to the simulations, we see the cost decreasing to a minimum value, indicating the robots reach a locally optimal configuration. We also see that over time, the difference $(w_i - h_i)$ converges to the same value across all robots, as predicted by Theorem 1.

[4] Experimental videos: http://sites.bu.edu/msl/research/adaptive-trust-coverage/



(a) (b)

Fig. 14. In the initial configuration (a), the robots have been assigned random initial weights. We see Robot 2 has the largest performance weight, despite it actually having the lowest health. In the final configuration (b), we see that the group has compensated, and now Robot 2 has the lowest relative weight, indicated by the red shading.



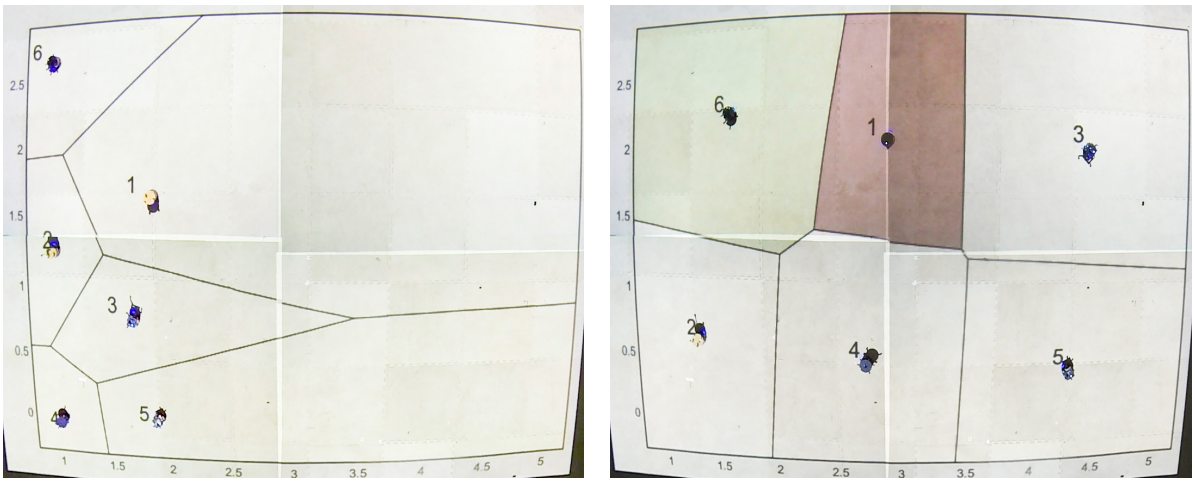
(a) (b)

Fig. 15. (a) Cost (3) over time. We see the cost decreases to a minimum value, indicating the group achieves a locally optimal final configuration. (b) The value of $(w_i - h_i)$, with Robot 2 indicated in red. Over time, the group reaches a consensus on the difference between the weights and the health.

6.2. Actuation Example

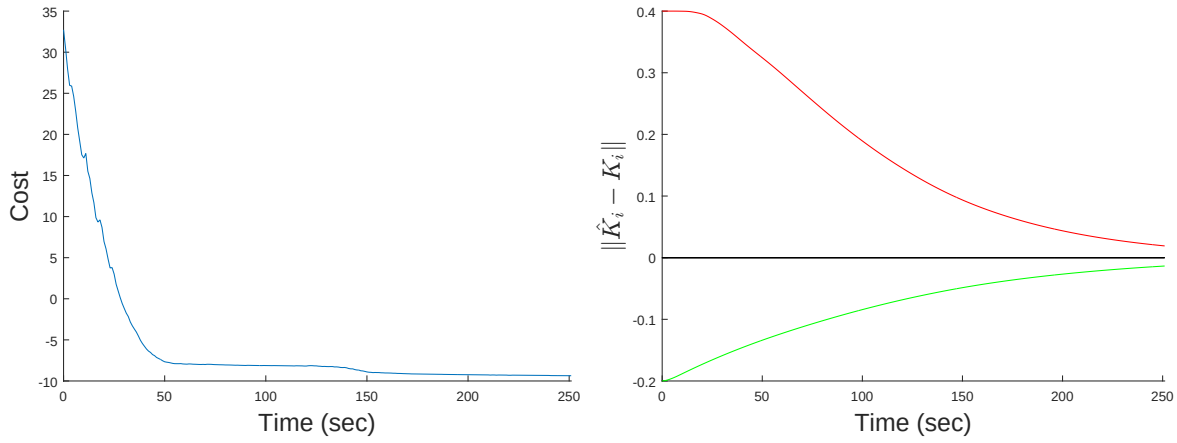
In this experiment, we used six robots in a constant information density function $\phi(q)$ environment. All robots were initial weights of $w_i = 1$. Robots were given $K_i = I$ matrix, except robot 1, which was assigned $K_1 = [0.6, 0; 0, 0.6]$, and robot 6, which was assigned $K_6 = [1.2, 0; 0, 1.2]$. Figure 16 shows the initial and final configurations of the agents.

In Figure 17(a) we can see that the cost function decreases over time, which means that the group converged to a locally optimal configuration. We can also see the successful convergence of our estimator $\hat{K}_i$ to K_i in Figure 17(b). A plot of the weightings over time in Figure 18(a) shows that the lowest performing agent 1, shown in red, has the lowest weighting over time. Figure 18(b) shows that the difference $(w_i - \|\hat{K}_i\|)$ converges to a common value.



(a) (b)

Fig. 16. In the intial configuration (a), note that all robots start off with equal weights. Robot 1 has the lowest health, but a relatively large cell, while Robot 6 has a high health, but small cell. In the final confiuration (b), the shading indicates the relative weights. Note that Robot 1's cell is now much smaller, and the red indicates its lower relative weight. Similarly, Robot 6's cell has increased in size, and the green indicates a realtively higher performance weight.

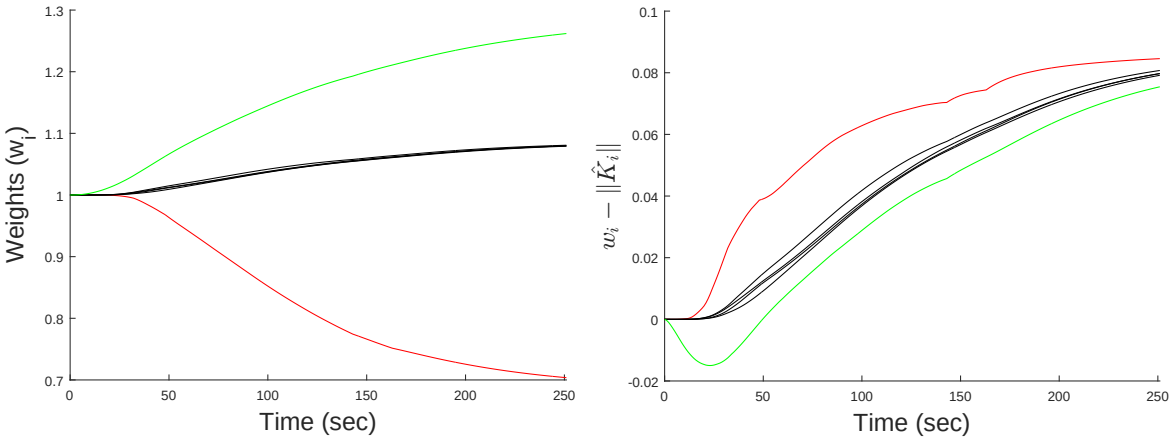


(a) (b)

Fig. 17. (a) Cost (3) over the course of the experiment. We see the cost decreases to a minimum value, indicating the group reaches a static, locally optimal configuration. (b) The $\hat{K}_i$ estimator versus the true value K_i over time. For agents 6 (green) and 1 (red), we see the estimator successfully converges to the true value over the experiment.

7. Conclusion

This paper presents a method of using adaptive weightings to adjust for individual variations in performance within multi-robot coverage control. We consider both errors due to sensing variations, and errors due to variation in actuation abilities. To account for these errors, the robots compare values of an error estimate with their neighbors, and using an adaptive weighting law, adjust the value of their weightings online. By controlling these weights, we are able to modify the Voronoi boundaries between neighboring robots, which adjusts a robot's cell size relative to its neighbors. The weightings adaptation law and error estimation occur online within the coverage control algorithm. We demonstrate the algorithm in both simulation and experiments using m3pi robots.



(a) (b)

Fig. 18. (a) True value of the weights over time. Note agent 6 (green) has a higher weight, corresponding to its better performance, and agent 1 (red) has a lower health, corresponding to its weaker performance. (b) The difference $(w_i - \|\hat{K}_i\|)$ over time, which converges to a common value across all agents.

Our method incorporates performance error into the decentralized algorithm while maintaining stability and performance. This can provide an additional level of robustness in real-world applications when the robots are in an unknown environment and may have varying capabilities across the team. It can also provide insight into identifying failures of a team member. In this work, we only consider robots with variations in performance, but they are not malicious or manipulative. In future work we plan to explore models of malicious agents and their impact on our algorithm.

Acknowledgments

The authors would like to thank Rebecca Wong for her help with the programming and implementation of the algorithms on the m3pi robots. This work was supported in part by ONR grant N00014-12-1000, NSF grant CNS-1330036, the Clare Boothe Luce Fellowship, and by the Brazilian Agencies: CAPES, CNPq, FAPEMIG, and FINEP. We are grateful for this support.

References

F. Aurenhammer (1987). 'Power Diagrams: Properties, Algorithms and Applications'. *SIAM J. Comput.* **16**(1):78–96.

F. Bullo, et al. (2009). *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press.

J. Cortés (2010). 'Coverage optimization and spatial load balancing by robotic sensor networks'. *Automatic Control, IEEE Transactions on* **55**(3):749–754.

J. Cortes, et al. (2004). 'Coverage control for mobile sensing networks'. *Robotics and Automation, IEEE Transactions on* **20**(2):243–255.

Z. Drezner (1995). *Facility location: a survey of applications and methods*. Springer series in operations research. Springer.

C. Godsil & G. Royle (2001). *Algebraic Graph Theory*. Graduate Texts in Mathematics. Springer-Verlag.

R. A. Horn & C. R. Johnson (1990). *Matrix analysis*. Cambridge university press.

H. Khalil (2002). *Nonlinear Systems*. Prentice Hall PTR.

A. Kwok & S. Martinez (2007). 'Energy-balancing cooperative strategies for sensor deployment'. In *Decision and Control, 2007 46th IEEE Conference on*, pp. 6136–6141. IEEE.

S. Lee, et al. (2015). 'Multirobot Control Using Time-Varying Density Functions'. *Robotics, IEEE Transactions on* **31**(2):489–493.

H. Mahboubi, et al. (2014a). 'Distributed Deployment Algorithms for Efficient Coverage in a Network of Mobile Sensors With Nonidentical Sensing Capabilities'. *Vehicular Technology, IEEE Transactions on* **63**(8):3998–4016.

H. Mahboubi, et al. (2014b). 'Distributed Deployment Algorithms for Improved Coverage in a Network of Wireless Mobile Sensors'. *Industrial Informatics, IEEE Transactions on* **10**(1):163–174.

J.-S. Marier, et al. (2011). 'Optimizing the location of sensors subject to health degradation'. In *American Control Conference (ACC), 2011*, pp. 3760–3765. IEEE.

J.-S. Marier, et al. (2012). 'Health-Aware Coverage Control With Application to a Team of Small UAVs'. *IEEE Transactions on Control Systems Technology* **21**.

S. Martínez (2010). 'Distributed interpolation schemes for field estimation by mobile sensor networks'. *IEEE Transactions on Control Systems Technology* **18**(2):419–500.

N. Michael & V. Kumar (2009). 'Planning and Control of Ensembles of Robots with Non-holonomic Constraints'. *Int. J. Rob. Res.* **28**(8):962–975.

M. Pavone, et al. (2009). 'Equitable partitioning policies for robotic networks'. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pp. 2356–2361. IEEE.

A. Pierson & M. Schwager (2013). 'Adaptive Inter-Robot Trust for Robust Multi-Robot Sensor Coverage'. In *International Symposium on Robotics Research*.

A. Pierson & M. Schwager (2015). 'Adapting to Performance Variations in Multi-Robot Coverage'. In *International Conference on Robotics and Automation*. IEEE.

L. Pimenta, et al. (2008). 'Sensing and coverage for a network of heterogeneous robots'. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pp. 3947–3952. IEEE.

S. Salapaka, et al. (2003). 'Constraints on locational optimization problems'. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol. 2, pp. 1741–1746. IEEE.

M. Schwager, et al. (2008). 'A ladybug exploration strategy for distributed adaptive coverage control'. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 2346–2353. IEEE.

M. Schwager, et al. (2009). 'Decentralized, adaptive coverage control for networked robots'. *The International Journal of Robotics Research* **28**(3):357–375.

P. C. Trusts (2015). 'Pew Unveils Pioneering Technology to Help End Illegal Fishing'. http://www.pewtrusts.org/en/about/news-room/press-releases/2015/01/21/pew-unveils-pioneering-technology-to-help-end-illegal-fishing.