2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

Decentralized Estimation and Control of Graph Connectivity in Mobile Sensor Networks

Peng Yang Randy A. Freeman Geoffrey J. Gordon Kevin M. Lynch Siddhartha S. Srinivasa Rahul Sukthankar

Abstract— **The ability of a robot team to reconfigure itself is useful in many applications: for metamorphic robots to change shape, for swarm motion towards a goal, for biological systems to avoid predators, or for mobile buoys to clean up oil spills. In many situations, auxiliary constraints, such as connectivity between team members and limits on the maximum hop-count, must be satisfied during reconfiguration. In this paper, we show that both the estimation and control of the graph connectivity can be accomplished in a decentralized manner. We describe a decentralized estimation procedure that allows each agent to track the algebraic connectivity of a time-varying graph. Based on this estimator, we further propose a decentralized gradient controller for each agent to maintain global connectivity during motion.**

I. Introduction

In this paper we consider the problem of maintaining connectivity in mobile sensor networks. Here, each sensor plans its own motion and communicates with other sensors within a limited range.

Recent advances in wireless communications and electronics have enabled the development of low-cost sensor networks [1]. In mobile sensor network systems, current research areas include target tracking [14], [22], [11], formation and coverage control [2]–[4], [6], environmental monitoring [10], [16], [18], and several others. To accomplish these tasks, it is often desirable to maintain a connected information flow during agent motions. So far this connectivity maintenance problem in mobile sensor networks has been addressed using two different approaches, based on local and global connectivity.

The first approach focuses on devising decentralized controllers for each agent to maintain local connectivity. For discrete-time second-order agents, a feasible control space is computed in [13] for each agent to maintain all existing pairwise connections. In comparison, in [17] each agent tries to maintain its two-hop communication neighbors. The use of local connectivity measures allows each agent to compute a feasible motion controller with only local information, yet the resulting controls may be too restrictive to allow the agents to accomplish any other task.

The second approach in [20], [21] uses global connectivity measures such as *algebraic connectivity* [5]. Given a graph, *k-connectivity matrix*[1] is computed in [20]. To maintain graph connectivity, gradient controllers are designed such that each off-diagonal entry of the n-connectivity matrix remains positive over time (where n is the total number of agents). In comparison, the gradient controller designed in [21] uses the fact that a graph being connected is equivalent to the determinant of the *deflated Laplacian matrix* being positive. However, computing the k-connectivity matrix and the determinant of a deflated Laplacian matrix are both centralized procedures.

In this paper we approach the problem in a different way. The key component in our solution is a *decentralized power iteration* algorithm that estimates the eigenvector corresponding to the second smallest eigenvalue of a weighted Laplacian matrix. This eigenvector estimation procedure is then used to estimate the algebraic connectivity of a graph. We use this estimate in a decentralized controller that maintains the global connectivity of the graph over time.

The rest of the paper is organized as follows. We summarize the necessary graph theoretical background in the Preliminaries. In Section III, we first review the centralized discrete-time power iteration algorithm and then describe two continuous versions. In Section IV, we describe a decentralized continuous-time power iteration procedure and use it to estimate the connectivity of a graph. A controller to maintain connectivity is proposed in Section V. Finally in the Appendix we analyze the convergence of the centralized power iteration algorithm.

II. Preliminaries

Given n mobile agents, their communication graph G and edge set E, the *adjacency* matrix $A \in \mathbb{R}^{n\times n}$ is defined as

$$A_{ij} = \begin{cases} A_{ji} = f(p^i, p^j), & i \text{ and } j \text{ are connected} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

where $p^i, p^j \in \mathbb{R}^m$ are the positions of node i and j and we restrict $f(p^i, p^j) > 0$ to be a positive weighting function symmetric in its arguments. The degree of each node is $d_i = \sum_{j=1}^{n} A_{ij}$ or $d = A\mathbf{1}$ where $\mathbf{1}$ is a column vector of all ones. The *degree* matrix is defined as $D = \text{diag}(d)$, and the *weighted Laplacian* matrix of the graph is defined

This work was supported in part by NSF grants ECS-0601661 and IIS-0308224.

Peng Yang and Kevin Lynch are with the Department of Mechanical Engineering, Northwestern University, Evanston, IL 60208-3111, USA p-yang@northwestern.edu, kmlynch@northwestern.edu.

Randy Freeman is with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208-3118, USA freeman@ece.northwestern.edu.

Geoffrey Gordon is with the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA ggordon@cs.cmu.edu.

Siddhartha Srinivasa and Rahul Sukthankar are with Intel Research, Pittsburgh, PA 15213, USA siddh@cs.cmu.edu, rahuls@cs.cmu.edu.

[1]Given a graph's adjacency matrix A, its k-connectivity matrix is defined as $I + A + \cdots + A^k$ for $k \in \{1, \cdots, n\}$.

as $L = D - A$. The unweighted Laplacian matrix $\overline{L}$ can be treated as a special case here where $f(p^i, p^j) = 1$. The spectral properties of $\overline{L}$ have been shown to be critical in many multiagent applications, such as formation control [4], [6], consensus seeking [15] and direction alignment [8].

For the weighted Laplacian L, we make the common assumption that the weights A_{ij} are positive [12], in which case the spectral properties of $\overline{L}$ are similar to those of L. Specifically, we know

1) $L\mathbf{1} = 0$.
2) The spectrum of L satisfies $0 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, and $\lambda_2 > 0$ iff the graph is connected. (Here the definition of connectedness is the same as the unweighted case.) As in the unweighted case, we call λ_2 the *algebraic connectivity* of the graph.

III. Centralized Power Iteration

We want to build an algorithm to estimate the graph connectivity measure λ_2. To do this, we first estimate the corresponding eigenvector v_2 ($Lv_2 = \lambda_2 v_2$), which is then used to determine λ_2.

Throughout the rest of the paper, we use superscripts to index the agents and components of a vector, and subscripts to index eigenvalues, eigenvectors, and their estimates. For example, a Laplacian L has n eigenvalues $\lambda_1, \ldots, \lambda_n$ and n eigenvectors $v_1, \ldots, v_n$. The components of an eigenvector are $v_i = (v_i^1, \ldots, v_i^n)^T$. In addition, if $x \in \mathbb{R}^n$ is the network's estimate of the eigenvector v_2, then $x^i \in \mathbb{R}$ is the ith component of the estimate x, stored by agent i. We also write $\lambda_2^i \in \mathbb{R}$ for agent i's estimate of λ_2.

A. Discrete-time Power Iteration

Given a square matrix Q and its spectrum $\mu_1 > \mu_2 \cdots > \mu_n$, *power iteration* is an established iterative method to compute its leading eigenvalue μ_1 and its associated eigenvector [19]. Now assume instead of μ_1, we are interested in its second largest eigenvalue μ_2. If we already know μ_1 and its associated eigenvector v_1, we can estimate μ_2 by running the power iteration on the deflated matrix

$$\widetilde{Q} = Q - v_1 v_1^T. \tag{2}$$

Specifically this iterative power iteration procedure is carried out in three steps. For a random initial vector w,

1) Deflate Q to get $\widetilde{Q}$, as in (2).
2) Power Iteration: $x \leftarrow \widetilde{Q}w$.
3) Renormalization: $w \leftarrow \frac{x}{\|x\|}$. Then go to step 3.

This power iteration method converges linearly in the ratio μ_2/μ_3. Once it converges, w is the eigenvector corresponding to the second largest eigenvalue μ_2 of Q. In the case of repeated eigenvalues where $\mu_2 = \ldots = \mu_{k-1} > \mu_k$, the iteration converges in the ratio of μ_2/μ_k. If $\mu_2 = \ldots = \mu_n$, then any unit vector w is a solution.

B. Continuous-time Power Iteration

To get the second smallest eigenvalue λ_2, the basic idea here is to run the power iteration on a deflated matrix of $-L$ where $-\lambda_2$ becomes the leading eigenvalue. However, because $|\lambda_2| \leq |\lambda_n|$, running the discrete-time iteration will cause the state w to converge to the eigenvector v_n associated with λ_n. Here we construct a continuous-time power iteration to instead converge to v_2. Let $x = (x^1 \ldots x^n)^T \in \mathbb{R}^n$ be the estimate of the eigenvector v_2. The continuous-time algorithm has three parts:

1) Deflation: $\dot{x} = -\text{Ave}(\{x^i\})\mathbf{1}$
2) Power Iteration: $\dot{x} = -Lx$
3) Renormalization: $\dot{x} = -(\text{Ave}(\{(x^i)^2\}) - 1)x$

where the function $\text{Ave}(\{q^i\}) \triangleq (\sum_i q^i)/n$. (In the next section, we will substitute each $\text{Ave}(\cdot)$ operation with a decentralized consensus estimation process.) Step 1 drives x to the null space of $\mathbf{1}$. The power iteration in step 2 drives x toward the direction of v_2. Step 3 drives x toward the unit sphere.

Now we combine the three pieces in a linearly weighted fashion:

$$\dot{x} = -k_1\text{Ave}(\{x^i\})\mathbf{1} - k_2 Lx - k_3(\text{Ave}(\{(x^i)^2\}) - 1)x \tag{3}$$

To investigate the gain conditions of k_1, k_2 and k_3 that guarantee correct convergence of system (3), first we look at a simplified version of this system.

C. Modified Continuous-time Power Iteration

We begin by showing that $S_1 = \{x \,|\, \mathbf{1}^T x = 0\}$ is an invariant manifold for system (3). For any initial estimate $x \in S_1$, system (3) in forward time satisfies

$$\frac{d}{dt}(\mathbf{1}^T x) = -k_2 \mathbf{1}^T L x = 0. \tag{4}$$

We can force the estimate x into the manifold S_1 by doing a reinitialization step. Starting with an arbitrary initial condition $x(0)$, the system does one discrete-time power iteration step and resets its state to $x(0^+) = -Lx(0)$. It is easy to verify that $x(0^+) \in S_1$. With this new initial condition $x(0^+)$, system (3) evolves within S_1 and its dynamics simplifies to

$$\dot{x} = -k_2 Lx - k_3(\text{Ave}(\{(x^i)^2\}) - 1)x. \tag{5}$$

Based on the following theorem we know system (5) converges to a desired eigenvector v_2 (not necessarily normalized) for proper choices of k_2 and k_3.

Theorem 1: When the gain condition

$$k_3 > k_2 \lambda_n \tag{6}$$

is satisfied, for almost all initial conditions system (5) converges to an eigenvector $\tilde{v}_2$ corresponding to the eigenvalue $-\lambda_2$ of the weighted Laplacian matrix $-L$ satisfying $\|\tilde{v}_2\| = \sqrt{n(\frac{k_3 - k_2\lambda_2}{k_3})}$.

Proof: See the Appendix. ■

Each agent can satisfy this condition without knowing the graph topology. First we know

$$\sum_i \lambda_i = \text{trace}(L) = 2 \sum_{(i,j)\in E} A_{ij} < n(n-1) \max_{(i,j)\in E} A_{ij}.$$

Additionally, in our exponentially-decaying edge weighting scheme introduced in Section V, we have $A_{ij} \leq 1$. Therefore in this paper, each agent can satisfy (6) by choosing $k_3 > n(n-1)k_2$ (assuming n is known to every agent), although the simulation in Example 2 shows this is a rather conservative bound.

Based on Theorem 1, the heuristic gain conditions we use for tuning system (3) are $k_1 \gg k_3, k_3 I - k_2 L > 0$. Under this condition we know $k_1 \gg k_3, k_1 \gg k_2$. By perturbation analysis [9], system (3) first converges close to S_1, and then the evolution of its dynamics is similar to the dynamics of system (5).

Although both estimation procedures (3) and (5) apply to time-varying graphs, in practice procedure (3) is numerically superior. This is because (5) requires the system state to iterate within a reduced-dimensional manifold, and in our MATLAB implementation numerical error causes the state to deviate from the manifold. Numerical error accumulates and causes incorrect convergence.

Next we modify the continuous-time power iteration (3) and (5) so that these algorithms can be decentralized over the graph.

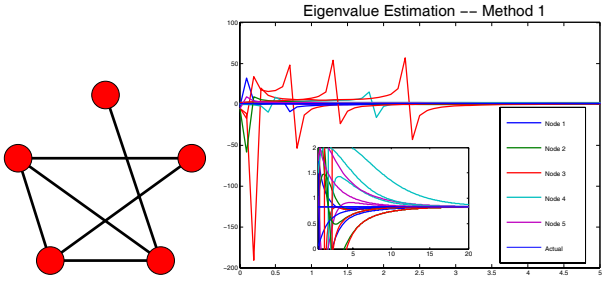
IV. Decentralized Power Iteration and Connectivity Estimation

Both (3) and (5) are centralized processes only because of the averaging operation Ave$(\cdot)$ in each algorithm. (Implementation of $\dot{x} = -Lx$ requires only local communication.) We can use the *PI average consensus estimator* [7] to decentralize this averaging operation. We need two consensus estimators to decentralize the dynamics (3), and only one for the dynamics (5). Consensus estimators allow n agents, each of which measures some time-varying scalar $\alpha^i(t)$, to compute an approximation of $\overline{\alpha}(t) = \frac{1}{n}\sum_i \alpha^i(t)$ using only local communication. The PI estimator has the form (see [7] for details):

$$\begin{aligned} \dot{y}^i =& \gamma(\alpha^i - y^i) - K_P \sum_{j\in\mathcal{N}^i} \left[y^i - y^j\right] \\ &+ K_I \sum_{j\in\mathcal{N}^i} \left[w^i - w^j\right] \end{aligned} \tag{7}$$

$$\dot{w}^i = - K_I \sum_{j\in\mathcal{N}^i} \left[y^i - y^j\right]. \tag{8}$$

Here y^i is the average estimate, $\gamma > 0$ is the rate new information replaces old information, $\mathcal{N}^i$ contains all one-hop neighbors of agent i in the communication network, and K_P, K_I are estimator gains. When the network is connected, the estimator error $e^i(t) = y^i(t) - \frac{1}{n}\sum_{i=1}^n \alpha^i(t)$ for each agent i approaches to an arbitrarily small ball around zero for slowly time-varying inputs $\alpha^i(t)$ [7].



(a) (b)

Fig. 1. (a) A five-node network with all weights equal to 1. (b) Eigenvalue estimation with the second method (equation (10)).

In the decentralized implementation of (3), agent i maintains a scalar x^i (which converges to the i-th component of the eigenvector $\tilde{v}_2$) and four consensus estimator states $\{y^{i,1}, w^{i,1}, y^{i,2}, w^{i,2}\}$ ($y^{i,1}$ and $y^{i,2}$ are agent i's estimates for Ave$(\{x^i\})$ and Ave$(\{(x^i)^2\})$ respectively) and receives from communication its neighbors' $\{x^j, y^{j,1}, w^{j,1}, y^{j,2}, w^{j,2}\}$ for all $j \in \mathcal{N}^i$. Noticing $-Lv_2 = \lambda_2 v_2$, agent i can estimate λ_2 through

$$\lambda_2^i = -\frac{\sum_{j\in\mathcal{N}^i} L_{ij} x^j}{x^i} \tag{9}$$

when $x^i \nrightarrow 0$.

There is a second way to estimate λ_2. From Theorem 1 we know $y^{i,2} \to \frac{k_3 - k_2\lambda_2}{k_3}$. Therefore agent i can compute its estimate as

$$\lambda_2^i = \frac{k_3}{k_2}(1 - y^{i,2}). \tag{10}$$

Note that no single agent maintains an estimate of the entire eigenvector v_2; instead, agent i maintains the single component x^i of the network's estimate x of v_2. This is sufficient to maintain an estimate λ_2^i of λ_2.

Example 1: We simulate these two eigenvalue estimation algorithms over the 5-node constant graph (Fig. 1), where the weights are set as $f(p^i, p^j) = 1$. The eigenvalue spectrum of its Laplacian matrix is $\{0, 0.83, 2, 69, 4.00, 4.48\}$.

The gains for the two PI average consensus estimators are $\gamma = 25, K_P = 50, K_I = 10$ and the gains for the eigenvector estimator are $k_1 = 6, k_2 = 1, k_3 = 20$. Fig. 1 and Fig. 2 show the estimated λ_2^i for each node i. With both methods each node's estimate converges to the correct eigenvalue $\lambda_2 = 0.83$. However, in the first method the trajectory of some node i will be nonsmooth when $\tilde{v}_2^i$ is close to zero. Therefore, method 2 is preferred in practice.

V. Control to Maintain Connectivity

We start by showing one additional property of λ_2.

Lemma 2: Given any positively weighted graph G, λ_2 is a nondecreasing function of each weight A_{ij}.

Remark 1: This lemma is easily demonstrated from the following alternative definition of λ_2:

$$\lambda_2 = \min_{x\perp 1, x\neq 0} \frac{x^T L x}{x^T x} = \min_{x\perp 1, x\neq 0} \frac{\sum_{(i,j)\in E} A_{ij}(x^i - x^j)^2}{x^T x}. \tag{11}$$

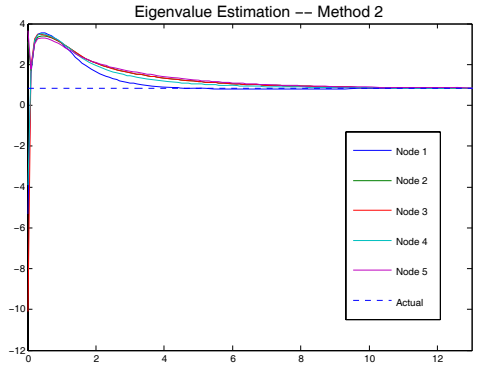


Fig. 2. Eigenvalue estimation with the first method (equation (9)). Nonsmooth behavior arises when individual $\tilde{v}_2^j$ is close to 0.

Based on this property, we can design a position-dependent weight function that is monotonically decreasing with respect to the inter-agent distance. Then we can design connectivity-maintaining motion controllers by bringing agents closer to each other.

If the maximal inter-agent communication distance is r, one simple weighting choice is

$$A_{ij} = \begin{cases} e^{-\|x^i - x^j\|_2^2/2\sigma^2}, & \|x^i - x^j\|_2 \leq r \\ 0, & \text{otherwise.} \end{cases}$$

The weight decreases as the inter-agent distance gets larger. We choose the parameter σ to satisfy a threshold condition $e^{-r^2/2\sigma^2} = \epsilon$, with ϵ being a small predefined threshold.

We know $\lambda_2 > 0$ for connected graphs, and based on Lemma 2, λ_2 increases as the graph adds more links or as individual link weights increase as two agents come closer. We can design a gradient controller where each node moves to maximize λ_2, and this will in effect maintain the connectivity of a graph. The gradient controller in [21] was designed based on a similar idea. In that paper, each node moves to maximize the determinant of the deflated Laplacian matrix of a graph, in effect guaranteeing the algebraic connectivity μ_2 is bounded away from 0.

Next we derive the analytical form of the gradient controller for fully-actuated first-order agents. Given a normalized eigenvector $\hat{v}_2$ ($\|\hat{v}_2\| = 1$) of λ_2, the differential of λ_2 is

$$\begin{aligned} \mathbf{d}\lambda_2 &= \mathbf{d}(\hat{v}_2^T L \hat{v}_2) \\ &= \mathbf{d}\hat{v}_2^T L v_2 + \hat{v}_2^T \mathbf{d}L \hat{v}_2 + \hat{v}_2^T L \mathbf{d}\hat{v}_2. \end{aligned} \quad (12)$$

Because $L^T = L$, we know that

$$\hat{v}_2^T L \mathbf{d}\hat{v}_2 = \mathbf{d}\hat{v}_2^T L v_2 = \lambda_2 \mathbf{d}\hat{v}_2^T \hat{v}_2 = \frac{1}{2}\mathbf{d}(\hat{v}_2^T \hat{v}_2) = 0. \quad (13)$$

Based on (12) and (13), the gradient controller for agent k is

$$u^k = \dot{p}^k = \frac{\partial \lambda_2}{\partial p^k} = \hat{v}_2^T \frac{\partial L}{\partial p^k} \hat{v}_2. \quad (14)$$

Next we replace the $\hat{v}_2$ in (14) with the $\tilde{v}_2$ in Theorem 1, which scales the control effort but not its direction:

$$u^k = \tilde{v}_2^T \frac{\partial L}{\partial p^k} \tilde{v}_2 = \sum_{(i,j)\in E} \frac{\partial A_{ij}}{\partial p^k} (\tilde{v}_2^i - \tilde{v}_2^j)^2. \quad (15)$$

Since we have defined $A_{ij} = e^{-\|p^i - p^j\|_2^2/2\sigma^2}$, we can compute

$$\begin{aligned} \frac{\partial A_{ij}}{\partial p^i} &= -A_{ij}(p^i - p^j)/\sigma^2 \qquad i \neq j & (16) \\ \frac{\partial A_{ij}}{\partial p^j} &= A_{ij}(p^i - p^j)/\sigma^2 \qquad i \neq j & (17) \\ \frac{\partial A_{ii}}{\partial p^i} &= 0 & (18) \\ \frac{\partial A_{ij}}{\partial p^k} &= 0 \qquad k \neq i, j. & (19) \end{aligned}$$

Plugging (16)-(19) into (15), we get

$$\begin{aligned} u^k &= \sum_{j \text{ such that } (k,j)\in E} \frac{\partial A_{kj}}{\partial p^k} (\tilde{v}_2^k - \tilde{v}_2^j)^2 \\ &= \sum_{j \text{ such that } (k,j)\in E} -A_{kj}(\tilde{v}_2^k - \tilde{v}_2^j)^2 \frac{p^k - p^j}{\sigma^2}. \end{aligned} \quad (20)$$

Compared to the eigenvector estimators (9) and (10), the implementation of (20) requires agent k to additionally obtain its neighbors' positions $\{p^j, j \in \mathcal{N}^i\}$ through communication. Then agent k can approximate the desired $\tilde{v}_2^k, \tilde{v}_2^j$ with the estimates x^k, x^j:

$$u^k = \sum_{j \text{ such that } (k,j)\in E} -A_{kj}(x^k - x^j)^2 \frac{p^k - p^j}{\sigma^2}. \quad (21)$$

Example 2: We simulate the connectivity-maintaining algorithm over a randomly-generated 6-node network. The communication radius is $r = 20$ and we set the threshold $\epsilon = 0.01$. In this network, the 3 big nodes are leaders. They all follow the same sinusoidal motion model $\dot{p}_x^i(t) = -0.2$, $p_y^i(t) = 0.5\cos(p_x^i)$ with different initial configurations. The three small nodes run (21) to move along with the leaders and maintain graph connectivity.

The gains for the two average consensus estimators are $\gamma = 100, K_P = 50, K_I = 200$ and the gains for the eigenvector estimator are $k_1 = 18, k_2 = 3, k_3 = 60$. We choose the consensus and eigenvector estimator gains to approximately achieve a time-scale separation: The time constant of consensus estimation is significantly less than the time constant of eigenvector estimation, which is significantly less than the time constant of the motion controller. Fig. 3 shows four snapshots of these nodes during the motion and Fig. 4 shows the estimated λ_2^i of each node i during the motion. A complete video of the simulation is available at `http://lims.mech.northwestern.edu/projects/swarm/ConnectivityMain/Connectivity_Sin_Bigmag.wmv`.

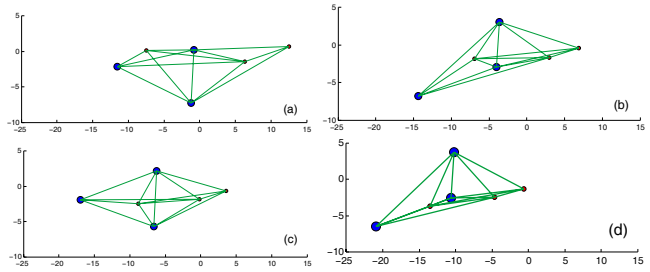


Fig. 3. Snapshots of the agents during motion: $(a)\, t = 0$; $(b)\, t = 14$; $(c)\, t = 27$; $(d)\, t = 47$.

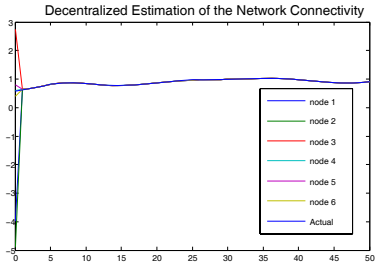


Fig. 4. Each agent's estimate of the the graph connectivity λ_2 over time (Method 2). All agents's estimates converge to the true algebraic connectivity of the graph within a few seconds.

VI. Future Work

We are interested in extending algorithm (3) in two possible ways. First, we would like to use certain local renormalization schemes other than the consensus scheme $\text{Ave}(\{(x^i)^2\})$ that can lessen the communication requirement and still guarantee correct convergence. Second, we would like to incorporate the information of the change of agent positions into either the eigenvector estimator or the consensus estimator, so that a rigorous small-gain type stability condition can be derived.

Appendix

We first show several stability properties of system (5) through three propositions, and at the end use these stability properties to prove Theorem 1.

Proposition 3: System (5) has an equilibrium point $x = 0$, and it is unstable when $-k_2 L + k_3 I > 0$.

Proof: It is easy to verify that $x = 0$ is an equilibrium state of system (5). Linearizing the system around the point $x = \tilde{x}$ we get

$$\dot{x} = [-k_2 L - k_3(\frac{\tilde{x}^T\tilde{x}}{n} - 1 + 2\frac{\tilde{x}\tilde{x}^T}{n})I]x \tag{22}$$

For the point $x = 0$, we can choose the gain condition

$$-k_2 L + k_3 I > 0 \tag{23}$$

to make it an unstable equilibrium point. ■

Now we proceed to investigate the other equilibrium points of system (5).

The weighted Laplacian matrix L is real symmetric, so it has an eigenvalue decomposition $L = T^T L^* T$ with $L^* = \text{diag}(0, \lambda_2, \cdots, \lambda_n)$. We denote $y = Tx$ and change the variable of system (5) to $y = (y^1 \cdots y^n)^T \in \mathbb{R}^n$:

$$\dot{y} = -k_2 L^* y - k_3(\text{Ave}(\{(y^i)^2\}) - 1)y. \tag{24}$$

Taking out the first row of equation (24), which is $\dot{y}^1 = 0$ (because $x \in \text{null}\{\mathbf{1}\}$), we get a reduced $(n-1)$-dimensional system

$$\begin{pmatrix} \dot{y}^2 \\ \vdots \\ \dot{y}^n \end{pmatrix} = -k_2 \widetilde{L}^* \begin{pmatrix} y^2 \\ \vdots \\ y^n \end{pmatrix} - k_3(\frac{y^T y}{n} - 1) \begin{pmatrix} y^2 \\ \vdots \\ y^n \end{pmatrix} \tag{25}$$

with $\widetilde{L}^* = \text{diag}\{\lambda_2, \cdots, \lambda_n\}$.

To simplify the analysis, in the following two propositions we only deal with the case when the eigenvalue spectrum of L has no repeated eigenvalues. The repeated eigenvalue case is discussed in the remark after Theorem 1.

Proposition 4: System (25) has $n-1$ distinct equilibrium points $\{y_i \mid 2 \le i \le n\}$ where y_i is

$$y_i^j = \begin{cases} 0, & 2 \le j \le n, j \ne i \\ \pm\sqrt{n(\frac{k_3 - k_2\lambda_i}{k_3})}, & j = i \end{cases} \tag{26}$$

Of all the $n-1$ equilibriums, only y_2 is stable when the gain condition (23) is satisfied.

Proof: For any nonzero equilibrium point of system (25), it has to be that $(y^2 \cdots y^n)^T$ is an eigenvector of the matrix $\widetilde{L}^*$ with an associated eigenvalue $\frac{k_3}{k_2}(\frac{y^T y}{n} - 1)$. There are $n-1$ distinct equilibrium points $\{y_i \mid 2 \le i \le n\}$ and for each equilibrium y_i

$$y_i^j = \begin{cases} 0, & 2 \le j \le n, j \ne i \\ \pm\sqrt{n(\frac{k_3 - k_2\lambda_i}{k_3})} & j = i. \end{cases} \tag{27}$$

The linearized model of system (25) is

$$\dot{y} = [-k_2\widetilde{L}^* - k_3(\frac{y_0^T y_0}{n} - 1 + 2\frac{y_0 y_0^T}{n})I]y. \tag{28}$$

Then for the equilibrium point y_i, its eigenvalue spectrum $\{\mu_i^j \mid j = 2, \cdots, n\}$ is

$$\begin{cases} \mu_i^j = k_2(\lambda_i - \lambda_j), & j = 2, \cdots, n, j \ne i \\ \mu_i^i = -2(-k_2\lambda_i + k_3). \end{cases} \tag{29}$$

Because $0 < \lambda_2 \le \cdots \le \lambda_n$, y_i is unstable for any $i > 2$ (at least in some directions), and y_2 is stable when (23) is satisfied. ■

Next we show system (5) will not converge to any limit cycle. We denote $\|x\| = \sqrt{x^T x}$ as the Euclidean norm of the estimate.

Proposition 5: Given system (5) and the gain condition (6), for almost all initial conditions $y \in S_1/\{0\}$, y converges to one of the two equilibria of y_2.

Proof: Given $V_1 = x^T x$, we have

$$\begin{aligned} \dot{V}_1 &= 2x^T\dot{x} \\ &= 2x^T[-k_2 L - k_3(\frac{x^T x}{n} - 1)I]x \\ &= 2x^T T^T[-k_2 L^* - k_3(\frac{x^T x}{n} - 1)I]Tx. \end{aligned} \tag{30}$$

The first entry of Tx is 0 because $x \in S_1$. Now given $0 < \lambda_2 \leq \cdots \leq \lambda_n$, we know $\dot{V}_1 < 0$ if

$$-k_2\lambda_2 - k_3(\frac{x^T x}{n} - 1) < 0 \tag{31}$$

and similarly $\dot{V}_1 > 0$ if

$$-k_2\lambda_n - k_3(\frac{x^T x}{n} - 1) > 0. \tag{32}$$

From (31) and (32) we conclude $\|x\|$ must converge into the bounded region $\left[\sqrt{n(\frac{k_3-k_2\lambda_n}{k_3})}, \sqrt{n(\frac{k_3-k_2\lambda_2}{k_3})}\right]$. Now assume there exists a limit cycle on which $\|x\|$ is bounded away from the upper bound $\sqrt{n(\frac{k_3-k_2\lambda_2}{k_3})}$ by $\epsilon > 0$. Then the first line in equation (25) gives

$$\dot{y^2} = (-k_2\lambda_2 - k_3(\frac{y^T y}{n} - 1))y^2 > \epsilon y^2. \tag{33}$$

Therefore

$$\|x\| = \|y\| \geq \|y^2\| \to \infty \tag{34}$$

as long as system (25) doesn't start from the measure zero manifold $y^2 = 0$ (all the $n-2$ unstable equilibrium points in Theorem 4 are on this plane). This contradicts the finite upper bound of $\|x\|$ and therefore each trajectory must converge to the sphere

$$\left\{ y | \, \|y\| = \sqrt{n(\frac{k_3 - k_2\lambda_2}{k_3})} \right\} \tag{35}$$

On this sphere, the other y-coordinates have dynamics

$$\dot{y^i} = k_2(\lambda_2 - \lambda_i)y^i \tag{36}$$

for $i > 2$. In the case of distinct eigenvalues $y^i \to 0$, system (25) eventually converges to the stable equilibrium y_2. ■

Based on previous discussions, now we can prove Theorem 1.

Proof: When the spectrum of L has no repeated eigenvalues, the result follows from Proposition 3, 4 and 5. ■

Remark 2: In case of repeated eigenvalues $\lambda_2 = \cdots = \lambda_k < \lambda_{k+1}$, Theorem 1 still holds. In this case almost all trajectories still converge to the sphere in equation (35). Furthermore, $y^i \to 0$ for $i > k$ and system (25) converges to the k-dimensional manifold $\{y | \, \|y\| = \sqrt{n(\frac{k_3-k_2\lambda_2}{k_3})}, y^i = 0, \forall i > k\}$ and it can be verified every point on that manifold is an equilibrium. The results are also verified through simulations.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communication Magazine*, 40(8):102–114, 2002.

[2] C. Belta and V. Kumar. Abstraction and control for groups of robots. *IEEE Transactions on Robotics*, 20(5):865–875, Oct. 2004.

[3] 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, 2004.

[4] J. A. Fax and R. M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, Sep 2004.

[5] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(98):298–305, 1973.

[6] R. A. Freeman, P. Yang, and K. M. Lynch. Distributed estimation and control of swarm formation statistics. In *American Control Conference*, 2006.

[7] R. A. Freeman, P. Yang, and K. M. Lynch. Stability and convergence properties of dynamic consensus estimators. In *IEEE International Conference on Decision and Control*, 2006.

[8] 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, Jun 2003.

[9] H. K. Khalil. *Nonlinear Systems*. Prentice Hall, Upper Saddle River, New Jersey, third edition, 2002.

[10] N. E. Leonard, D. Paley, F. Lekien, R. Sepulchre, D. Fratantoni, and R. Davis. Collective motion, sensor networks, and ocean sampling. In *Proceedings of the IEEE Special Issue on Networked Control Systems*, volume 95, pages 48–74, Jan 2007.

[11] S. Martínez and F. Bullo. Optimal sensor placement and motion coordination for target tracking. *Automatica*, 42:661–668, Apr 2006.

[12] B. Mohar. The Laplacian spectrum of graphs. *Graph Theory, Combinatorics, and Applications*, 2:871–898, 1991.

[13] G. Notarstefano, K. Savla, F. Bullo, and A. Jadbabaie. Maintaining limited-range connectivity among second-order agents. In *American Control Conference*, pages 2124–2129, 2006.

[14] S. Oh and S. Sastry. Tracking on a graph. In *Proc. of the Fourth International Conference on Information Processing in Sensor Networks (IPSN05)*, Los Angeles, CA, April 2005.

[15] R. Olfati-Saber and R. M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. Automat. Contr.*, 49(9):1520–1533, Sept. 2004.

[16] S. Simic and S. Sastry. Distributed environmental monitoring using random sensor networks. In *Proceedings of the 2nd International Workshop on Information Processing in Sensor Networks*, pages 582–592, Palo Alto, California, 2003.

[17] D. P. Spanos and R. M. Murray. Controlling connectivity of dynamic graphs. In *IEEE International Conference on Decision and Control*, 2005.

[18] S. Susca, S. Martínez, and F. Bullo. Monitoring environmental boundaries with a robotic sensor network. In *American Control Conference*, pages 2072–2077, 2006.

[19] L. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, Philadelphia, 1997.

[20] M. M. Zavlanos and G. J. Pappas. Controlling connectivity of dynamic graphs. In *IEEE Conference on Decision and Control*, pages 6388–6393, Dec 2005.

[21] M. M. Zavlanos and G. J. Pappas. Potential fields for maintaining connectivity of mobile networks. *IEEE Transactions on Robotics*, 23(4):812–816, Aug 2007.

[22] F. Zhao, J. Shin, and J. Reich. Information-driven dynamic sensor collaboration for tracking applications. *IEEE Signal Processing Magazine*, 19(2):61–72, March 2002.