Decentralized, Adaptive Coverage Control for Networked Robots

Mac Schwager, Daniela Rus*, and Jean-Jacques Slotine†

November 1, 2007

Abstract

A decentralized, adaptive control law is presented to drive a network of mobile robots to an optimal sensing configuration. The control law is adaptive in that it uses sensor measurements to learn an approximation of the distribution of sensory information in the environment. It is decentralized in that it requires only information local to each robot. The controller is then improved upon by implementing a consensus algorithm in parallel with the learning algorithm, greatly increasing parameter convergence rates. Convergence and consensus of parameters is proven. Finally, several variations on the learning algorithm are explored with a discussion of their stability in closed loop. The controller with and without parameter consensus is demonstrated in numerical simulations. These techniques are suggestive of broader applications of adaptive control methodologies to decentralized control problems in unknown dynamical environments.

1 Introduction

In this paper we present a family of control strategies which are both adaptive and decentralized, thereby combining two of the defining qualities of biological systems. More importantly, the adaptive, decentralized control laws have provable stability and convergence properties. It is hoped that this work will provide a design methodology that can be applied to other problems requiring the control of distributed systems in unknown environments.

The specific problem we address is coverage control for networks of mobile robots. We consider controlling a group of mobile robots to monitor some quantity of interest over an area. Our solution to this problem would be useful in controlling teams of robots to carry out a number of tasks including environmental monitoring (e.g. for forest fires), automatic surveillance of rooms, buildings, or towns, or simulating collaborative predatory behavior. Virtually any application in which a group of automated mobile agents is required to monitor an area could benefit from the proposed control law.

Our control law causes the robots to spread over an area in such a way that their density in different regions of the area is directly related to the sensory importance of those regions. Thus regions of greater importance receive more concentrated coverage than regions that are less important. What is more, the robots do not know before hand what are the regions of sensory importance, but learn this information dynamically.[1] More precisely, each robot's controller assumes a linear parameterization of the sensory information in the environment. The controller continuously adapts its linear parameters to fit the robot's sensor measurements while the robot moves toward an optimal sensing position. We prove with a Lyapunov type proof that the network converges to a near-optimal sensing configuration, and to an optimal sensing configuration if the robot paths are sufficiently rich.

*Mac Schwager and Daniela Rus are with the Computer Science and Artificial Intelligence Lab, MIT, Cambridge, MA 02139, Email: schwager@mit.edu, rus@csail.mit.edu.

†Jean-Jacques Slotine is with the Nonlinear Systems Lab, MIT, Cambridge, MA 02139, Email: jjs@mit.edu.

[1]We will use the words learning and adaptation interchangeably. Learning and adaptation are specifically meant in the sense of parameter tuning, as in adaptive control, rather than the broader meaning often used in Biology and Bio-inspired applications.

We then improve upon the basic controller by introducing coupling in the parameter adaptation laws between neighboring robots. In other words, we implement a consensus algorithm in parallel with the learning algorithm. The main effect of this coupling is that sensor measurements from any one robot propagate around the network to be used by all robots. We prove convergence and consensus with parameter coupling and present results from numerical simulations that demonstrate its effectiveness in comparison to the control law without parameter coupling. Specifically, we show that convergence rates are greatly increased, a better final network configuration is achieved, and parameters for all robots in the network are guaranteed to converge to a common parameter vector. Figure 1 shows an overview of the control scheme.

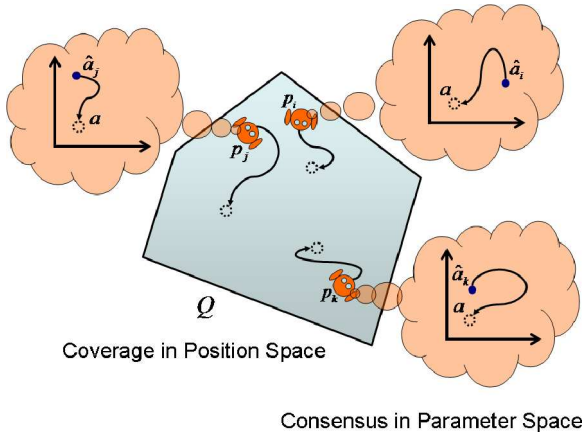


Figure 1: An overview of the decentralized control scheme is shown. The robots, at positions p_i, p_j, and p_k, spread out over the area, Q, to reach optimal final positions. Simultaneously, each robot adapts a parameter vector ($\hat{a}_i$, $\hat{a}_j$, and $\hat{a}_k$) to build an approximation of the sensory environment. For the consensus controller, the parameter vectors for neighboring robots are coupled in such a way that their final value, a, is the same for all robots in the network.

Additionally, we present several stable variations to the learning law itself. We define two families of learning algorithms, gradient algorithms and Least Squares algorithms, discuss their optimization-based interpretation, and prove convergence for each family in closed loop with the adaptive control law. Finally we describe the role of a data weighting function in the learning algorithm, and propose several examples of data weighting functions, again describing the virtues and drawbacks of each.

The picture that emerges is an adaptive control architecture which can modularly accommodate a combination of learning algorithms, parameter couplings, and data weighting functions while preserving stability. The design of an adaptive controller for a given coverage task then becomes a matter of balancing the trade-offs of each modular feature.

1.1 Relation to Previous Work

Multi-agent coordination problems have enjoyed a flurry of research activity in recent years ([1, 10, 27] to name only a few). Most relevant to our work is a body of results originating with [6], which introduced a formalism from locational optimization [8] to address the problem of optimal sensor positioning. More specifically, [6] proposed a stable, decentralized control law to drive a robot network to an optimal coverage configuration. Other works have investigated variations upon this control law [5,9,16], however in all of these works the robots are required to know *a priori* the distribution of sensory information in the environment. This requirement was relaxed in [20] by introducing a controller with a simple memoryless approximation from sensor measurements. The controller was demonstrated in hardware experiments, though a stability proof was not found. In the present work we introduce an adaptive controller inspired by the architecture in [17] with provable convergence properties to definitively remove the *a priori* knowledge requirement.

Unfortunately, the basic adaptive controller suffers from slow parameter convergence in numerical simulations. We address this problem in the present work by including a consensus term[2] in the parameter adaptation law. Consensus phenomena have been studied in many fields, and appear ubiquitously in biological systems of all scales. However, they have only recently yielded to rigorous mathematical treatment; first in the distributed and parallel computing community [2, 3, 28, 29] in discrete time, and more recently in the controls community in continuous time [4, 7, 11, 15, 30–32]. In the present work, consensus is used to learn the distribution of sensory information in the environment in a decentralized way by propagating sensor measurements gathered by each robot around the network. Consensus improves parameter convergence rates, which in turn causes the robots to converge more quickly to their optimal positions.

We also provide a discussion of different learning algorithms compatible with the adaptive controller and explain their interpretation as algorithms to solve optimization problems. Adaptive controllers have been formulated with a number of different learning schemes [14,18,24] and many authors have drawn the parallel with optimization (e.g. [24] Sec. 8.7, 8.8), so it is likely that the ones we present are not new in themselves. Of course, their application to the coverage control problem and the stability proofs that guarantee their convergence coupled with the coverage controller are new.

In short, the main contributions of this work are to 1) provide a controller that uses parameter adaptation to accomplish coverage without *a priori* knowledge of the sensory environment, 2) incorporate a consensus algorithm within the parameter adaptation law to propagate sensory information among the robots in the network, and 3) describe a number of variations to the parameter adaptation laws that have different optimization interpretations. Using a Lyapunov-like proof, we show that, regardless of which combination of learning and consensus features used, the control law causes the network to converge to a near-optimal sensing configuration, and if the robots' paths are sufficiently rich, the network will converge to an optimal configuration. The dynamics and the environment are assumed to exist in a deterministic setting throughout this work, as is typical in adaptive control. The results in this paper build upon our previous works [21, 22].

We set up the problem, provide some background on the results of locational optimization, and state the main assumptions and definitions in Section 2. We present the basic controller and prove convergence to an optimal coverage configuration in Section 3. The parameter consensus controller is introduced and parameter consensus is proved in Section 4. In Section 5 we discuss and compare parameter convergence rates for the consensus and basic controllers. Alternative adaptation laws are discussed in Section 6. Section 7 describes the results of numerical simulations, and conclusions are given in Section 8. We include tables of mathematical symbols in the Appendix.

2 Problem Set-up

Let there be n robots in a bounded, convex area $Q \subset \mathbb{R}^N$. An arbitrary point in Q is denoted q, the position of the i^{th} robot is denoted $p_i \in Q$. Let $\{V_1, ..., V_n\}$ be the Voronoi partition of Q, for which the robot positions are the generator points. Specifically,

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

The robots are assumed to have some physical extent, therefore no two robots can be at the same position, $p_i \neq p_j \ \forall i \neq j$. Define the sensory function $\phi : Q \mapsto \mathbb{R}$, where $\phi(q) > 0 \ \forall q \in Q$, that determines a weighting of sensory importance of points $q \in Q$. The function $\phi(q)$ is not known by the robots in the network, but the robots are equipped with sensors from which a measurement of $\phi(p_i)$ can be derived at the robot's position p_i.

Let the *unreliability* of the sensor measurement be defined by a quadratic function $\frac{1}{2}\|q-p_i\|^2$. Specifically, $\frac{1}{2}\|q-p_i\|^2$ describes how unreliable is the measurement of the information at q by a sensor at p_i (henceforth, $\|.\|$ is used to denote the ℓ^2-norm).

[2]The phenomenon of decentralized consensus is known by many names including flocking, herding, swarming, agreement, rendezvous, gossip algorithms, and oscillator synchronization. All of these are, at root, the same phenomenon—convergence of the states of a group of dynamical systems to a common final vector (or manifold) through local coupling.

2.1 Locational Optimization

In this section, we state the basic definitions and results from locational optimization that will be useful in this work. More thorough discussions can be found in [6, 20].

We can formulate the cost incurred by the network sensing over the region, Q, as

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

Notice that unreliable sensing is expensive and high values of the sensory function, $\phi(q)$, are also expensive. Qualitatively, a low value of $\mathcal{H}$ corresponds to a good coverage configuration for the robot network.

Next we define three properties analogous to mass-moments of rigid bodies. The mass, first moment, and centroid of a Voronoi region, V_i, are defined as

$$M_{V_i} = \textstyle\int_{V_i} \phi(q)\, dq, \quad L_{V_i} = \int_{V_i} q\phi(q)\, dq, \quad \text{and } C_{V_i} = L_{V_i}/M_{V_i}, \tag{2}$$

respectively. Note that $\phi(q)$ strictly positive imply both $M_{V_i} > 0$ and $C_{V_i} \in V_i \backslash \partial V_i$ (C_{V_i} is in the interior of V_i). Thus M_{V_i} and C_{V_i} have properties intrinsic to physical masses and centroids. A standard result in locational optimization is that

$$\frac{\partial \mathcal{H}}{\partial p_i} = -\int_{V_i} (q - p_i)\phi(q)\, dq = -M_{V_i}(C_{V_i} - p_i). \tag{3}$$

Equation (3) implies that local minima of $\mathcal{H}$ correspond to the configurations such that $p_i = C_{V_i}\ \forall i$, that is, each agent is located at the centroid of its Voronoi region. This brings us to the concept of optimal coverage summarized in the following definition.

Definition 1 (Optimal Coverage Configuration) *A robot network is said to be in a (locally) optimal coverage configuration if every robot is positioned at the centroid of its Voronoi region, $p_i = C_{V_i}\ \forall i$.*

We restrict ourselves to looking at local minima of $\mathcal{H}$ in this paper. Thus when we refer to an optimal coverage configuration we mean a locally optimal one. Variations on the control law which attempt to find global minima are discussed in [16, 19].

2.2 Assumptions and Definitions

Let the robots have dynamics

$$\dot{p}_i = u_i, \tag{4}$$

where u_i is the control input. We can equivalently assume there is a low-level controller in place to cancel existing dynamics and enforce (4).

The robots also are able to compute their own Voronoi cell, $V_i = \{q \mid \|q - p_i\| \leq \|q - p_j\|\}$. This assumption is common in the literature [6, 9, 16], though it presents a practical conundrum. One does not know beforehand how far away the farthest Voronoi neighbor will be, thus this assumption cannot be translated into a communication range constraint (aside from the overly conservative requirement for each robot to have a communication range as large as the diameter of Q). In practice, only Voronoi neighbors within a certain distance will be in communication, in which case results can be derived, though with considerable complication [5]. We will take this assumption as implicit and leave the burden of relaxing this constraint for future work. Indeed, the results of [5] suggest that performance degrades gracefully with decreasing communication range among robots.

More central to this work, we assume that the sensory function, $\phi(q)$, can be parameterized as an unknown linear combination of a set of known basis functions. This requirement is formalized in the following two assumptions.

Assumption 1 (Matching Conditions) *$\exists a \in \mathbb{R}^m$ and $\mathcal{K} : Q \mapsto \mathbb{R}^m$, where $\mathcal{K}(q) > 0 \ \forall q \in Q$, such that*

$$\phi(q) = \mathcal{K}(q)^T a, \tag{5}$$

where the vector of basis functions, $\mathcal{K}(q)$, is known by each agent, but the parameter vector, a, is unknown.

Assumption 2 (Lower Bound)

$$a(j) \geq a_{\min} \quad \forall j = 1, \ldots, m, \tag{6}$$

where $a(j)$ denotes the j^{th} element of the vector a, and $a_{\min} > 0$ is a lower bound known by each robot.

Let $\hat{a}_i(t)$ be robot i's approximation of the parameter vector. Naturally, $\hat{\phi}_i(q,t) = \mathcal{K}(q)^T \hat{a}_i(t)$ is robot i's approximation of $\phi(q)$. Figure 2 shows a graphical representation of this function approximation scheme. The figure shows the basis functions as Gaussians, since they are a common choice, though they could also be wavelets, sigmoids, splines, or any number of other basis function families.

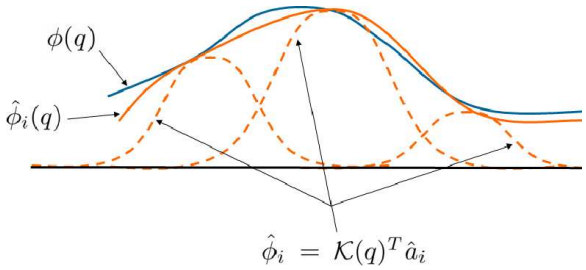


Figure 2: The sensory function approximation is illustrated in this simplified 2-D schematic. The true sensory function is represented by ϕ (blue curve) and robot i's approximation of the sensory function is $\hat{\phi}_i$ (orange curve). The basis function vector $\mathcal{K}(q)$ is shown as three Gaussians (dashed curves), and the parameter vector $\hat{a}_i$ denotes the weighting of each Gaussian. According to Assumption 1 there is some value of $\hat{a}_i$ that makes the approximation equal to the true function.

Define the moment approximations

$$\hat{M}_{V_i}(t) = \int_{V_i} \hat{\phi}_i(q,t)\, dq, \quad \hat{L}_{V_i} = \int_{V_i} q\hat{\phi}_i(q,t)\, dq, \quad \text{and } \hat{C}_{V_i}(t) = \hat{L}_{V_i}(t)/\hat{M}_{V_i}(t). \tag{7}$$

Next, define the parameter error

$$\tilde{a}_i(t) = \hat{a}_i(t) - a, \tag{8}$$

and notice the relation

$$\hat{\phi}_i(q,t) - \phi(q) = \mathcal{K}(q)^T \tilde{a}_i(t). \tag{9}$$

In order to compress the notation, we introduce the shorthand $\mathcal{K}_i(t) = \mathcal{K}(p_i(t))$ for the value of the basis function vector at the position of robot i, and $\phi_i(t) = \phi(p_i(t))$ for the value of the sensory function at the position of robot i. As previously stated, robot i can measure ϕ_i with its sensors. We will also commonly refer to quantities without explicitly writing their arguments. However, we may include arguments in some instances to avoid ambiguity.

The function approximation framework described above brings us to another concept of optimality for coverage.

Definition 2 (Near-Optimal Coverage Configuration) *A robot network is said to be in a near-optimal coverage configuration if each robot is positioned at the estimated centroid of its Voronoi region, $p_i = \hat{C}_{V_i}$ $\forall i$.*

Finally, we distinguish between two qualities of function approximations.

Definition 3 (Globally True Approximation) *A robot is said to have a globally true (or just true) approximation of the sensory function if its approximation is equal to the actual sensory function at every point of its domain,* $\hat{\phi}_i(q) = \phi(q)\ \forall q \in Q$.

Definition 4 (Locally True Approximation) *A robot is said to have a locally true approximation of the sensory function over a subset* $\Omega_i \subset Q$ *if its approximation is equal to the true function at every point in the subset,* $\hat{\phi}_i(q) = \phi(q)\ \forall q \in \Omega_i$.

In light of the above definitions, if the parameter error is zero, $\tilde{a}_i = 0$, then robot i has a *true* approximation of the sensory function. Also, if $\tilde{a}_i = 0\ \forall i$, then a *near-optimal* coverage configuration is also *optimal*.

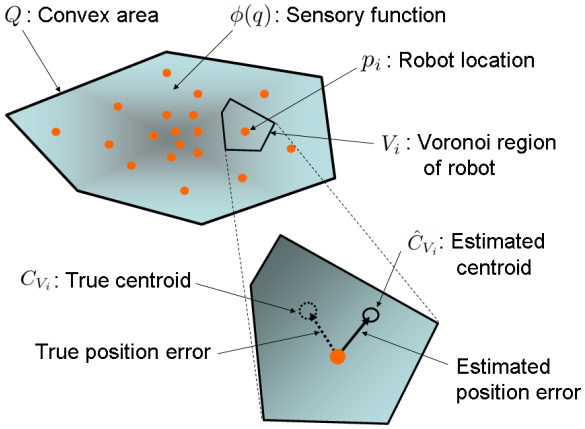


Figure 3: A graphical overview of the quantities involved in the controller and environment is shown. The robots move to cover a bounded, convex area Q their positions are p_i, and they each have a Voronoi region V_i with a true centroid C_{V_i} and an estimated centroid $\hat{C}_{V_i}$. The true centroid is determined using a sensory function $\phi(q)$, which indicates the relative importance of points q in Q. The robots do not know $\phi(q)$, so they calculate an estimated centroid using an approximation $\hat{\phi}_i(q)$ learned from sensor measurements of $\phi(q)$.

3 Decentralized Adaptive Control Law

We will design a control law with an intuitive interpretation and prove that it causes the network to converge to a near-optimal coverage configuration, and an optimal configuration if the robots' paths are sufficiently rich. The control law will integrate sensory measurements available to each robot to form an on-line approximation of the sensory function parameters, which will be used to calculate an approximation of the centroid of its Voronoi region.

We propose to use the control law

$$u_i = k(\hat{C}_{V_i} - p_i), \tag{10}$$

where $k \in \mathbb{R}$, $k > 0$, is a proportional control gain. The area Q is required to be convex so that the control law is feasible, that is, the robots never attempt to cross the boundaries of Q. Since $\hat{C}_{V_i} \in Q$ and $p_i \in Q$, by convexity, the segment connecting the two is in Q, and the control law is feasible.

The parameters $\hat{a}_i$ used to calculate $\hat{C}_{V_i}$ are adjusted according to a set of adaptation laws which are introduced below. First, define two quantities,

$$\Lambda_i = \int_0^t w(\tau)\mathcal{K}_i(\tau)\mathcal{K}_i(\tau)^T\, d\tau, \quad \text{and} \quad \lambda_i = \int_0^t w(\tau)\mathcal{K}_i(\tau)\phi_i(\tau)\, d\tau. \tag{11}$$

The function $w(t) \in \mathcal{L}^1$, where $w(t) \geq 0$, determines a data collection weighting. Note that these quantities can be calculated differentially by robot i using $\dot{\Lambda}_i = w(t)\mathcal{K}_i\mathcal{K}_i^T$, and $\dot{\lambda}_i = w(t)\mathcal{K}_i\phi_i$, with zero initial conditions. Define another quantity

$$F_i = \frac{\int_{V_i} \mathcal{K}(q)(q-p_i)^T\,dq \int_{V_i}(q-p_i)\mathcal{K}(q)^T\,dq}{\int_{V_i}\hat{\phi}_i(q)\,dq}. \tag{12}$$

Notice that F_i is a positive semi-definite matrix. It can also be computed by robot i as it does not require any knowledge of the true parameter vector, a. The adaptation law for $\hat{a}_i$ is now defined as

$$\dot{\hat{a}}_{\mathrm{pre}_i} = -F_i\hat{a}_i - \gamma(\Lambda_i\hat{a}_i - \lambda_i), \tag{13}$$

$$\dot{\hat{a}}_i = \Gamma(\dot{\hat{a}}_{\mathrm{pre}_i} - I_{\mathrm{proj}_i}\dot{\hat{a}}_{\mathrm{pre}_i}), \tag{14}$$

where $\Gamma \in \mathbb{R}^{m\times m}$ is a diagonal, positive definite adaptation gain matrix, and $\gamma \in \mathbb{R}$, $\gamma > 0$, is an adaptation gain scalar. The diagonal matrix I_{proj_i} is defined element-wise as

$$I_{\mathrm{proj}_i}(j) = \begin{cases} 0 & \text{for } \hat{a}_i(j) > a_{\min} \\ 0 & \text{for } \hat{a}_i(j) = a_{\min} \text{ and } \dot{\hat{a}}_{\mathrm{pre}_i}(j) \geq 0 \\ 1 & \text{otherwise} \end{cases}, \tag{15}$$

where (j) denotes the j^{th} element for a vector and the j^{th} diagonal element for a matrix. Equations (14) and (15) implement a projection operation [25] that prevents any element of $\hat{a}_i$ from dropping below the lower bound $a_{\min}$. This is done by forcing $\dot{\hat{a}}_i(j) = 0$ whenever $\hat{a}_i(j) = a_{\min}$ and $\dot{\hat{a}}_{\mathrm{pre}_i}(j) < 0$. The projection is desirable for two reasons: 1) because the control law has a singularity at $\hat{a}_i = 0$, since $\hat{M}_{V_i} = 0$ at this point, and $\hat{C}_{V_i}$ becomes unbounded, and 2) because we know from Assumption 2 that the true parameters are lower bounded by $a_{\min}$.

The controller described above will be referred to as the basic controller, and its behavior is formalized in the following theorem.

Theorem 1 (Basic Convergence) *Under Assumptions 1 and 2, the network of robots with dynamics (4), control law (10), and adaptation law (13,14) converges to a* near-optimal *coverage configuration. Furthermore, each robot converges to a* locally true *approximation of the sensory function over the set* $\Omega_i = \{p_i(\tau) \mid \tau \geq 0, w(\tau) > 0\}$*, made up of all points on the robot's trajectory with positive weighting.*

Proof

We will define a Lyapunov-like function and show that its time derivative is non-increasing. The quantities $\|\hat{C}_{V_i}(t) - p_i(t)\|$ and $w(\tau)(\hat{\phi}_i(p_i(\tau),t) - \phi(p_i(\tau)))^2$, $0 \geq \tau \geq t$, will be included in the time derivative of this function for all i. We will show using Barbalat's lemma, that the time derivative converges to zero, thereby implying $p_i(t) \to \hat{C}_{V_i}(t)$, and $\hat{\phi}_i(q,t) \to \phi(q)\ \forall q \in \Omega_i$ for all i.

Define a Lyapunov-like function

$$\mathcal{V} = \mathcal{H} + \sum_{i=1}^{n} \frac{1}{2}\tilde{a}_i^T k\Gamma^{-1}\tilde{a}_i, \tag{16}$$

which incorporates the sensing cost $\mathcal{H}$, and is quadratic in the parameter errors $\tilde{a}_i$ (recall $k > 0$ and $\Gamma > 0$ are control and adaptation gains respectively). It is bounded below by zero since $\mathcal{H}$ is a sum of integrals of strictly positive functions, and the quadratic parameter error terms are each bounded below by zero.

Taking the time derivative of $\mathcal{V}$ along the trajectories of the system gives

$$\dot{\mathcal{V}} = \sum_{i=1}^{n}\left[\frac{\partial\mathcal{H}}{\partial p_i}^T \dot{p}_i + \tilde{a}_i^T k\Gamma^{-1}\dot{\tilde{a}}_i\right],$$

and substituting from (3) and noticing that $\dot{\tilde{a}}_i = \dot{\hat{a}}_i$ yields

$$\dot{\mathcal{V}} = \sum_{i=1}^{n}\left[-\int_{V_i}(q-p_i)^T\phi(q)\,dq\dot{p}_i + \tilde{a}_i^T k\Gamma^{-1}\dot{\hat{a}}_i\right].$$

Using (9) to substitute for $\phi(q)$ gives

$$\dot{\mathcal{V}} = \sum_{i=1}^{n} \big[- \int_{V_i} (q - p_i)^T \hat{\phi}_i \, dq\dot{p}_i + \int_{V_i} \tilde{a}_i^T \mathcal{K}(q)(q - p_i)^T \, dq\dot{p}_i + \tilde{a}_i^T k\Gamma^{-1}\dot{\hat{a}}_i \big].$$

Substituting for $\dot{p}_i$ with (4) and (10), and pulling $\tilde{a}_i$ out of the second integral (since its not a function of q) leads to

$$\dot{\mathcal{V}} = \sum_{i=1}^{n} \big[- \hat{M}_{V_i} k \|\hat{C}_{V_i} - p_i\|^2 + \tilde{a}_i^T k \int_{V_i} \mathcal{K}(q)(q - p_i)^T \, dq(\hat{C}_{V_i} - p_i) + \tilde{a}_i^T k\Gamma^{-1}\dot{\hat{a}}_i \big].$$

Expanding $(\hat{C}_{V_i} - p_i)$ in the second term, and substituting for $\dot{\hat{a}}_i$ with (14) gives

$$\dot{\mathcal{V}} = \sum_{i=1}^{n} \big[- \hat{M}_{V_i} k \|\hat{C}_{V_i} - p_i\|^2 + \tilde{a}_i k F_i \hat{a}_i - \tilde{a}_i^T k F_i \hat{a}_i - \tilde{a}_i^T k\gamma(\Lambda_i \hat{a}_i - \lambda_i) - \tilde{a}_i^T k I_{\text{proj}_i} \dot{\hat{a}}_{\text{pre}_i} \big].$$

Now we can expand $(\Lambda_i \hat{a}_i - \lambda_i)$, noting that $\lambda_i = \int_0^t w(\tau)\mathcal{K}_i(\mathcal{K}_i^T a)\, d\tau$, to get

$$\dot{\mathcal{V}} = -\sum_{i=1}^{n} \Big[\hat{M}_{V_i} k \|\hat{C}_{V_i} - p_i\|^2 + \tilde{a}_i^T k\gamma \int_0^t w(\tau)\mathcal{K}_i\mathcal{K}_i^T \tilde{a}_i(t)\, d\tau + \tilde{a}_i^T k I_{\text{proj}_i} \dot{\hat{a}}_{\text{pre}_i} \Big],$$

and, finally, bringing $\tilde{a}_i^T$ inside the integral (it is not a function of τ, though it is a function of t) results in

$$\dot{\mathcal{V}} = -\sum_{i=1}^{n} \Big[\hat{M}_{V_i} k \|\hat{C}_{V_i} - p_i\|^2 + k\gamma \int_0^t w(\tau)(\mathcal{K}_i(\tau)^T \tilde{a}_i(t))^2 \, d\tau + \tilde{a}_i^T k I_{\text{proj}_i} \dot{\hat{a}}_{\text{pre}_i} \Big], \tag{17}$$

Inside the sum, the first and second terms are clearly non-negative. We focus momentarily on the third term. Expanding it as a sum of scalar terms, we see that the j^{th} scalar term is of the form

$$k\tilde{a}_i(j) I_{\text{proj}_i}(j) \dot{\hat{a}}_{\text{pre}_i}(j). \tag{18}$$

From (15), if $\hat{a}_i(j) > a_{\min}$, or $\hat{a}_i(j) = a_{\min}$ and $\dot{\hat{a}}_{\text{pre}_i}(j) \geq 0$, then $I_{\text{proj}_i}(j) = 0$ and the term vanishes. Now, in the case $\hat{a}_i(j) = a_{\min}$ and $\dot{\hat{a}}_{\text{pre}_i}(j) < 0$, we have $\tilde{a}_i(j) = \hat{a}_i(j) - a(j) \leq 0$ (from Assumption 2). Furthermore, $I_{\text{proj}_i}(j) = 1$ and $\dot{\hat{a}}_{\text{pre}_i}(j) < 0$ implies that the term is non-negative. In all cases, then, each term of the form (18) is non-negative, and all three terms inside the sum in (17) are non-negative. Thus $\dot{\mathcal{V}} \leq 0$.

Also, the facts that u_i is continuous $\forall i$, $\mathcal{V}$ has continuous first partial derivatives, $\mathcal{V}$ is radially unbounded, and $\dot{\mathcal{V}} \leq 0$ imply that $\dot{\mathcal{V}}$ is uniformly continuous, therefore, by Barbalat's lemma $\lim_{t\to\infty} \dot{\mathcal{V}} = 0$. This, in turn, implies $\lim_{t\to\infty} \|p_i(t) - \hat{C}_{V_i}(t)\| = 0 \ \forall i$ (from the first term in the sum), so the network converges to a near-optimal coverage configuration.

Furthermore, from $\mathcal{K}_i(\tau)^T \tilde{a}_i(t) = \hat{\phi}_i(p_i(\tau), t) - \phi(p_i(\tau))$, we have in the second term

$$\lim_{t\to\infty} \int_0^t w(\tau)(\hat{\phi}_i(p_i(\tau), t) - \phi(p_i(\tau)))^2 \, d\tau = 0 \quad \forall i = 1, \ldots, n. \tag{19}$$

Now notice that the integrand in (19) is non-negative, therefore it must converge to zero for all τ, which implies $\hat{\phi}_i(q,t) \to \phi(q)\ \forall q \in \Omega_i$ and $\forall i$. $\square$

In [19] the following extension to the above theorem was derived. We restate it here to give a more thorough characterization the controller's behavior.

Corollary 1 (Sufficient Richness) *In addition to the conditions for Theorem 1, if the robots' paths are such that the matrix* $\lim_{t\to\infty} \Lambda_i(t)$ *is positive definite* $\forall i$*, the network converges to an* optimal *coverage configuration, and each robot converges to a* globally true *approximation of the sensory function,* $\phi(q)$*.*

Proof

Consider the second term in (17). Take the two $\tilde{a}_i(t)$ outside of the integral (since they are not a function of τ) to get

$$k\gamma\tilde{a}_i(t)^T \left[\int_0^t w(\tau)\mathcal{K}_i\mathcal{K}_i^T\, d\tau\right]\tilde{a}_i(t) = k\gamma\tilde{a}_i(t)^T\Lambda_i(t)\tilde{a}_i(t).$$

Since $\dot{\mathcal{V}} \to 0$, if $\lim_{t\to\infty}\Lambda_i(t)$ is positive definite (we know the limit exists because $\mathcal{K}(q)$ is bounded and $w(t) \in \mathcal{L}^1$), then $\tilde{a}_i(t) \to 0$. This implies that robot i converges to a *globally true* approximation of the sensory function, $\phi(q)$. Furthermore, if $\lim_{t\to\infty}\Lambda_i(t) > 0\ \forall i$, then $\hat{C}_{V_i} = C_{V_i}\ \forall i$, so the network converges to an *optimal* coverage configuration. $\square$

4 Parameter Consensus

In this section we first state some elementary properties of graph Laplacians, then use these properties to prove convergence and consensus of a modified adaptive control law. The controller from (3) is modified so that the adaptation laws among Voronoi neighbors are coupled causing parameter consensus. Adaptation and consensus were also combined in [32], however in that case it was the agent velocities that reached consensus, not the learning parameters.

4.1 Graph Laplacians

A graph $G = (V, E)$ is defined by a set of indexed vertices $V = \{v_1, \ldots, v_n\}$ and a set of edges $E = \{e_1, \ldots, e_l\}$, $e_i = \{v_j, v_k\}$. In the context of our application, a graph is induced in which each agent is identified with a vertex, and an edge exists between any two agents that are Voronoi neighbors. This graph is that of the Delaunay triangulation

Let $\mathcal{N}_i = \{j \mid \{v_i, v_j\} \in E\}$ be the neighbor set of vertex v_i. Then $|\mathcal{N}_i|$ is the number of Voronoi neighbors of agent i. Let A be the adjacency matrix of G, defined element wise by

$$A(i,j) = A(j,i) = \begin{cases} 1 & \text{for } \{v_i, v_j\} \in E \\ 0 & \text{otherwise.} \end{cases}$$

The graph Laplacian is defined as $L = \operatorname{diag}_{i=1}^{n}(|\mathcal{N}_i|) - A$.

Loosely, a graph is connected if there exists a set of edges that defines a path between any two vertices. The graph of any triangulation is connected, specifically, the graph is connected in our application. It is well known that for a connected graph $L \geq 0$ and L has exactly one zero eigenvalue, with the associated eigenvector $\mathbf{1} = [1, \ldots, 1]^T$. In particular, $L\mathbf{1} = \mathbf{1}^T L = 0$, and $x^T L x > 0$, $\forall x \neq c\mathbf{1}, c \in \mathbb{R}$. These properties will be important in what follows.

4.2 Consensus Learning Law

We add a term to the parameter adaptation law in (13) to couple the adaptation of parameters between neighboring agents. Let the new adaptation law be given by

$$\dot{\hat{a}}_{\text{pre}_i} = -F_i\hat{a}_i - \gamma\left(\Lambda_i\hat{a}_i - \lambda_i\right) - \zeta\sum_{j\in\mathcal{N}_i}(\hat{a}_i - \hat{a}_j), \tag{20}$$

where $\mathcal{N}_i$ is the neighbor set defined above, and $\zeta \in \mathbb{R}$, $\zeta > 0$, is a positive gain. The projection remains the same as in (14), namely

$$\dot{\hat{a}}_i = \Gamma(\dot{\hat{a}}_{\text{pre}_i} - I_{\text{proj}_i}\dot{\hat{a}}_{\text{pre}_i}).$$

Theorem 2 (Convergence with Parameter Consensus) *Under the conditions of Theorem 1, using the parameter adaptation law (20), the claims from Theorem 1 are true. Additionally,*

$$\lim_{t\to\infty}(\hat{a}_i - \hat{a}_j) = 0 \quad \forall i, j \in \{1, \ldots, n\}. \tag{21}$$

Proof

We will use the same method as in the proof of Theorem 1, adding the extra term for parameter coupling. It will be shown that this term is non-positive. The claims of the proof follow as before from Barbalat's lemma.

Define $\mathcal{V}$ to be (16), which leads to

$$\dot{\mathcal{V}} = -\sum_{i=1}^{n}\Big[\hat{M}_{V_i}k\|\hat{C}_{V_i} - p_i\|^2 + k\gamma\int_0^t w(\tau)(\mathcal{K}_i(\tau)^T\tilde{a}_i(t))^2\,d\tau + \tilde{a}_i^T k I_{\text{proj}_i}\dot{\hat{a}}_{\text{pre}_i}\Big] - \sum_{i=1}^{n}\tilde{a}_i^T k\zeta\sum_{j\in\mathcal{N}_i}(\hat{a}_i - \hat{a}_j). \tag{22}$$

We have already shown that the three terms inside the first sum are non-negative. Now consider the parameter coupling term. We can rewrite this term using the graph Laplacian defined in Section 4.1 as

$$\sum_{i=1}^{n}\tilde{a}_i^T k\zeta\sum_{j\in\mathcal{N}_i}(\hat{a}_i - \hat{a}_j) = k\zeta\sum_{j=1}^{m}\tilde{\alpha}_j^T L(t)\hat{\alpha}_j,$$

where $\alpha_j = a(j)\mathbf{1}$, $\hat{\alpha}_j = [\hat{a}_1(j) \quad \cdots \quad \hat{a}_n(j)]^T$, and $\tilde{\alpha}_j = \hat{\alpha}_j - \alpha_j$. Recall the ideal parameter vector $a = [a(1) \quad \cdots \quad a(j) \quad \cdots \quad a(m)]^T$, and the parameter estimate for each agent $\hat{a}_i = [\hat{a}_i(1) \quad \cdots \quad \hat{a}_i(j) \quad \cdots \quad \hat{a}_i(m)]^T$. We have simply regrouped the parameters by introducing the α_j notation. The Laplacian is a function of time since as the agents move around they may acquire new neighbors or loose old ones. Fortunately, we are guaranteed that $L(t)$ will have the properties discussed in Section 4.1 for all $t \geq 0$.

From Section 4.1 we saw that $\alpha_j^T L(t) = a(j)\mathbf{1}^T L = 0$. This gives

$$k\zeta\sum_{j=1}^{m}\tilde{\alpha}_j^T L\hat{\alpha}_j = k\zeta\sum_{j=1}^{m}\hat{\alpha}_j^T L\hat{\alpha}_j \geq 0,$$

since $L(t) \geq 0\ \forall t \geq 0$.

Thus $\dot{\mathcal{V}} \leq 0$. The previous argument still applies for the uniform continuity of $\dot{\mathcal{V}}$, therefore, by Barbalat's lemma $\lim_{t\to\infty}\dot{\mathcal{V}} = 0$. As before this implies the two claims of Theorem 1. Since the graph Laplacian is positive semi-definite, and $\hat{a}_i(j) \geq a_{\min}$, $\lim_{t\to\infty}\hat{\alpha}_j^T L(t)\hat{\alpha}_j = 0 \Rightarrow \lim_{t\to\infty}\hat{\alpha}_j = a_{\text{final}}(j)\mathbf{1}\ \forall j \in \{1, \ldots, m\}$, where $a_{\text{final}} \in \mathbb{R}^m$ is some undetermined vector, which is the common final value of the parameters for all of the agents. The consensus assertion (21) follows.□

Remark 1 *To guarantee that the network converges to an* optimal *coverage configuration, and that the robots converge to a* globally true *approximation of the sensory function, we require an extra sufficient richness condition, which takes the form of a relaxed version of Corollary 1. This will be discussed in Section 5.*

Remark 2 *Introducing parameter coupling greatly increases parameter convergence rates and makes the controller equations better conditioned for numerical integration, as will be discussed in Section 7. However there is a cost in increased communication overhead. With the parameter consensus controller the robots must communicate their parameter vector (m floating point numbers) in addition to their nominal communication overhead. Even with a low bandwidth communication system, this should represent a negligible cost.*

5 Parameter Convergence Analysis

In this section we show that parameter convergence is not exponential, though if the robot's trajectory is sufficiently rich, the parameter dynamics can be represented as a stable linear system driven by a signal that converges to zero. In other words, parameter convergence has a number of exponential modes and a number of (presumably slower) asymptotic modes. The exponential modes are shown to be faster for the controller with parameter consensus than the one without. In this section we neglect the projection operation (14), as the discrete switching considerably complicates the convergence analysis.

Suppose robot i has a sufficiently rich trajectory after some time $T > 0$, so that $\lambda_{\min_i} > 0$ is the minimum eigenvalue of $\Lambda_i(T)$. Then from (13) we have

$$\dot{\hat{a}}_i = -\Gamma(F_i\hat{a}_i + \gamma(\Lambda_i\hat{a}_i - \lambda_i)),$$

which can be written as

$$\dot{\tilde{a}}_i = -\Gamma\gamma\Lambda_i(t)\tilde{a}_i - \Gamma F_i\hat{a}_i, \tag{23}$$

leading to

$$\dot{\tilde{a}}_i \leq -\Gamma\gamma\lambda_{\min_i}\tilde{a}_i - \Gamma F_i\hat{a}_i \quad \forall t \geq T, \tag{24}$$

Equation (24) implies that, after some time T, $\tilde{a}_i$ is at least as fast as an exponentially stable system driven by the signal $-\Gamma F_i\hat{a}_i$. We proved in Theorem 1 that $(\hat{C}_{V_i} - p_i) \to 0$ and all other quantities in $F_i\hat{a}_i$ are bounded, therefore $F_i\hat{a}_i \to 0$, but we cannot prove that it does so exponentially. However, the gains Γ and γ can be set such that $\Gamma F_i\hat{a}_i$ is arbitrarily small compared to $\Gamma\gamma\Lambda_i(t)$ without affecting stability. Thus exponentially fast convergence to an arbitrarily small parameter error can be achieved.

For the parameter consensus controller, from (20) we have

$$\dot{\hat{a}}_i = -\Gamma\Big(F_i\hat{a}_i + \gamma(\Lambda_i\hat{a}_i - \lambda_i) + \zeta\sum_{j\in\mathcal{N}_i}(\hat{a}_i - \hat{a}_j)\Big).$$

Recall that for the basic controller, conditions for parameter convergence and sufficient richness of trajectories are decoupled among robots (i.e. they are determined on a robot by robot basis). This is not the case for the parameter consensus control law. To analyze parameter convergence for this case, we must consider a concatenated vector consisting of all the robots' parameter errors

$$\tilde{A} = [\tilde{a}_1^T \quad \cdots \quad \tilde{a}_n^T]^T.$$

Also, define the block diagonal matrices $F = \text{diag}_{i=1}^n(\Gamma F_i)$, $\Lambda = \text{diag}_{i=1}^n(\Gamma\Lambda_i)$, and the generalized graph Laplacian matrix

$$\mathcal{L} = \begin{bmatrix} \Gamma(1)L(1,1)I_m & L(1,2)I_m & \cdots \\ \vdots & \ddots & \vdots \\ \cdots & L(n,n-1)I_m & \Gamma(n)L(n,n)I_m \end{bmatrix}.$$

The matrix $\mathcal{L}$ can be thought of as ΓL with each entry multiplied by the $m \times m$ identity matrix. Then the coupled dynamics of the parameters over the whole network can be written

$$\dot{\tilde{A}} = -(\gamma\Lambda + \zeta\mathcal{L})\tilde{A} - F\hat{A}, \tag{25}$$

with $\hat{A}$ defined in the obvious way. Notice the similarity in form between (23) and (25). Again this is a linear system in $\tilde{A}$ driven by a term that converges to zero. The eigenvalues of $\mathcal{L}$ are the same as those of ΓL, but each eigenvalue has multiplicity m. As for a typical graph Laplacian, $\mathcal{L}$ is positive semi-definite, and has m zero eigenvalues. Therefore, the trajectory of the network is sufficiently rich if $\gamma\Lambda + \zeta\mathcal{L}$ is positive definite. This is a less restrictive condition than for the basic controller. Furthermore, if parameter convergence takes place for the basic controller, then it will occur *more quickly* for the parameter consensus controller, since $\mathcal{L}$ always contributes a stabilizing affect. As before, convergence is presumably limited by the non-exponential driving term $F\hat{A}$, though this term can be made arbitrarily small by choosing Γ small, and γ and ζ correspondingly large.

6 Alternative Learning Laws

The adaptation law for parameter tuning (13) can be written more generally as

$$\dot{\hat{a}}_i = -F_i \hat{a}_i + f_i(p_i, V_i, \hat{a}_i, t), \tag{26}$$

where we have dropped the projection operation for clarity. There is considerable freedom in choosing the learning function $f_i(\cdot)$. We are constrained only by our ability to find a suitable Lyapunov-like function to accommodate Barbalat's lemma.

6.1 Gradient Algorithms

The form of $f_i(\cdot)$ chosen in Section 3 can be called a gradient algorithm, since

$$f_i = -\frac{\partial}{\partial \hat{a}_i}\left[\frac{1}{2}\gamma \int_0^t w(\tau)(\hat{\phi}_i - \phi_i)^2 \, d\tau\right]. \tag{27}$$

The algorithm follows the negative gradient of the Least Squares cost function, seeking a minimum.

Another possible learning algorithm is the simple gradient algorithm, given by

$$\begin{aligned} f_i &= -\frac{\partial}{\partial \hat{a}_i}\left[\frac{1}{2}\gamma w(\tau)(\hat{\phi}_i - \phi_i)^2\right] \\ &= -\gamma w(t) \mathcal{K}_i (\mathcal{K}_i^T \hat{a}_i - \phi_i). \end{aligned} \tag{28}$$

Using the same Lyapunov function as before, it can be verified that this learning law results in a *near-optimal* coverage configuration.

These two gradient algorithms can be combined to give

$$f_i = -\gamma\left[w(t)\mathcal{K}_i(\mathcal{K}_i^T \hat{a}_i - \phi_i) + (\Lambda_i \hat{a}_i - \lambda_i)\right], \tag{29}$$

which is, in fact, equivalent to the first algorithm with a weighting function $w_c(t,\tau) = \delta(t-\tau)w(t) + w(\tau)$, where $\delta(t-\tau)$ is the delta-Dirac function (we can make $w(\cdot)$ a function of t, and τ with minimal consequences to the proof). The same Lyapunov-like function can be used, such that the resulting time derivative is

$$\dot{\mathcal{V}} = -\sum_{i=1}^{n}\Big[\hat{M}_{V_i} k \|\hat{C}_{V_i} - p_i\|^2 + \tilde{a}_i^T k I_{\mathrm{proj}_i} \dot{\hat{a}}_{\mathrm{pre}_i} + k\gamma \tilde{a}_i^T\left[w(t)\mathcal{K}_i\mathcal{K}_i^T + \Lambda_i\right]\tilde{a}_i\Big],$$

leading to the same convergence claims as in Theorem 1 and Corollary 1.

6.2 Recursive Least Squares Algorithms

Another interesting possibility for a learning law is the Recursive Least Squares algorithm. This algorithm can be interpreted as continuously solving the Least Squares minimization problem recursively as new data is acquired. Let

$$J = 1/2 \int_0^t w(\tau)(\hat{\phi}_i - \phi_i)^2 \, d\tau$$

be the standard Least Squares cost function with a data weighting function $w(\tau)$. Then, taking the gradient with respect to $\hat{a}_i$ and setting to zero we find

$$\Lambda_i(t)\hat{a}_i = \lambda_i(t).$$

If the matrix $\Lambda_i(t)$ is full rank, we can pre-multiply both sides by its inverse to solve the Least Squares problem. However, we seek a recursive expression, so taking the time derivative we obtain

$$\dot{\hat{a}}_i = -P_i(t)w(t)\mathcal{K}_i(\mathcal{K}_i^T\hat{a}_i - \phi_i), \text{ where } P_i(t) = \Lambda_i(t)^{-1}.$$

Using an identity from vector calculus, P_i can be computed differentially by $\dot{P}_i = -P_i w(t)\mathcal{K}_i\mathcal{K}_i^T P_i$, but the initial conditions are ill defined. Instead, we must use some nonzero initial condition, P_{i0}, with the differential equation $\dot{P}_i = -P_i w(t)\mathcal{K}_i\mathcal{K}_i^T P_i$, to give the approximation

$$P_i = \Lambda_i^{-1} + P_{i0}. \tag{30}$$

The initial condition can be interpreted as the inverse covariance of our prior knowledge of the parameter values. We should choose this to be small if we have no idea of the ideal parameter values when setting initial conditions.

Before we can apply the Recursive Least Squares algorithm to our controller, there is one additional complication that must be dealt with. We can no longer use the same projection operator to prevent the singularity when $\hat{M}_{V_i} = 0$. However, it is possible to formulate a different stable controller that eliminates this singularity altogether. This formulation also has the advantage that it no longer requires Assumption 2. In essence, we simply multiply the existing controller by $\hat{M}_{V_i}$. In particular, we can use the controller

$$u_i = k(\hat{L}_{V_i} - \hat{M}_{V_i}p_i), \tag{31}$$

with the adaptation law

$$\dot{\hat{a}}_i = -P_i\left[\hat{M}_{V_i}F_i\hat{a}_i + w(t)\mathcal{K}_i(\mathcal{K}_i^T\hat{a}_i - \phi_i)\right] \tag{32}$$

to approximate the Recursive Least Squares algorithm. Asymptotic stability can be proven for this case by using the Lyapunov function

$$\mathcal{V} = \sum_{i=1}^{n}\left[\int_{V_i}\frac{1}{2}\|q - p_i\|^2\phi(q)\,dq + \frac{1}{2}\tilde{a}_i^T k P_i^{-1}\tilde{a}_i\right], \tag{33}$$

which leads to

$$\dot{\mathcal{V}} = -\sum_{i=1}^{n}\left[k\hat{M}_{V_i}^2\|\hat{C}_{V_i} - p_i\|^2 + \frac{1}{2}k\tilde{a}_i^T\left[w(t)\mathcal{K}_i\mathcal{K}_i^T\right]\tilde{a}_i\right],$$

Note that the only difference in the Lyapunov function is that Γ has been replaced with the time-varying quantity P_i.

We can also formulate a learning algorithm analogous to the combined gradient algorithm (29). This algorithm uses the learning law

$$\dot{\hat{a}}_i = -P_i\left(\hat{M}_{V_i}F_i\hat{a}_i + \frac{1}{2}w(t)\mathcal{K}_i(\mathcal{K}_i^T\hat{a}_i - \phi_i) + (\Lambda_i\hat{a}_i - \lambda_i)\right), \tag{34}$$

with Λ_i and λ_i defined as before. The same Lyapunov function can be used (33), resulting in

$$\dot{\mathcal{V}} = -\sum_{i=1}^{n}\left[k\hat{M}_{V_i}^2\|\hat{C}_{V_i} - p_i\|^2 + k\tilde{a}_i^T\Lambda_i\tilde{a}_i\right].$$

Interestingly, the integral terms (those involving Λ_i and λ_i) of the learning law in (34) have a gradient interpretation. Taking just those terms we have

$$\begin{aligned} f_i &= -P_i(\Lambda_i\hat{a}_i - \lambda_i) \\ &= -\tilde{a}_i + P_{i0}\Lambda_i\tilde{a}_i \\ &= -\frac{\partial}{\partial\hat{a}_i}\left(\frac{1}{2}\tilde{a}_i^T\tilde{a}_i\right) + P_{i0}\Lambda_i\tilde{a}_i, \end{aligned} \tag{35}$$

so the law approximates the gradient of the squared parameter error. The last term on the right hand side arises from the mismatch in initial conditions between P_i and Λ_i.

The combination of Least Squares and gradient learning apparent in this law is quite similar to the Composite Adaptation described in [24, 26]. In fact, if one identifies the prediction error as $\mathcal{K}_i^T \hat{a}_i - \phi_i$ and the tracking error as $\Lambda_i - \lambda_i \phi_i$ we have composite adaptation (except, of course, for the term containing F_i, which is required for the stability proof).

Unfortunately, it is found that the equations resulting from the Least Squares formulation are difficult to solve numerically, often causing robots to jump outside of the area Q, which then corrupts the Voronoi calculation. Alleviating this problem is a matter of ongoing research.

6.3 Weighting Functions

The form of the function $w(\cdot)$ can be designed to encourage parameter convergence. One obvious choice is to make $w(\tau)$ a square wave, such that data is not incorporated into $\int_0^t w(\tau)\mathcal{K}_i\mathcal{K}_i^T \, d\tau$ after some fixed time. This can be generalized to an exponential decay, $w(\tau) = \exp(-\tau)$, or a decaying sigmoid $w(\tau) = 1/2(\mathrm{erf}(c-t)+1)$. Many other options exist.

One intuitive option for $w(\cdot)$ is $w(\tau) = \|\dot{p}_i\|^2$, since the rate at which new data is collected is directly dependent upon the rate of travel of the robot. This weighting, in a sense, normalizes the effects of the rate of travel so that all new data is incorporated with equal weighting. Likewise, when the robot comes to a stop, the value of $\phi(p_i)$ at the stopped position does not overwhelm the learning law. This seems to make good sense, but there is an analytical technicality: to ensure that Λ_i and λ_i remain bounded we have to prove that $\dot{p}_i \in L_2$. In practice, we can set $w(\tau) = \|\dot{p}_i\|^2$ up to some fixed time, after which it is zero.

We can also set $w(t, \tau) = \exp\{-(t-\tau)\}$, which turns the integrators Λ_i, P_i, and λ_i into first order systems. This essentially introduces a forgetting factor into the learning law which has the advantage of being able to track slowly varying sensory distributions. Forgetting factors can have other significant benefits such as improving parameter convergence rates and allowing the flexibility to reject certain frequencies of noise in the error signal. A thorough discussion of forgetting factors can be found in [24], Section 8.7.

7 Numerical Simulations

7.1 Practical Algorithm

A practical method for implementing the proposed control law on a network of robots is detailed in Algorithm 1. Notice that the control law in (10) and adaptation law in (14) both require the computation of integrals over V_i, thus robot i must be able to calculate continuously its Voronoi region. Several algorithms exist for computing V_i in a distributed fashion, for example those given in [6, 12].

Algorithm 1 Adaptive Coverage Control Algorithm

Require: Each robot can compute its Voronoi region
Require: $\phi(q)$ can be parameterized as in (5)
Require: $a(j)$ are lower bounded as in (6)
  Initialize Λ_i, λ_i to zero, and $\hat{a}_i(j)$ to $a_{\min}$
  loop
    Compute the robot's Voronoi region
    Compute $\hat{C}_{V_i}$ according to (7)
    Update $\hat{a}_i$ according to (14)
    Update Λ_i and λ_i according to (11)
    Apply control input $u_i = -k(\hat{C}_{V_i} - p_i)$
  end loop

Algorithm 1 is decentralized, fully distributed, and requires minimal communication between neighboring robots. It can be used on teams of large robots, on teams of small robots such as [13], or on mobile sensor network nodes with limited computation and storage capabilities such as the mobile Mica Motes described by [23].

7.2 Implementation

Simulations were carried out in a Matlab environment. The dynamics in (4) with the control law in (10), and the adaptation laws in (14) (with (13) for the basic controller and (20) for the consensus controller) for a group of $n = 20$ robots were modeled as a system of coupled differential equations. A fixed-time-step numerical solver was used to integrate the equations of motion of the group of robots. The area Q was taken to be the unit square. The sensory function, $\phi(q)$, was parameterized as a Gaussian network with nine Gaussians. In particular, for $\mathcal{K} = [\ \mathcal{K}(1) \quad \cdots \quad \mathcal{K}(9)\ ]^T$, each component, $\mathcal{K}(j)$, was implemented as

$$\mathcal{K}(j) = \frac{1}{\sigma_j\sqrt{2\pi}} e^{-\frac{(q-\mu_j)^2}{2\sigma_j^2}}, \tag{36}$$

where $\sigma_j = .18$. The unit square was divided into an even 3×3 grid and each μ_j was chosen so that one of the nine Gaussians was centered at the middle of each grid square. The parameters were chosen as $a = [100 \quad a_{\min} \quad \cdots \quad a_{\min} \quad 100]^T$, with $a_{\min} = .1$ so that only the lower left and upper right Gaussians contributed significantly to the value of $\phi(q)$, producing a bimodal distribution.

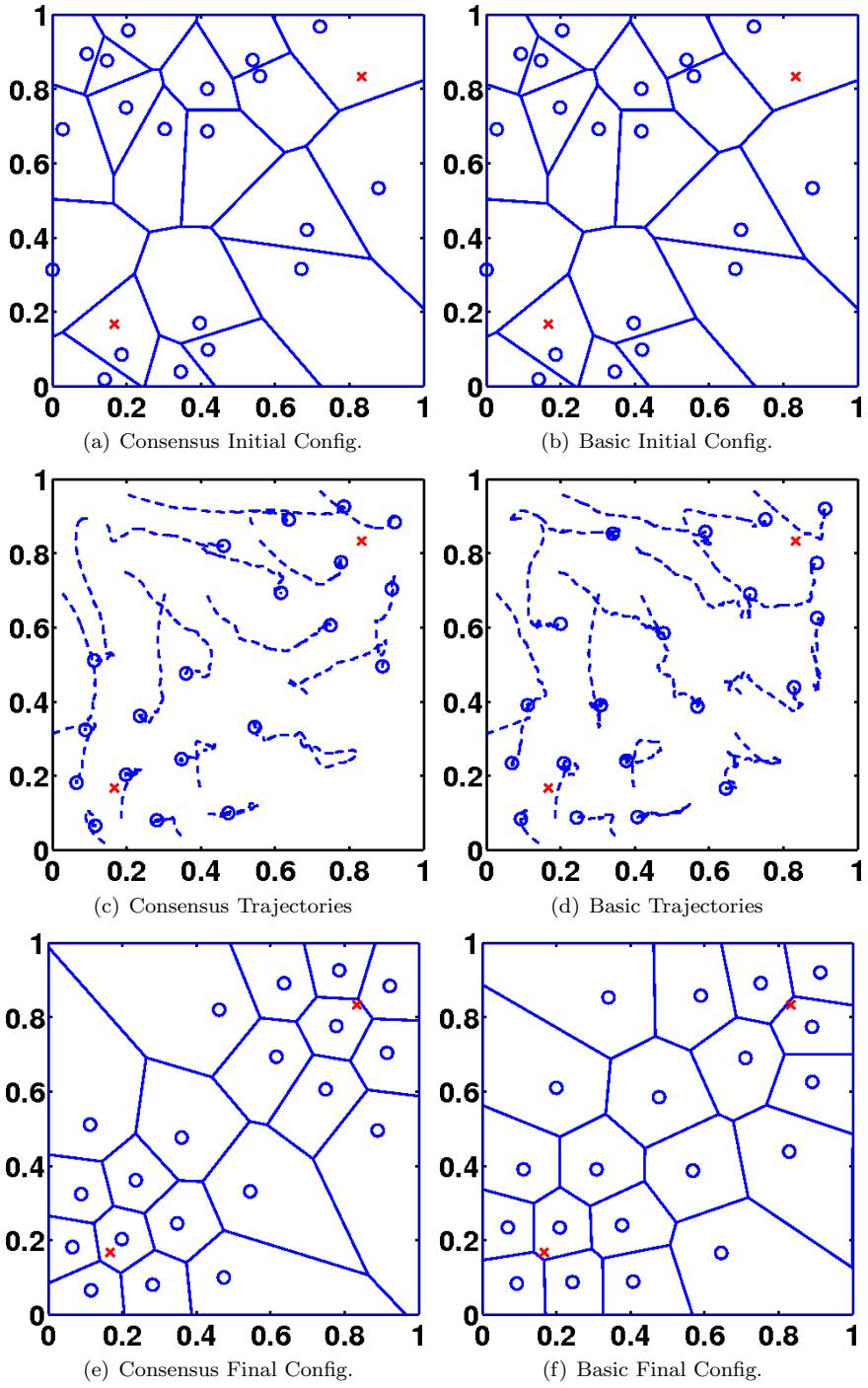
The robots in the network were started from random initial positions. Each robot used a copy of the Gaussian network described above for $\mathcal{K}(q)$. The estimated parameters $\hat{a}_i$ for each robot were started at a value of $a_{\min}$, and Λ_i and λ_i were each started at zero. The gains used by the robots were $k = 3$, $\Gamma = I_{10}$, $\gamma = 1000$ for the basic controller, and $\gamma = 100$ and $\zeta = 5$ for the consensus controller. In practice, the first integral term in the adaptive law (13) seems to have very little effect on the performance of the controller. Choosing Γ small and γ comparatively large puts more weight on the second term, which is responsible for integrating measurements of $\phi(p_i)$ into the parameters. The spatial integrals in (7) and (13) required for the control law were computed by discretizing each Voronoi region V_i into a 7×7 grid and summing contributions of the integrand over the grid. Voronoi regions were computed using a decentralized algorithm similar to the one in [6].

7.3 Simulation Results

Figure 4 shows the positions of the robots in the network over the course of a simulation run for the parameter consensus controller (left column) and the basic controller (right column). The centers of the two contributing Gaussian functions are marked with $\times$s. It is apparent from the final configurations that the consensus controller caused the robots to group more tightly around the Gaussian peaks than the basic controller. The somewhat jagged trajectories are caused by the discrete nature of the spatial integration procedure used to compute the control law.

We now investigate quantitative metrics to compare the performance of the consensus and basic controllers. Note that for all metrics shown, the convergence time scales are so different for the two controllers that a logarithmic scale had to be used on the time axis to display both curves on the same plot. The right of Fig. 5 shows that both controllers converge to a near-optimal configuration—one in which every robot is located at the estimated centroid of its Voronoi region, in accordance with Theorem 1. However, the true position error also converged to zero for the consensus controller, indicating that it achieved an optimal coverage configuration, as shown in the left of Fig. 5. The basic controller did not reach an optimal coverage configuration. Again, the somewhat jagged time history is a result of the discretized spatial integral computation over the Voronoi region.

Figure 6 demonstrates that a locally true sensory function approximation is achieved for each robot over $\Omega_i = \{p_i(\tau) \mid \tau \geq 0, w(\tau) > 0\}$, the set of points along the robot's trajectory with positive weighting. The plot shows the integral in (19) as a function of time averaged over all the robots in the network converging

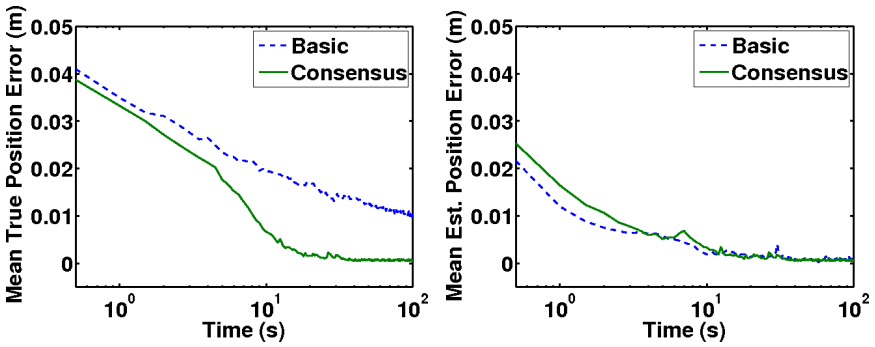


(a) Consensus Initial Config. (b) Basic Initial Config.

(c) Consensus Trajectories (d) Basic Trajectories

(e) Consensus Final Config. (f) Basic Final Config.

Figure 4: Simulation results for the parameter consensus controller are shown in the left column (4(a), 4(c), and 4(e)), and for the basic controller in the right column (4(b), 4(d), and 4(f)). The Gaussian centers of $\phi(q)$ are marked by the red x's.



(a) Mean True Position Error (b) Mean Estimated Position Error

Figure 5: The true position error, $\|C_{V_i} - p_i\|$, and the estimated position error, $\|\hat{C}_{V_i} - p_i\|$, averaged over all the robots in the network is shown for the network of 20 robots for both the basic and parameter consensus controllers. The true position error converges to zero only for the parameter consensus controller, 5(a). However, in accordance with Theorem 1, the estimated error converges to zero in both cases, 5(b). Note the logarithmic time scale.

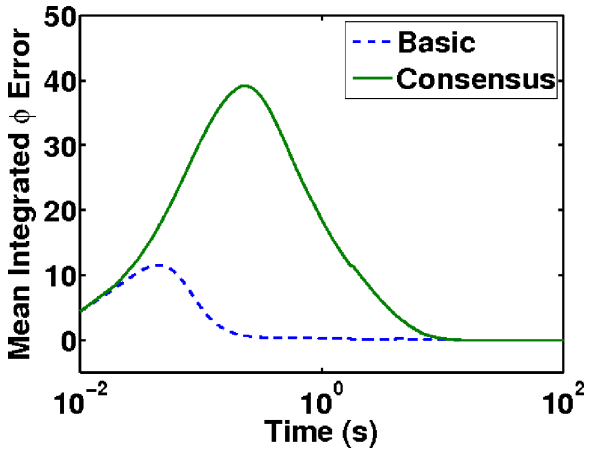


Figure 6: The integrated sensory function error, namely $\int_0^t w(\tau)(\mathcal{K}_i \tilde{a}_i)^2 \, d\tau$, averaged over all the robots is shown for the network of 20 robots for both the the basic and parameter consensus controllers. The plot demonstrates that each robot converges to a locally true function approximation over all points along its trajectory with positive weighting, $w(\tau) > 0$, as asserted in Theorem 1.

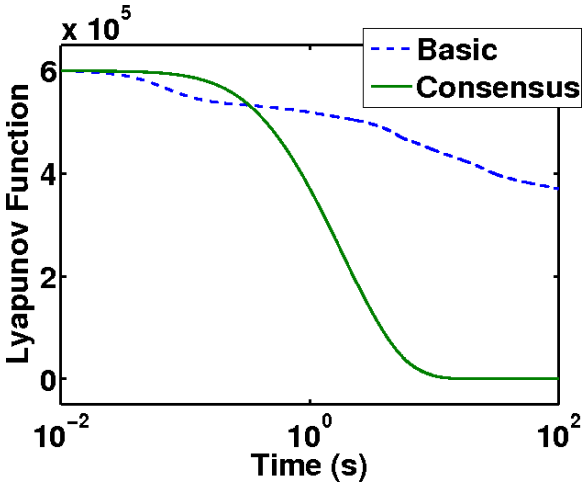


Figure 7: The Lyapunov function is shown for both the basic and parameter consensus controller. Notice that the parameter consensus controller results in a faster decrease and a lower final value of the function.

asymptotically to zero. This implies that the parameters adapt in such a way that the final estimate, $\lim_{t\to\infty} \hat{\phi}_i$, matches all previously measured values of $\phi(q)$.

Figure 7 shows that the consensus controller obtained a lower value of the Lyapunov function at a faster rate than the basic controller, indicating both a lower-cost configuration and a better function approximation. The final value for the consensus controller is not zero, as it appears to be in the plot, but is several orders of magnitude less than the final value for the basic controller.

Figure 8 shows the normed parameter error $\|\tilde{a}_i\|$ averaged over all of the robots. The parameter errors for the consensus controller all converged to zero, indicating that, in fact, sufficient richness conditions were achieved. This was also evidenced in Fig. 5(a). For the basic controller, on the other hand, the parameters did not converge to the true parameters.

Finally, the disagreement among the parameter values of robots is shown in Fig. 9. The larger the value in the plot, the more different the parameters are from one another. The parameters were initialized to $a_{\min}$ for all robots, so this value starts from zero in both cases. However, the consensus controller clearly causes the parameters to reach consensus, while for the basic controller the parameters do not converge to a common value.

8 Conclusion

In this work we described a decentralized control law for positioning a network of robots optimally in their environment. The controller used an adaptive control architecture to learn a parameterized model of the sensory distribution in the environment while seeking an optimal coverage configuration. The controller was proven to cause the robots to move to the estimated centroids of their Voronoi regions, while also causing their estimate of the sensory distribution to improve over time. Parameter coupling was introduced in the adaptation laws to increase parameter convergence rates and cause the robots' parameters to achieve a common final value. We also described several stable variations of the learning law for the controller. The control law was demonstrated in numerical simulations of a group of 20 robots sensing over an area with a bimodal Gaussian distribution of sensory information.

We expect that the techniques used in this paper will find broader application beyond the problem chosen here. It appears that consensus algorithms could be a fundamental and practical tool for enabling distributed learning, and has compelling parallels with distributed learning mechanisms in biological systems. We hope that the approach will yield fruitful combinations of adaptive control and decentralized control to produce

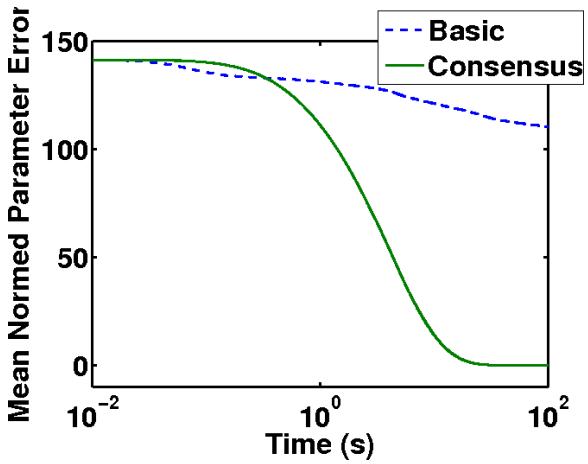


Figure 8: The normed parameter error $\|\tilde{a}_i\|$ averaged over all robots is shown for both the basic and parameter consensus controllers. Notice that the parameter error converges to zero with the consensus controller indicating that the robot trajectories were sufficiently rich.

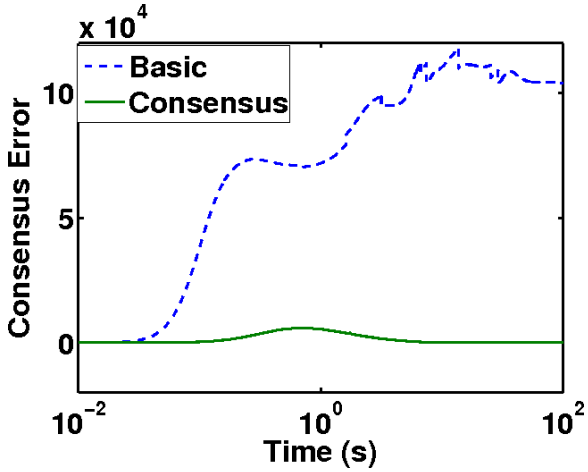


Figure 9: The quantity $\sum_{i=1}^{n} \tilde{a}_i^T \sum_{j \in \mathcal{N}_i} (\hat{a}_i - \hat{a}_j)$ is shown, representing a measure of the disagreement of parameters among robots. The disagreement converges to zero for the consensus controller, as asserted in Theorem 2, but does not converge for the basic controller.

engineered agents that can cooperate with one another while gathering information from their environment to proceed toward a common goal.

Appendix

This section contains tables of the symbols used in this work.

Table of symbols of primary importance:

Symbol	Description
Q	convex bounded area which the robots are to cover
q	an arbitrary point in Q
p_i	position of robot i
V_i	Voronoi region of robot i
$\phi(q)$	sensory function
$\phi_i(t)$	the value of the sensory function at a robot position, $\phi(p_i(t))$
$\hat{\phi}_i(q,t)$	robot i's approximation of $\phi(q)$
$\mathcal{K}(q)$	vector of basis functions for the sensory function, $\phi(q) = \mathcal{K}(q)^T a$
$\mathcal{K}_i(t)$	the value of the basis functions at a robot position, $\mathcal{K}(p_i(t))$
a	parameter vector of the sensory function, $\phi(q) = \mathcal{K}(q)^T a$
$\hat{a}_i(t)$	robot i's approximation of a
$\tilde{a}_i(t)$	parameter vector approximation error, $\hat{a}_i(t) - a$
M_{V_i}	mass of V_i
$\hat{M}_{V_i}(t)$	approximation of M_{V_i}
L_{V_i}	first mass moment of V_i
$\hat{L}_{V_i}(t)$	approximation of L_{V_i}
C_{V_i}	centroid of V_i
$\hat{C}_{V_i}(t)$	approximation of C_{V_i}
$\mathcal{H}(p_1, \ldots, p_n)$	locational cost function
u_i	control input
Λ_i	weighted integral of basis functions vector over robot trajectory
λ_i	weighted integral of sensory measurements over robot trajectory
$w(t)$	data weighting function
$\mathcal{V}$	Lyapunov function
$\mathcal{N}_i$	neighbor set of robot i
L	graph Laplacian of the robot network

Table of symbols of secondary importance:

Symbol	Description
n	number of robots in the network
N	number of dimensions of the space in which the robots exist
m	number of parameters
Ω_i	the set of points in the trajectory of $p_i(t)$ with positive weighting, $w(t) > 0$
k	positive control gain
F_i	term in Lyapunov proof resulting from imperfect sensory approximation
$\dot{\hat{a}}_{\mathrm{pre}_i}$	time derivative of the parameter approximation before projection
γ	adaptive gain for learning law
Γ	diagonal positive definite adaptive gain matrix
I_{proj_i}	matrix for implementing parameter projection
G	a graph
V	vertex set of a graph
v_i	a vertex in a graph
E	edge set of a graph
e_i	an edge in a graph
A	adjacency matrix of a graph
c	an arbitrary real constant
ζ	positive consensus gain
α_j	vector containing the jth parameter of each robot
T	some fixed time
$\lambda_{\min_i}$	the minimum eigenvalue of $\Lambda_i(T)$
$\tilde{A}$	vector containing the parameter errors of all robots
$\hat{A}$	vector containing the parameter estimates of all robots
F	block diagonal matrix with ΓF_i on each block
Λ	block diagonal matrix with $\Gamma\Lambda_i$ on each block
$\mathcal{L}$	generalized graph Laplacian for the network of robots
$f_i(p_i, V_i, \hat{a}_i, t)$	a general learning law
P_i	an approximation of Λ_i^{-1}
P_{i0}	the initial condition for P_i
σ_j	standard deviation of the jth Gaussian basis function
μ_j	mean of the jth Gaussian basis function

Acknowledgements

This work was supported in part by the MURI SWARMS project grant number W911NF-05-1-0219, and NSF grant numbers IIS-0513755, IIS-0426838, and CNS-0520305.

References

[1] Calin Belta and Vijay Kumar. Abstraction and control for groups of robots. *IEEE Transactions on Robotics and Automation*, 20(5):865–875, October 2004.

[2] D. P. Bertsekas and J. N. Tsitsiklis. Comments on "coordination of groups of mobile autonomous agents using nearest neighbor rules". *IEEE Transactions on Automatic Control*, 52(5):968–969, 2007.

[3] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice Hall, 1989.

[4] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. In *Proceedings of the Joint IEEE Conference on Decision and Control and European Control Conference*, Seville, Spain, December 2005.

[5] J. Cortés, S. Martínez, and F. Bullo. Spatially-distributed coverage optimization and control with limited-range interactions. *ESIAM: Control, Optimisation and Calculus of Variations*, 11:691–719, 2005.

[6] J. Cortés, S. Martínez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, April 2004.

[7] F. Cucker and S. Smale. Emergent behavior in flocks. *IEEE Transactions on Automatic Control*, 52(5):852–862, May 2007.

[8] Z. Drezner. *Facility Location: A Survey of Applications and Methods*. Springer Series in Operations Research. Springer-Verlag, New York, 1995.

[9] A. Ganguli, J. Cortés, and F. Bullo. Maximizing visibility in nonconvex polygons: nonsmooth analysis and gradient algorithm design. In *Proceedings of the American Control Conference*, pages 792–797, Portland, OR, June 2005.

[10] Jianghai Hu, Aria Prandini, and Shankar Sastry. Optimal coordinated motions of multiple agents moving on a plane. *SIAM Journal on Control and Optimization*, 42(2), 2003.

[11] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, June 2003.

[12] Qun Li and Daniela Rus. Navigation protocols in sensor networks. *ACM Transactions on Sensor Networks*, 1(1):3–35, Aug. 2005.

[13] James McLurkin. Stupid robot tricks: A behavior-based distributed algorithm library for programming swarms of robots. Master's thesis, MIT, 2004.

[14] Kumpati S. Narendra and Anuradha M. Annaswamy. *Stable Adaptive Systems*. Prentice-Hall, Englewood Cliffs, NJ, 1989.

[15] Reza Olfati-Saber and Richard R. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, September 2004.

[16] S. Salapaka, A. Khalak, and M. A. Dahleh. Constraints on locational optimization problems. In *Proceedings of Conference on Decision and Control*, Maui, Hawaii, USA, December 2003.

[17] R. Sanner and J.J.E. Slotine. Gaussian networks for direct adaptive control. *IEEE Transactions on Neural Networks*, 3(6), 1992.

[18] Shankar Sastry and Marc Bodson. *Adaptive control: stability, convergence, and robustness*. Prentice-Hall, Inc., Upper Saddle River, NJ, 1989.

[19] Mac Schwager, Francesco Bullo, David Skelly, and Daniela Rus. A ladybug exploration strategy for distributed adaptive coverage control. In *Proceedings of International Conference on Robotics an Automation*, Pasadena, CA, May 2008. Under review.

[20] Mac Schwager, James McLurkin, and Daniela Rus. Distributed coverage control with sensory feedback for networked robots. In *Proceedings of Robotics: Science and Systems*, Philadelphia, PA, August 2006.

[21] Mac Schwager, Jean-Jacques Slotine, and Daniela Rus. Decentralized, adaptive control for coverage with networked robots. In *Proceedings of International Conference on Robotics and Automation*, Rome, April 2007.

[22] Mac Schwager, Jean-Jacques Slotine, and Daniela Rus. Consensus learning for distributed coverage control. In *Proceedings of International Conference on Robotics and Automation*, Pasadena, CA, May 2008. Under review.

[23] Gabriel T. Sibley, Mohammad H. Rahimi, and Gaurav S. Sukhatme. Robomote: A tiny mobile robot platform for large-scale sensor networks. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington, DC, May 2002.

[24] Jean-Jacques E. Slotine and Weiping Li. *Applied Nonlinear Control*. Prentice-Hall, Upper Saddle River, NJ, 1991.

[25] J.J.E. Slotine and J.A. Coetsee. Adaptive sliding controller synthesis for nonlinear systems. *International Journal of Control*, 43(4), 1986.

[26] J.J.E. Slotine and W. Li. Composite adaptive control of robot manipulators. *Automatica*, 25(4), 1989.

[27] Herbert G. Tanner, George J. Pappas, and R. Vijay Kumar. Leader-to-formation stability. *IEEE Transactions on Robotics and Automation*, 20(3):443–455, June 2004.

[28] J. N. Tsitsiklis. *Problems in Decentralized Decision Making and Computation*. PhD thesis, Department of EECS, MIT, November 1984.

[29] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986.

[30] Tamas Vicsek, Andras Czirok, Eshel Ben-Jacob, Inon Cohen, and Ofer Shochet. Novel type of phase transition in a system of self-driven particles. *Physical Review Letters*, 75(6):1226–1229, August 1995.

[31] W. Wang and J. J. E. Slotine. On partial contraction analysis for coupled nonlinear oscillators. *Biological Cybernetics*, 23(1):38–53, December 2004.

[32] W. Wang and J. J. E. Slotine. A theoretical study of different leader roles in networks. *IEEE Transactions on Automatic Control*, 51(7):1156–1161, July 2006.