Adapting to Performance Variations in Multi-Robot Coverage

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

Abstract— **This paper proposes a new approach for a group of robots carrying out a collaborative task to adapt on-line to actuation performance variations among the robots. We consider the problem of multi-robot coverage, where a group of robots has to spread out over an environment to cover the environment with their sensors. We suppose that some robots have poor actuation performance (e.g. weak motors, friction losses in the gear train, wheel slip, etc.) and some have strong actuation performance (powerful motors, little friction, favorable terrain, etc.). The robots do not know before hand the relative strengths of their actuation compared to the others in the team. The algorithm in this paper learns the relative actuation performance variations among the robots on-line, in a distributed fashion, and automatically compensates by giving the weak robots a small portion of the environment for sensing, and giving the strong robots a larger portion. Using a Lyapunov-type proof, it is proven that the robots converge to locally optimal positions for coverage. The algorithm is demonstrated in both Matlab simulations and experiments using seven Pololu m3pi mobile robots.**

I. Introduction

For multi-robot systems to perform robustly and practically in real-world settings, it is necessary for the robots to adapt to individual deficiencies and performance variations among the team. Our work considers a team of robots carrying out a coverage control task, in which the robots must spread out over the environment to cover it with their sensors. The team is heterogeneous in that some robots have better actuation performance than others. The difference may be due to different hardware components in the robots, differences in aging and degradation over time, or differences in the underlying terrain on which the robots are moving. In any case, we assume that some robots move faster and more accurately towards their goal than others. Furthermore, the robots do not know their relative performance with respect to the group, as would be the case in a real world setting. We propose an algorithm that incorporates online learning and Voronoi based coverage control. Using this algorithm, the robots learn a "performance weighting" indicating their own performance relative to the others in the group, using only local communication, and knowledge of their own actuation errors measured online. The effect of a low "performance weighting" is to shrink the size of the robot's dedicated coverage area, while robots with a high "performance weighting" take charge of covering a larger portion of the environment.

This ability to adapt online to actuation performance variations is important in several examples. First, consider a situation with a group of robots working in a warehouse. In this scenario, robots are assigned regions of the warehouse and must retrieve items within their region as necessary. Errors in actuation could limit the robot's speed and precision and impair efficiency of the whole operation. Our algorithm accounts for this actuation quality and adapts the performance weights accordingly so that items are retrieved efficiently despite robot performance variations. Consider another situation in which robots are covering an environment with highly variable terrain (e.g. paved in some places, forested in other, sandy in others). In this case the wear and tear on the vehicles as well as the terrain itself will affect actuation across the terrain. These variations can be accounted for with our algorithm, so that the team maintains efficient coverage despite the heterogeneous terrain. Indeed, many coverage scenarios with real world performance variations can benefit from our algorithm.

Our algorithm builds upon the Voronoi based coverage strategy first proposed by Cortés et al. [1], [2]. In this algorithm, often referred to as the move-to-centroid controller, all robots continuously drive toward the centroids of their Voronoi cells. This builds upon previous work in optimally locating retail facilities [3], as well as applications in data compression (i.e. "vector quantization") [3]. Other extensions of Voronoi coverage control have used the weighted Voronoi diagram, also called the power diagram, where the weightings account for heterogeneity among robots. Using the Power Diagram, Pavone et. al showed that different cell weights allowed for agents to take on varying sensing responsibility [4]. Within a heterogeneous group of robots, the weighting can be used to correspond to the sensing radius of the robot [5]. Another approach used weighted Voronoi cells as an energy-efficiency metric, allowing the group to compensate for low-energy robots [6]. By using the Voronoi weights to quantify sensor health, Marier et. al were able to assign low-performing robots smaller areas of coverage ([7], [8]).

Within this variety of existing research on weighted Voronoi cells, most assume the correct weightings are known *a priori*. In contrast, our work seeks to *learn* performance weightings *online* using only information about the robot's actuation, and data from its neighbors. This paper builds on the authors' previous work, which incorporated sensing performance into an adaptive trust weighting [9]. Here, instead we consider actuation error instead of sensing errors.

[1]Alyssa Pierson and Mac Schwager are with the Department of Mechanical Engineering, Boston University, Boston, MA 02215 pierson@bu.edu, schwager@bu.edu

[2]Lucas Coelho Figueiredo and Luciano Pimenta are with the Department of Electronic Engineering, Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, MG 31270-901 Brazil lucascoelhof@gmail.com, lucpim@cpdee.ufmg.br

Actuator performance is significantly more challenging to learn than sensing performance because it requires learning a gain matrix using all past history of actuation errors, leading to a two part adaptation law for the performance weights. In contrast, our previous work learns a scalar value (indicating sensor performance) with a one part adaptation law. Specifically, in this work we derive a two part adaptation law for each robot to (i) learn an unknown control gain matrix that quantifies actuation errors, and (ii) use this to adapt a performance weight to penalize or reward the robots based on its learned gain matrix. This adaptation occurs simultaneously while the robots carry out a Voronoi based coverage control algorithm. We prove with a Lyapunov style proof that the robots converge to a locally optimal coverage configuration, while the weightings converge to a set of values defined by the actuation error. We demonstrate the performance of the algorithm in Matlab simulations, and in hardware experiments with 7 Pololu m3pi robots.

II. Problem Set-Up

Consider a set of n robots in a bounded, convex environment $Q \subset \Re^2$. Points in Q are denoted q, and the position of the i-th agent is $p_i \in Q$. Prior coverage control algorithms use the standard Voronoi partition of the environment. Let $\{V_1, \dots, V_n\}$ be the Voronoi partition of Q, with each cell defined as

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

For our work, we use the weighted Voronoi partition, also known as the Power Diagram, with each weighting w_i serving as the performance weighting for robot i. Let $\{W_1, \dots, W_n\}$ be the weighted Voronoi partition of Q, with each cell defined as

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

For our bounded region Q, we also define an integrable function $\phi : Q \to \Re_{>0}$ to represent the areas of importance in the environment. Areas with large values of $\phi(q)$ are more important than those with small values, and all the robots have knowledge of this function. When the robots do not know this function, techniques have been developed to learn it online from sensor data [10], [11].

A. Locational Optimization

Before introducing our problem formulation, we will state some basic nomenclature and results from Voronoi based coverage control. A complete discussion can be found in [2], [4]. We can formulate a cost function for the sensing network over some area Q as

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

Intuitively, a low value of $\mathcal{H}$ indicates a good configuration of the robots for coverage of the environment. Our formulation uses the weighted Voronoi cell, also known as the Power Cell, given in (1). We can formulate a similar cost function from the weighted cells as

$$\mathcal{W}(p_1, ..., p_n, w_1, ..., w_n) = \sum_{i=1}^{n} \int_{W_i} \frac{1}{2} \left(\|q - p_i\|^2 - w_i\right) \phi(q) dq, \tag{3}$$

where W_i is the robot's weighted Voronoi cell, and w_i is the robot's individual performance weighting. From this cost function, we can also define the quantities M_{W_i} and C_{W_i}, analogous to the physical masses and centroids of the Voronoi cells. Although there is a complex dependency between the robot position and the geometry of the Voronoi cells, a surprising result from locational optimization [3] is that the critical points of $\mathcal{W}$ correspond to the configurations in which all the robots are located at the centroid of their Voronoi cell, or $p_i = C_{W_i}$, for all i. Cortés introduced a gradient-based controller that is guaranteed to drive the robots to the critical points corresponding to local minimum [2] and which has been generalized to the weighted Voronoi cell ([4], [7]). We restrict ourselves to only considering local minima of $\mathcal{H}$ since global optimization of (2) is known to be difficult (NP-hard). Thus, when we refer to optimal coverage configurations, we mean locally optimal configurations. Variations on the control law which attempt to find global minima through exploration are discussed by Salapaka et al. [12] and Schwager et al. [13].

Our work utilizes an adaptive weighting that adjusts the robots weigh corresponding to variations in the actuation performance. Ideally, we could incorporate the actuator error into our cost function. Consider some function ζ_i, where ζ_i represents the actuation error quantity. We can then consider an ideal cost function as

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

In practice, we may not be able to calculate the value of ζ_i. Instead, the can look at our performance weights, which change based on the actuation error. These weights converge to some constant value $c = (w_i - \zeta_i)$. Consider then

$$\mathcal{W} = \sum_{i=1}^{n} \int_{W_i} \frac{1}{2} \left(\|q - p_i\|^2 - \zeta_i - c\right) \phi(q) dq$$

which reduces to

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

This allows us to incorporate the actuation error into our cost function while still measuring known quantities.

B. Robot and Actuator Model

In this section, we describe our model for the dynamics of the robots and the quality of the actuation. First, we assume that the robots have integrator dynamics, where

$$\begin{aligned} \dot{p}_i &= u_i + \Delta_i, \quad \text{and} \\ \dot{w}_i &= g_i. \end{aligned} \tag{4}$$

Here, u_i is the control input to the robot, Δ_i is an actuation uncertainty, and g_i is an adaptation law for the weightings. We can equivalently assume there are low-level controllers in place to cancel existing dynamics and enforce (4). We also assume that the robots will be able to communicate with their neighbors and share information about their actuation error. We define the communication network as an undirected graph in which two robots share an edge of the graph if they share Voronoi cell boundaries, also known as the Delaunay graph. We can then write the set of neighbors for any robot i as $\mathcal{N}_i := \{j | V_i \cap V_j \neq 0\}$. Additionally, robots are able to compute their own weighted Voronoi cells, as defined by (1), which is a common assumption in the literature ([2], [7], [12]).

III. Decentralized Control

The main goals of our work are to 1) drive the robots to an optimal coverage configuration in the environment and 2) adjust weightings to account for variations in performance in the positional controller. To accomplish these goals, we propose one control law to change the positions of the robots and one adaptation law to change the weightings of the robots. We will then prove that both of these control laws will drive the robots to converge asymptotically to a stable equilibrium configuration corresponding to a local minimum of the sensing cost function.

With respect to the position controller we will use

$$u_i = K_p(C_{W_i} - p_i),$$

where

$$K_p = \begin{bmatrix} k_p & 0 \\ 0 & k_p \end{bmatrix}$$

with k_p as a positive proportional gain constant. We will assume that the actuation errors can be quantified as

$$\Delta_i = K_{\Delta_i} K_p^{-1} u_i\,,$$

where K_{Δ_i} is a matrix. This yields the overall closed loop equation

$$\dot{p}_i = u_i + \Delta_i = K_i(C_{W_i} - p_i), \tag{5}$$

where C_{W_i} is the centroid of the weighted Voronoi cell and

$$K_i = K_p + K_{\Delta_i}.$$

This controller is a modification of the move-to-centroid control law, first proposed by Cortés ([2]) and extended and modified in ([14],[7], [10]). While the original control law used the unweighted Voronoi cell centroid, C_{V_i}, it does not impact the performance of the controller to use the weighted Voronoi centroid, C_{W_i} ([7], [5]). In the unlikely case of an empty Voronoi cell, we evaluate (5) using the integral form of the gradient descent based controller, letting $u_i = 0$.

For the purposes of this paper, we can consider the K_{Δ_i} matrix as one way to capture the imperfections in the movement of the robot. This paper considers the errors such that K_i is still a positive definite matrix. In practice, we may not know the exact value of K_{Δ_i}, so we will introduce an estimator, $\hat{K}_i$, derived from known quantities. From the estimate $\hat{K}_i$, the robot can then adjust for positional errors with the following adaptation law

$$\dot{w}_i = g_i = \frac{-k_w}{M_{W_i}} \sum_{j \in \mathcal{N}_i} \Big((w_i - f(\hat{K}_i)) - (w_j - f(\hat{K}_j)) \Big) \tag{6}$$

where k_w is a positive proportional gain constant and $f(\hat{K}_i)$ is some function of the properties of $\hat{K}_i$. Note that the choice of $f(\hat{K}_i)$ is subjective, based on the desired performance metrics of the system. The adaptation controller is inspired by consensus algorithms ([15],[16], [17]).

A. Estimating K_i

In practice, it is unlikely that the robots have direct knowledge of the actuator error K_{Δ_i}, so we will introduce an estimator, $\hat{K}_i$, that utilizes known quantities. We propose the following online estimator for the robots to use

$$\begin{aligned} \dot{\hat{K}}_i &= \lambda_i - \hat{K}_i \Lambda_i \\ \dot{\lambda}_i &= \dot{p}_i (C_{w_i} - p_i)^T \\ \dot{\Lambda}_i &= (C_{w_i} - p_i)(C_{w_i} - p_i)^T. \end{aligned} \tag{7}$$

We can further simplify this expression as

$$\begin{aligned} \dot{\hat{K}}_i &= (K_i - \hat{K}_i) \int (C_{W_i}(\tau) - p_i(\tau))(C_{W_i}(\tau) - p_i(\tau))^T d\tau \\ &= -\tilde{K}_i \Lambda_i(t) \end{aligned} \tag{8}$$

where

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

Note that while (7) and (8) are mathematically equivalent, the robots can only directly compute (7) because they do not have knowledge of the true error $\tilde{K}_i$. However, the form in (8) is useful for analysis.

B. Controller Performance

The behavior of our system with our positional control law, weightings adaptation law, and $\hat{K}_i$ estimator is formalized in the following theorem.

Theorem 1: Using the positional control law (5), weightings adaptation law (6), and estimator for $\hat{K}_i$ (7), the robots converge to the centroids of their weighted Voronoi cells,

$$\lim_{t \to \infty} \|p_i(t) - C_{W_i}(t)\| = 0 \;\; \forall \;\; i \in \{1, ..., n\}. \tag{9}$$

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

$$\lim_{t \to \infty} \tilde{K}_i(t) \Lambda_i(t) = 0 \;\; \forall \;\; i \in \{1, ..., n\}. \tag{10}$$

Proof:

To prove (9) and (10), we will invoke a global version of LaSalle's Invariance Principle ([18], Theorem 1.20). We will first introduce a continuously differentiable Lyapunov-like function V. We use this to show that show all trajectories of the system are bounded, and that the function is non-increasing, $\dot{V} \leq 0$. We then use LaSalle's Principle to prove the claim of the theorem. Consider the function

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

with derivative

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

We can break this into three parts as

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

By plugging in our controller (5) for $\dot{p}_i$, the time derivative $\dot{\mathcal{V}}_1$ becomes

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

Given that K_i is positive definite and M_{W_i} is positive, we know that overall $\dot{\mathcal{V}}_1 \leq 0$.

For $\dot{\mathcal{V}}_2$, plugging in our controller (6) for $\dot{w}_i$ yields

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

For $\dot{\mathcal{V}}_3$, we see that

$$\dot{\mathcal{V}}_3 = \text{Tr} \left[-\tilde{K}_i \Lambda_i(t) \tilde{K}_i^T \right] \leq 0.$$

Which allows us to say overall,

$$\dot{\mathcal{V}} \leq 0.$$

Given that $\dot{\mathcal{V}} \leq 0$, we see that the trajectories of both $p_i(t)$ and $\tilde{K}_i(t)$ are bounded, which also implies that $\hat{K}_i$ is bounded.

To determine whether the weightings are bounded, we can consider the vector form $\dot{w}$ as

$$\dot{w} = -k_w M^{-1} L w + k_w M^{-1} L f(t) \tag{11}$$

where M^{-1} is a diagonal matrix of positive entries, L is the Laplacian of the graph in which all Voronoi neighbors share an edge (the Delaunay graph), and

$$f(t) = \begin{bmatrix} f(\hat{K}_1(t)) \\ \cdots \\ f(\hat{K}_n(t)) \end{bmatrix}.$$

It is known that the graph Laplacian L is positive semi-definite ([19], [20]), and it can be shown that the product $M^{-1}L$ is positive semi-definite. Therefore $w(t)$ is the state of a marginally stable filter defined by (11). Also, since $\hat{K}_i$ is bounded for all i, $f(t)$ is bounded, and we have that the input to the filter defining $w(t)$ is bounded. From the input-to-state properties of marginally stable filters we know that for $w(t)$ to go unbounded the driving signal $k_w M^{-1} L f(t)$ must lie in the null space of the dynamics matrix $-k_w M^{-1} L$ of the filter. However, since the input $f(t)$ is itself multiplied by $k_w M^{-1} L$, the driving signal $k_w M^{-1} L f(t)$ can have no component in the null space of $-k_w M^{-1} L$, hence $w(t)$ remains bounded. This shows that all trajectories of the system ($p_i(t)$, $\tilde{k}_i(t)$, and $w_i(t)$ for all i) remain bounded.

Notice that we have already showed that $\dot{V} \leq 0$. Therefore, to complete the proof, we have to find the largest invariant set within the set defined by $\dot{V} = 0$. We can see that $\dot{V} = 0$ occurs when $p_i = C_{W_i}$, and $\tilde{K}_i \Lambda_i = 0$. From our control law (5) and estimator (8), we can see that this is itself 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 (9) and (10) from Theorem 1. ■

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

$$\lim_{t \to \infty} \hat{K}_i(t) = K_i \quad \forall \ i \in \{1, ..., n\}. \tag{12}$$

Furthermore,

$$\lim_{t \to \infty} (w_i(t) - w_j(t)) = f(K_i) - f(K_j) \tag{13}$$

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

Proof: In Theorem 1, we stated 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. Hence if at some $\tau > 0$, $\Lambda_i(\tau)$ has full rank, then $\Lambda_i(t)$ has full rank for all $t \geq \tau$, and the null space of $\Lambda_i(t)$ is the set comprised of the zero vector. Therefore for $\tilde{K}_i = (\hat{K}_i - K_i)$,

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

proving (12). Furthermore, as shown before, our weightings adaptation law (6) can be written in vector form as (11). By the properties of stable linear filters, we know that as the input approaches a limit, the state will also approach a limit [21], which will satisfy the steady state equation

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

where f_∞ is the limit of $f(t)$, written as

$$f_\infty = \begin{bmatrix} f(K_i) \\ \vdots \\ f(K_n) \end{bmatrix}.$$

Solving for w_∞ we find

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

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

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

proving (13) from Corollary 1. ■

Remark 1: In practice, we have seen in every simulation and experiment 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 highly unlikely given the nonlinear nature of the system. However, this fact is difficult to prove rigorously.

IV. Simulations

To demonstrate our move-to-centroid controller (5), weightings adaptation law (6), and $\hat{K}_i$ estimator (7), we conducted a series of simulations in Matlab. Using a rectangular environment Q and seven agents, we present three different scenarios. Scenario A initializes each agent with equal weights, but one of the robots has a decreased performance. Scenario B illustrates the case that all agents are performing equally, but one agent starts with a decreased weighting. Finally, Scenario C illustrates a randomized configuration to demonstrate a more complex configuration.

For each of the simulations, we chose the following function to measure the actuation performance

$$f(\hat{K}_i) = \|\hat{K}_i\|.$$

A. Scenario A

In this Scenario, we use a constant information density function $\phi(q)$, and all the agents have a zero K_{Δ_i} matrix, except agent 4, which has $K_{\Delta_4} = [-0.4, 0.0; 0.0, -0.4]$. Also, all initial weights are equal to one. Figure 1 shows the initial and final positions of the agents and Figure 2 shows the global sensing cost function and the performance of the agents, $w_i - \|\hat{K}_i\|$. The performance of agent 4 is in red.

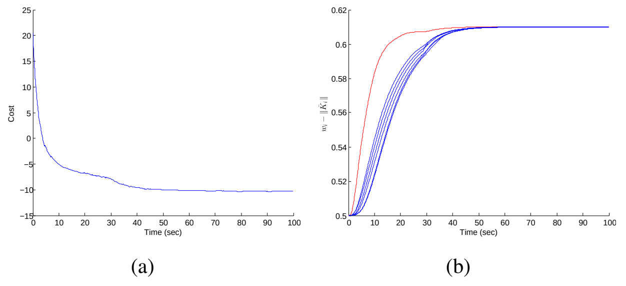

Fig. 2. (a) Global cost and (b) $(w_i - \|\hat{K}_i\|)$ for Scenario A.

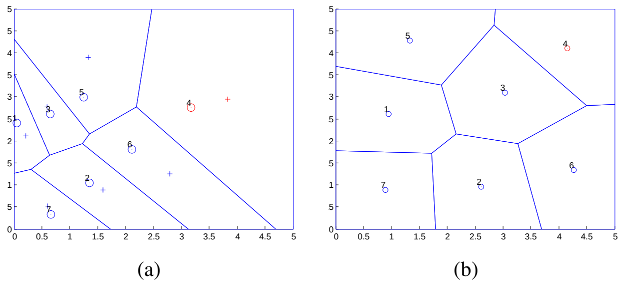

Fig. 1. (a) Initial and (b) final configurations in Scenario A.

Comparing the final and initial configurations, we can see that robot 4 has a smaller area on the final configuration, due to the weightings adaptation law. The values of the global cost function also decreased over time, settling to a local minimum once all agents reach their centroids and the weightings no long change. As expected, the set values of $(w_i - \|\hat{K}_i\|)$ converge to be equal across all agents. Finally, Figure 3 illustrates the convergence of the estimator $\hat{K}_i$ to K_i over time. Note that we only expect the estimator of agent 4 to change, shown in red. Over the course of the simulation, the estimator converges to the true value, K_i.

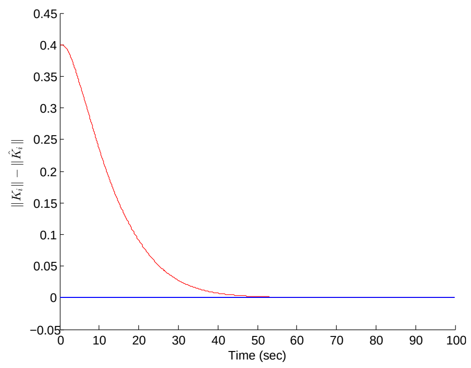

Fig. 3. Convergence of $\|\hat{K}_i\|$ to $\|K_i\|$

B. Scenario B

Scenario B also uses a constant $\phi(q)$, with all the robots performing equally, but robot 7 was assigned a lower initial weight $w_7 = 0.5$. Figure 4 shows the initial and final configurations of the agents. We can notice that in the initial configuration, robot 7 has a significantly smaller area than the other robots, but by the end of the simulation, its area is similar to the area of the other robots. Figure 5 shows both the cost function over time, as well as the convergence of $(w_i - \|\hat{K}_i\|)$.

In Figure 4, we can notice that robot 7 (in red) starts with a lower value of the performance function, but then all the robots converge to the same value. The cost function also decreases over time, which indicates that the robots reached a local minimum of the cost function. Note that since this scenario assumes all robots perform equally and at full health, our $\hat{K}_i$ estimator is the true K_i value over all time.

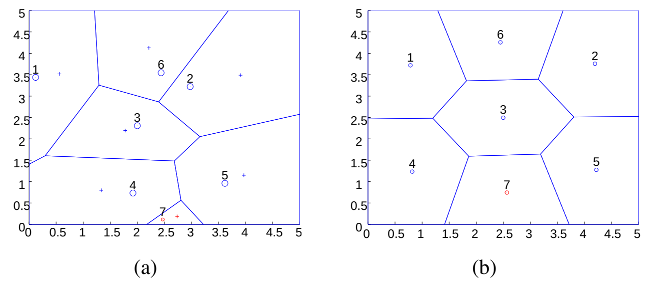
Fig. 4. (a) Initial and (b) final configurations for Scenario B.

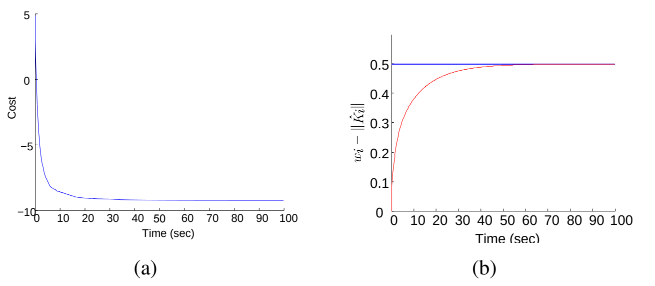
Fig. 5. Global sensing cost 5(a) and $(w_i - \|\hat{K}_i\|)$ 5(b) in Scenario B.

C. Scenario C

Scenario C has a constant information density function $\phi(q)$, but a randomized configuration of weights and K_{Δ_i} matrices. Overall, we set $K_{\Delta_1} = [-0.4, 0; 0, -0.4]$ and $K_{\Delta_6} = [0.2, 0; 0, 0.2]$. All initial weights are set $w_i = 1$, with the exception of $w_7 = 0.5$. Figure 6 shows the initial and final configuration of the agents, with robot 1 shown in red, robot 6 shown in green, and robot 7 shown in magenta.

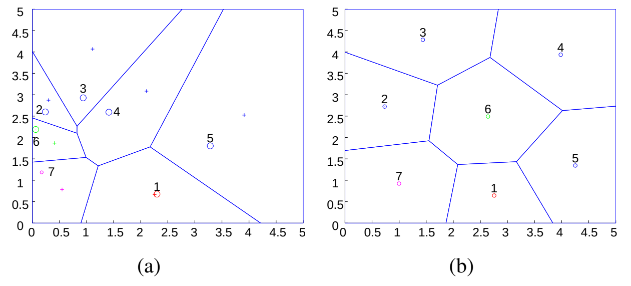
Fig. 6. Initial and final configurations in Scenario C.

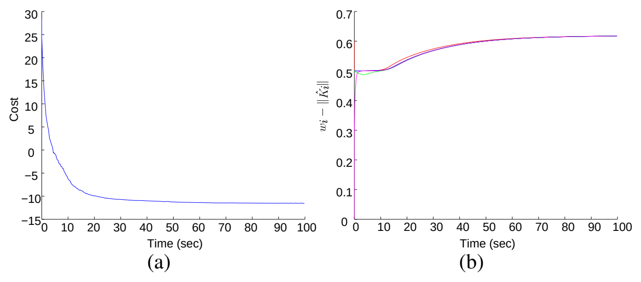
Fig. 7. Cost and $w_i - \|K_{\Delta_i}\|$ in Scenario C.

In Figure 7 we can see that the cost function decreases over time, indicating a successful simulation. We can also see that the performance of the agents, $(w_i - \|\hat{K}_i\|)$, converges to the expected set of values.

V. 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 in our Simulation section. A video with the experimental runs is attached to accompany this paper.

At the time the experiments were conducted, we had not finalized the $\hat{K}_i$ estimator. Instead, we assume the robots have full knowledge of their K_{Δ_i} matrix, which is analogous to the case that the estimator has already converged to the true K_i value. The experiments use the performance function

$$f(K_i) = \|K_{\Delta_i}\|.$$

In our experiments, we used the m3pi, a differential and compact robot from Pololu Robotics. For the Desktop/robot communication, we used XBee radios from Digi International. The m3pi robot utilizes an onboard mbed microcontroller to handle actuation and communication. The mbed handles the control of the motors on the robot, and we send velocity data to the robots based on the Voronoi calculations in Matlab.

To localize our robots, we used NaturalPoint's OptiTrack system with sixteen IR cameras. Each robot was equipped with a unique configuration of IR markers, as pictured in Figure 8, tracked by the OptiTrack system. We used two computers during the experiment, one running OptiTrack calculations, and the second performing the Voronoi calculations and communicating with the robots. The frame data was broadcast to the Matlab computer using a TCP/IP protocol, and information was sent from Matlab to the robots via XBee radios. Short-throw projectors were used to display the centroid and Voronoi boundaries on the floor mats during the experiments.

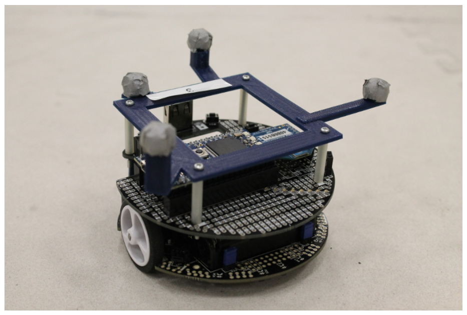
Fig. 8. One of the m3pi robots used in the experiments.

We used the same Matlab code used in the simulation with minor changes to deal with the OptiTrack localization system

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

and the control of the m3pi robots. We also incorporated a low-level point-offset controller [22] to account for the nonholonomic dynamics of the m3pis.

A. Scenario A

In this scenario, we used seven robots in a constant information density function $\phi(q)$ environment, all the robots with the same weight, equal to one, with $K_{\Delta_3} = [-0.5, 0; 0, -0.5]$. Figure 9 shows the initial and final configurations of the agents.

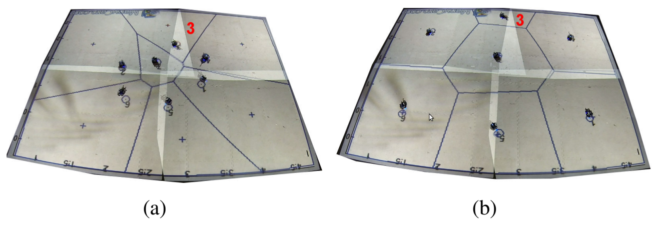

Fig. 9. Initial and final configurations in Scenario A.

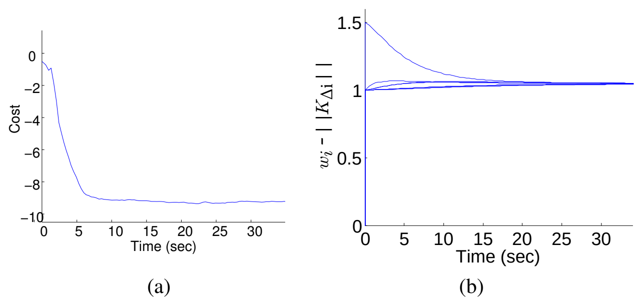

Fig. 10. Cost and $w_i - \|K_{\Delta_i}\|$ in Scenario A.

In Figure 10(a) we can see that the cost function decreases over time, which means that the group found a better configuration by following the move-to centroid controller. Also, in the Figure 10(b), the performance of the agents converged to a common value, validating the weightings controller.

B. Scenario B

In this scenario, we used again seven robots in a constant information density function $\phi(q)$ environment, all the robots have the same weight and performance, except robot 5, which has a lower weight. Figure 11 shows the initial and final configurations.

In this experiment, as all the robots have a zero K_{Δ_i} matrix, Figure 12(b) shows the weights of the agents. We can see that robot 5 starts with a lower weight, but its weight converges to the weights of the other agents. The cost function in Figure 12(a), also decreases over time, reaching a minimal value by the end of the simulation.

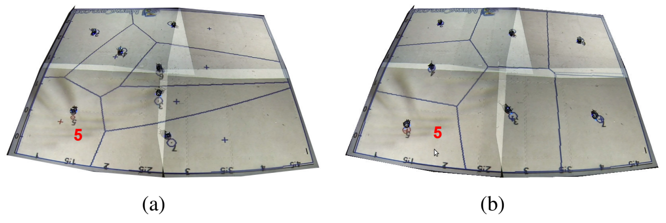

Fig. 11. Initial and final configurations in Scenario B.

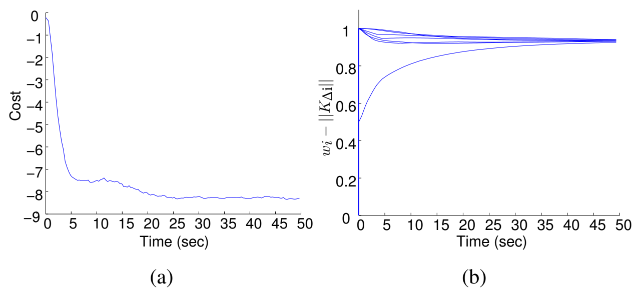

Fig. 12. Cost and $w_i - \|K_{\Delta_i}\|$ in Scenario B.

C. Scenario C

In Scenario C, we used a constant information density function $\phi(q)$. Robot 1 has $K_{\Delta_1} = [-0.5, 0; 0, -0.5]$ and $w_1 = 1$, robot 2 has $w_2 = 1.2$ and zero K_{Δ_2} matrix, and robot 5 has $w_5 = 0.75$ and also K_{Δ_5} matrix equal to zero. All the other robots have weights $w_i = 1$ and zero K_{Δ_i} matrix. Figure 13 shows the initial and final positions of the robots.

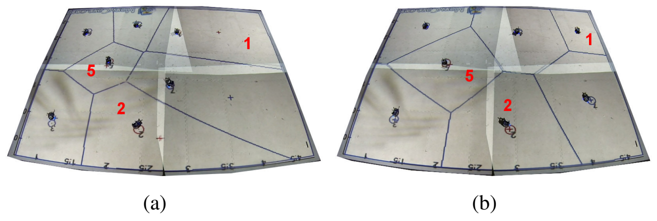

Fig. 13. Initial and final configurations in Scenario C.

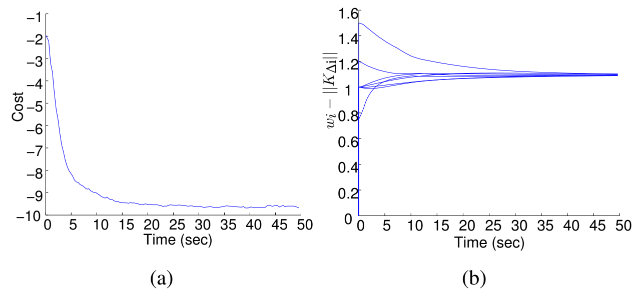

Fig. 14. Cost and $w_i - \|K_{\Delta_i}\|$ in Scenario C.

In this Scenario, we had 3 robots with different initial configurations, but in Figure 14, we can see that they all converge to the same value of performance function $w_i - \|K_{\Delta_i}\|$.

VI. Conclusion

In this paper, the authors have built upon their previous method of using adaptive weightings to adjust for individual variations in performance within multi-robot coverage control. To account for errors in actuation, the robots compare values of an error estimate with their neighbors, and using an adaptive weightings law, change the value of their weightings. 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. The positional controller is similar to previous implementations of Voronoi coverage control. We illustrate the success of our algorithm using m3pi robots.

Our method incorporates actuation 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. This can be achieved by adjusting for internal variations in actuation, as well as robustness against external factors such as rough terrain. It can also provide insight into identifying failures of a robot's actuation. A current limitation of the algorithm is the assumption that for each robot, $K_i > 0$. When considering simple actuation errors, this assumption of $K_i > 0$ is reasonable. Future work will examine the cases $K_i = 0$, which is akin to a failure, and $K_i < 0$, which could be one model of a malicious agent. It is also possible to model malicious agents with other control laws, and so other extensions may study the robustness of this algorithm to malicious agents. Another direction for future work ties to the author's previous work, where the weightings quantify both sensing and actuation performance within the group.

Acknowledgment

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

[1] J. Cortés, "Coverage optimization and spatial load balancing by robotic sensor networks," *Automatic Control, IEEE Transactions on*, vol. 55, no. 3, pp. 749–754, 2010.

[2] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *Robotics and Automation, IEEE Transactions on*, vol. 20, no. 2, pp. 243–255, 2004.

[3] Z. Drezner, *Facility location: a survey of applications and methods*, ser. Springer series in operations research. Springer, 1995.

[4] M. Pavone, A. Arsie, E. Frazzoli, and F. Bullo, "Equitable partitioning policies for robotic networks," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 2356–2361.

[5] L. Pimenta, V. Kumar, R. C. Mesquita, and G. Pereira, "Sensing and coverage for a network of heterogeneous robots," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*. IEEE, 2008, pp. 3947–3952.

[6] A. Kwok and S. Martinez, "Energy-balancing cooperative strategies for sensor deployment," in *Decision and Control, 2007 46th IEEE Conference on*. IEEE, 2007, pp. 6136–6141.

[7] J.-S. Marier, C.-A. Rabbath, and N. Léchevin, "Optimizing the location of sensors subject to health degradation," in *American Control Conference (ACC), 2011*. IEEE, 2011, pp. 3760–3765.

[8] J.-S. Marier, C. A. Rabbath, and N. Léchevin, "Health-aware coverage control with application to a team of small uavs," *IEEE Transactions on Control Systems Technology*, vol. 21, 2012.

[9] A. Pierson and M. Schwager, "Adaptive inter-robot trust for robust multi-robot sensor coverage," in *International Symposium on Robotics Research*, 2013.

[10] M. Schwager, D. Rus, and J.-J. Slotine, "Decentralized, adaptive coverage control for networked robots," *The International Journal of Robotics Research*, vol. 28, no. 3, pp. 357–375, 2009.

[11] S. Martínez, "Distributed interpolation schemes for field estimation by mobile sensor networks," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 419–500, March 2010.

[12] S. Salapaka, A. Khalak, and M. Dahleh, "Constraints on locational optimization problems," in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol. 2. IEEE, 2003, pp. 1741–1746.

[13] M. Schwager, F. Bullo, D. Skelly, and D. Rus, "A ladybug exploration strategy for distributed adaptive coverage control," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 2346–2353.

[14] A. Breitenmoser, M. Schwager, J.-C. Metzger, R. Siegwart, and D. Rus, "Voronoi coverage of non-convex environments with a group of networked robots," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4982–4989.

[15] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *Automatic Control, IEEE Transactions on*, vol. 48, no. 6, pp. 988–1001, 2003.

[16] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *Automatic Control, IEEE Transactions on*, vol. 49, no. 9, pp. 1520–1533, 2004.

[17] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[18] F. Bullo, J. Cortés, and S. Martinez, *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press, 2009.

[19] C. Godsil and G. Royle, *Algebraic Graph Theory*, ser. Graduate Texts in Mathematics. Springer-Verlag, 2001.

[20] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 1990.

[21] H. Khalil, *Nonlinear Systems*. Prentice Hall PTR, 2002.

[22] N. Michael and V. Kumar, "Planning and control of ensembles of robots with non-holonomic constraints," *Int. J. Rob. Res.*, vol. 28, no. 8, pp. 962–975, Aug. 2009. [Online]. Available: http://dx.doi.org/10.1177/0278364909340280