Distributed motion constraints for algebraic connectivity of robotic networks

Michael Schuresko and Jorge Cortés

***Abstract*— This paper studies connectivity maintenance of robotic networks that communicate at discrete times and move in continuous space. We propose a distributed algorithm that allows the robots to decide whether a desired collective motion breaks connectivity. Our algorithm works under imperfect information caused by delays in communication and the robots' mobility. We analyze the correctness of our algorithm by formulating it as a game against a hypothetical adversary who chooses system states consistent with observed information. The technical approach combines tools from algebraic graph theory, linear algebra, nonsmooth analysis, and systems and control.**

I. Introduction

Network connectivity is a critical issue in cooperative robotics. In many applications, connectivity is needed in order to guarantee the successful completion of a desired coordination task. Examples include rendezvous at a point and distributed sensor fusion. In sensor fusion, distributed agreement protocols have convergence rates which depend on the degree of connectivity of the underlying communication network. Since connectivity is a global property, it is difficult to maintain it in a distributed manner. The objective of this paper is to develop a distributed approach to preserving network connectivity that allows for flexibility of individual robot motions and, at the same time, does not impose a heavy communication burden on the network operation.

Literature review: We classify previous work on connectivity of robotic networks into two main categories. The first deals with how to design the network motion to maximize some desired measure of connectivity under a given set of position constraints. In [1], convex optimization is used to solve this problem in the presence of convex constraints on the strength of inter-agent links. A solution with nonconvex constraints is presented in [9]. [5] provides a distributed algorithm when the strength of each link is a convex function of inter-robot distance. Potential fields are used in [19] to maximize algebraic connectivity. The second category deals with a measure of the connectivity of the interaction graph, a connectivity threshold, and some coordination task. In this category, algorithms are designed so that the robots' motions achieve the task subject to the value of the measure of connectivity never crossing the threshold. A solution to such a problem is proposed in [18]. This solution allows for a general range of agent motions, but is not distributed. A distributed solution that makes agents with second-order dynamics maintain a fixed set of edges appears in [12]. [14] presents a distributed solution which allow for varying set of edges to be preserved. Connectivity problems have been studied also in the context of formation control. In [16], connectivity-preserving motions between pairs of formations are generated. Control laws based on the Laplacian matrix of the interconnection graph are designed in [8] to solve formation control problems while preserving connectivity.

Statement of contributions: In this paper, our approach considers a measure of the connectivity of the interaction graph based on its Laplacian matrix. The Laplacian matrix of a graph is an analog to the Laplacian operator over the graph: its second smallest eigenvalue, λ_2, determines many connectivity properties of the graph. Given a pre-specified (arbitrary) lower threshold on λ_2, and a proposed instantaneous direction of physical motion, we set out to solve the following problem: how can the robots cooperatively decide which proposed motions can be taken without causing the measure of connectivity λ_2 to cross below the threshold? We propose a coordination algorithm which solves this problem under imperfect information caused by delays in communication and the robots' mobility, and has the added advantage of allowing for nonconvex mappings from inter-robot distance to edge weights. We provide correctness guarantees for the algorithm and simulate it with the underlying algorithms of random motion and trajectory following.

Notation: Throughout the paper, $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, and $\mathbb{R}_{>0}$ denote the sets of real, non-negative real, and positive real numbers, respectively. $\mathbb{F}(S)$ is the collection of finite subsets of a set S. When providing pseudo-code, we use $a \leftarrow b$ to mean "a is assigned a value of b." $\mathbb{R}^{m\times n}$ is the set of $m \times n$ matrices, and $\mathrm{Sym}(n)$ is the set of symmetric $n \times n$ matrices. The *Frobenius inner product* of $A, B \in \mathbb{R}^{m\times n}$ is

$$A \bullet B = \sum_{i=1}^{m}\sum_{j=1}^{n} A_{i,j}B_{i,j}.$$

For convenience, we introduce the "vectorization" $\mathrm{vec} : \mathbb{R}^{n\times n} \mapsto \mathbb{R}^{n^2}$ of a matrix defined by $\mathrm{vec}(M)_{in+j} = M_{i,j}$. Note that $(\mathrm{vec}(A))^T\mathrm{vec}(B) = A \bullet B$. Finally, we denote $\mathbf{1} = (1, \ldots, 1)^T \in \mathbb{R}^n$ and $\mathbf{0} = (0, \ldots, 0)^T \in \mathbb{R}^n$.

II. Preliminaries

This section presents preliminary notions on algebraic graph theory, proximity graphs, and nonsmooth analysis.

A. The graph Laplacian and its spectrum

We deal with undirected graphs. An undirected graph $G = (V, \mathcal{E})$ consists of a vertex set V and an edge set $\mathcal{E} \subset V \times V$ of unordered pairs of vertexes, i.e., $(i, j) \in \mathcal{E}$ implies that $(j, i) \in \mathcal{E}$. A weighted graph is a graph where each edge $(i, j) \in \mathcal{E}$ has an associated weight $w_{i,j} \in \mathbb{R}_{\geq 0}$. For a weighted graph $G = (V, \mathcal{E})$, the (weighted) adjacency

Michael Schuresko is with the Department of Applied Mathematics and Statistics, University of California, Santa Cruz, mds@soe.ucsc.edu

Jorge Cortés is with the Department of Mechanical and Aerospace Engineering, University of California, San Diego, cortes@ucsd.edu

$A(G) \in \mathrm{Sym}(n)$ and the Laplacian $L(G) \in \mathrm{Sym}(n)$ are given by

$$A(G)_{i,j} = w_{i,j}, \quad L(G)_{i,j} = \begin{cases} \sum_{k \neq i} w_{i,k} & i = j, \\ -w_{i,j} & i \neq j. \end{cases}$$

When the specific graph is clear from the context, we simply use A and L. We denote by $\Lambda : \mathrm{Sym}(n) \to \mathrm{Sym}(n)$ the linear map that transforms an adjacency matrix A onto the Laplacian L:

$$\Lambda(A) = \mathrm{diag}(A\mathbf{1}) - A = L.$$

Properties of the Laplacian matrix include [6]: the vector $\mathbf{1} \in \mathbb{R}^n$ is an eigenvector with eigenvalue 0; $L(G)$ is positive semidefinite; and $\dim(\ker(L(G)))$ is equal to the number of connected components of G. An undirected graph is connected if and only if the second smallest eigenvalue of its Laplacian is greater than zero. Adding weight to an edge of a graph does not decrease any of the eigenvalues of its Laplacian [17].

B. Proximity graphs and proximity functions

We use proximity graphs as an abstraction of network connectivity among spatially distributed robots. A proximity graph is an association of a set of positions with a weighted graph. Let $\mathcal{P} = (p_1, \ldots, p_n) \in (\mathbb{R}^d)^n$ be a vector of n robot positions, with each robot in $\mathbb{R}^d$. Let $\mathbb{G}(n)$ be the set of weighted graphs whose vertex set is $\{\{1, \ldots, n\}\}$. Then, we have the following definition [7], [4].

Definition 2.1: A *proximity graph* $\mathcal{G} : (\mathbb{R}^d)^n \to \mathbb{G}(n)$ associates to $\mathcal{P} \in (\mathbb{R}^d)^n$ a graph with vertex set $\{1, \ldots, n\}$, edge set $\mathcal{E}_\mathcal{G}(\mathcal{P})$, where $\mathcal{E}_\mathcal{G} : (\mathbb{R}^d)^n \to \{1, \ldots, n\} \times \{1, \ldots, n\}$, and weights $w_{i,j} \in \mathbb{R}_{>0}$ for all $(i, j) \in \mathcal{E}_\mathcal{G}(x)$. A proximity graph must satisfy that $\mathcal{G}(p_{\sigma(1)}, \ldots, p_{\sigma(n)})$ is isomorphic to $\mathcal{G}(p_1, \ldots, p_n)$ for any n-permutation σ and $(p_1, \ldots, p_n) \in (\mathbb{R}^d)^n$. •

For a given proximity graph, we often use the associated *proximity function* $(\mathbb{R}^d)^n \to \mathrm{Sym}(n)$ that maps a tuple $\mathcal{P} \in (\mathbb{R}^d)^n$ to the adjacency matrix $A(\mathcal{G}(\mathcal{P})) \in \mathrm{Sym}(n)$. We are particularly interested in a class of proximity functions defined by $f(p_1, \ldots, p_n)_{i,j} = g_{\text{wgt}}(\|p_i - p_j\|)$, with $g_{\text{wgt}} : \mathbb{R}_{\geq 0} \to \mathbb{R}$. For this paper we consider the added restrictions that g_{wgt} is $\mathcal{C}^2$ and monotonically decreasing.

C. Elements of nonsmooth analysis

It is possible to define a notion of gradient for locally Lipschitz functions [3]. Let $f : \mathbb{R}^d \to \mathbb{R}$ be locally Lipschitz at $x \in \mathbb{R}^d$. For any $v \in \mathbb{R}^d$, the *generalized directional derivative of f at x in the direction v*, denoted $f^\circ(x; v)$, is

$$f^\circ(x; v) = \limsup_{y \to x, t \downarrow 0} \frac{f(y + tv) - f(y)}{t}.$$

The generalized directional derivative has the property of always being well-defined, whereas the one-sided directional derivative might not exist in some cases. The *generalized gradient of f* at $x \in X$, denoted $\partial f(x)$, is the subset

$$\partial f(x) = \{\xi \in X \mid f^\circ(x; v) \geq \xi^T v \text{ for all } v \text{ in } X\}.$$

If f is C^1 at x, then $\partial f(x) = \{\nabla f(x)\}$.

D. Nonsmooth analysis of algebraic connectivity

Here we specify our scalar measure of network connectivity. Denote the (not necessarily distinct) eigenvalues of $M \in \mathrm{Sym}(n)$ by $\lambda_1(M) \leq \lambda_2(M) \leq \cdots \leq \lambda_n(M)$. We denote by $f_{\lambda_i} : \mathrm{Sym}(n) \to \mathbb{R}$ the function that maps M to $\lambda_i(M)$. Given a proximity function $f : (\mathbb{R}^d)^n \to \mathrm{Sym}(n)$,

$$f_{i-\text{conn}} = f_{\lambda_i} \circ \Lambda \circ f : (\mathbb{R}^d)^n \to \mathbb{R}. \tag{1}$$

We refer to $f_{2-\text{conn}}$ as the *algebraic connectivity function.*

Next, we analyze the smoothness properties of the functions $f_{i-\text{conn}}$, for $i \in \{1, \ldots, n\}$. We are particularly interested in $f_{2-\text{conn}}$.

Lemma 2.2: For $i \in \{1, \ldots, n\}$,

- the function f_{λ_i} is globally Lipschitz with constant 1.
- for a locally Lipschitz proximity function f, the connectivity function $f_{i-\text{conn}}$ is also locally Lipschitz. •

The following result [10] specifies the gradient of f_{λ_i}.

Theorem 2.3: For $i \in \{1, \ldots, n\}$, the generalized directional derivative (in the direction $X \in \mathrm{Sym}(n)$) and the generalized gradient of f_{λ_i} at $M \in \mathrm{Sym}(n)$ are given by

$$f^\circ_{\lambda_i}(M; X) = \max_{\{v \in \mathbb{S}^n \mid Mv = \lambda_i v\}} vv^T \bullet X,$$

$$\partial f_{\lambda_i}(M) = \mathrm{co}_{\{v \in \mathbb{S}^n \mid Mv = \lambda_i v\}}\{vv^T\}. \quad \bullet$$

The next result is a consequence of (1) and the nonsmooth chain rule [3, Theorem 2.3.10].

Theorem 2.4: Given a continuously differentiable proximity function, $f : (\mathbb{R}^d)^n \to \mathrm{Sym}(n)$, we have at $\mathcal{P} \in (\mathbb{R}^d)^n$, and $L = \Lambda(f(\mathcal{P}))$,

$$\partial f_{i-\text{conn}}(\mathcal{P}) \subseteq (\mathrm{vec}(\partial f_{\lambda_i}(L)))^T (\nabla \mathrm{vec}(L)). \quad \bullet$$

III. Problem formulation

Here, we describe our robotic network model and state the problem we address. Each robot has fully actuated first-order dynamics, and operates under a continuous-time control law. At discrete time intervals, each robot communicates with its neighbors over some proximity graph and re-computes an internal state which is used by its control law. The reader is referred to [2], [11] for a more detailed presentation.

The problem we address here is that of deciding when a proposed motion can be made while maintaining connectivity of the robotic network. Each robot should be able to solve the following problem.

Problem 3.1: Consider robot i with a desired motion specified by the input u_i. Given bounded sets, $\{U_j\}_{j \in \{1,\ldots,n\} \setminus \{i\}}$, such that each agent's control input, u_j must belong to U_j, a time interval $[t_0, t_0 + \delta T]$, and $[\lambda_-, \lambda_+] \subset \mathbb{R}_{>0}$, Spectral Connectivity Decision Problem consists of providing a procedure which, for each network configuration $\mathcal{P}$, returns a value, $f_{\text{safe}} \in \mathbb{R}$ having $f_{\text{safe}} \geq 0$ only if the following hold for all $t \in [t_0, t_0 + \delta T]$ and all $u_j \in U_j$, $j \in \{1, \ldots, n\} \setminus i$,

- $f_{2-\text{conn}}(\mathcal{P}(t)) \notin [\lambda_-, \lambda_+]$, or
- $f^\circ_{2-\text{conn}}(\mathcal{P}(t); [\mathbf{0}, \ldots, u_i^T, \ldots, \mathbf{0}]^T) \geq 0$.

where $\mathcal{P}(t)$ is the network evolution starting from $\mathcal{P}$ under control $\{u_i\}_{i=1}^n$. •

We assume that, initially, the algebraic connectivity λ_2 is larger than λ_+. During the network evolution, λ_2 might

decrease below λ_+. Our algorithm should guarantee that in such case, λ_2 never crosses below λ_-.

One can regard the solution to this problem as a building block towards the solution of more complex problems involving connectivity. For instance, given a specific strategy which achieves a coordination task, one could envision the synthesis of a procedure that modifies the directions of motion specified by the strategy as little as possible while preserving network connectivity. For space reasons, we do not deal with this problem here, and instead refer to [15].

IV. Eigenvalue games

In this section we introduce the main components of our solution to the problem 3.1. In Section IV-A, we reformulate Spectral Connectivity Decision Problem as a game, termed Graph Picking Game, which can be played with out-of-date information on the state of the network and in Section IV-B we study the properties of its solutions. In Section IV-C we present a distributed procedure that allows network agents to decide whether an intended motion wins Graph Picking Game. A final component of our solution is a distributed information dissemination algorithm.

A. Graph Picking Game

We are interested in characterizing the rates of change of Laplacian matrices arising from instantaneous robot motions which solve Spectral Connectivity Decision Problem. To do this, we reformulate this problem as a game and study the properties of its solutions.

*Definition 4.1: (*Matrix Form of the Spectral Connectivity Decision Problem*):* Given bounds on each edge weight of a graph G, $A_{i,j} \leq w_{i,j} \leq B_{i,j}$, $(i,j) \in E$, find the set of matrices $\mathcal{X}$ in $\{M \in \text{Sym}(n) \,|\, M\mathbf{1} = \mathbf{0}\}$ such that for each $\mathcal{G}$ having $f_{\lambda_2}(L(G)) \in [\lambda_-, \lambda_+]$, one has $f^\circ_{\lambda_2}(L(G); M) \geq 0$ for $M \in \mathcal{X}$. •

First, let us introduce notation used to discuss bounded intervals on the space of graph Laplacian matrices. Let

$$\text{LAP}_\pm(n) = \{M \in \text{Sym}(n) \,|\, M\mathbf{1} = \mathbf{0}\},$$
$$\text{LAP}(n) = \{M \in \text{LAP}_\pm(n) \,|\, M_{i,j} \leq 0 \text{ for all } i \neq j\}.$$

Lemma 4.2: The directional derivative of any function whose range is $\text{LAP}(n)$ lies in $\text{LAP}_\pm(n)$. •

Consider the following partial order in $\text{LAP}_\pm(n)$. For $A, B \in \text{LAP}_\pm(n)$, we write $A <_{\text{LAP}} B$ iff $A_{i,j} > B_{i,j}$ for all $i \neq j \in \{1, \dots n\}$. Likewise, $A \leq_{\text{LAP}} B$ if and only if $A_{i,j} \geq B_{i,j}$ for all $i \neq j \in \{1, \dots n\}$. For $A \leq_{\text{LAP}} B$, we define

$$[A, B]_{\text{LAP}} = \{L \in \text{LAP}_\pm(n) \,|\, A \leq_{\text{LAP}} L \leq_{\text{LAP}} B\}.$$

Note that $A, B \in \text{LAP}(n)$ and $L \in [A, B]_{\text{LAP}}$ imply $L \in \text{LAP}(n)$. The following result provides more properties of the matrices in the interval $[A, B]_{\text{LAP}}$.

Lemma 4.3: Let $A, B \in \text{LAP}(n)$, $L \in [A, B]_{\text{LAP}}$. Then,

(i) $f_{\lambda_2}(L) \in [f_{\lambda_2}(A), f_{\lambda_2}(B)]$
(ii) $vv^T \bullet L \in [vv^T \bullet A, vv^T \bullet B]$ for $v \in \mathbb{R}^n$.

We can now express the Matrix Form of the Spectral Connectivity Decision Problem in Definition 4.1 as a game played against a graph-picking opponent.

*Definition 4.4 (*Graph Picking Game*):* Given $A, B \in \text{LAP}(n)$ with $A \leq_{\text{LAP}} B$, we pick $Y \in \text{LAP}_\pm(n)$. Our opponent then selects $L \in [A, B]_{\text{LAP}}$. We win if either of the following conditions hold

- $f_{\lambda_2}(L) \notin [\lambda_-, \lambda_+]$, or
- $f^\circ_{\lambda_2}(L; Y) \geq 0$. •

Our objective is to characterize the choices Y that ensure that Graph Picking Game is won. Any instantaneous rate of change of the Laplacian belongs to $\text{LAP}_\pm(n)$ by Lemma 4.2.

B. Bounds on matrices which win Graph Picking Game

A direction that a robot can take in physical space induces an instantaneous rate of change of the Laplacian matrix of the underlying communication graph of the network. Given out-of-date information on the state of the network, each robot can produce bounds on the actual Laplacian of the graph. In this section we answer the following question: given lower and upper bounds $A, B \in \text{LAP}(n)$ on the Laplacian matrix of the communication graph and a range of possible instantaneous rates of change of the Laplacian matrix due to a proposed physical motion, can we guarantee that the proposed motion will not decrease the second smallest eigenvalue of the graph Laplacian? We do this by answering the related question: given the information listed above, and a range of "unsafe" eigenvalues, $[\lambda_-, \lambda_+]$, can we guarantee the proposed motion will not decrease the second smallest eigenvalue of the Laplacian matrix whenever the said eigenvalue is outside of the range $[\lambda_-, \lambda_+]$?

More formally, we bound the union of all possible gradients of f_{λ_2} evaluated at $L \in [A, B]_{\text{LAP}}$. Let $L \in [A, B]_{\text{LAP}}$ such that $f_{\lambda_2}(L) \in [\lambda_-, \lambda_+]$. Note that for each $w \in \mathbb{R}^n$ such that $L \bullet (ww^T) = f_{\lambda_2}(L)$ (i.e., $ww^T \in \partial f_{\lambda_2}(L)$), we must have $A \bullet (ww^T) \leq \lambda_+$ as a consequence of Lemma 4.3.

We now proceed to bound the set of $w \in \mathbb{R}^n$ which satisfy $A \bullet (ww^T) \leq \lambda_+$ and thus have $ww^T \in \partial f_{\lambda_2}(L)$. Let $\{u_1, \dots, u_m\}$ be the m eigenvectors of A corresponding to eigenvalues $\lambda_j \leq \lambda_+$ and let $\{u_{m+1}, \dots, u_n\}$ be the $n - m$ eigenvectors of A corresponding to eigenvalues $\lambda_j > \lambda_+$. Given $\tilde{m} \geq m$, define

$$\epsilon_A(\tilde{m}) = \sqrt{\frac{\lambda_+ - \lambda_2(A)}{\lambda_{\tilde{m}+1}(A) - \lambda_2(A)}},$$
$$\mathbf{u}_{\text{span-}A}(\tilde{m}) = \text{span}\{u_1, \dots, u_{\tilde{m}}\},$$
$$U_A(\tilde{m}) = \{w \in \mathbb{S}^n \mid \text{there exists } u \in \mathbb{S}^n \cap \mathbf{u}_{\text{span-}A}(\tilde{m}) \text{ such that } w \in B(u, \epsilon_A(\tilde{m}))\}.$$

We pick $U_A(\tilde{m})$ to contain the w satisfying $ww^T \in \partial f_{\lambda_2}(L)$ for $\tilde{m} \geq m$. We show that this inclusion holds next.

Proposition 4.5: Let $A, B \in \text{LAP}(n)$, $L \in [A, B]_{\text{LAP}}$, $f_{\lambda_2}(L) \leq \lambda_+$, $w \in \mathbb{S}^n$, $\tilde{m} \geq m$. If $w \notin U_A(\tilde{m})$, then $ww^T \notin \partial f_{\lambda_2}(L)$. •

The bound induced by $U_A(\tilde{m})$ works for any $\tilde{m} \geq m$. Our idea is to check for all such $\tilde{m}$, in the hope of finding one which verifies that our proposed motion is allowable.

The following result is a consequence of Theorem 2.4.

Corollary 4.6: Any instantaneous change in robot positions, $\{u_i\}_{i \in \{1,\dots,n\}}$ which induces an instantaneous rate

of change of the Laplacian, $Y \in \text{LAP}_{\pm}(n)$, satisfying $Y \bullet (uu^T) \geq 0$ for all $u \in U_A(\tilde{m})$ for some $\tilde{m} \geq m$ satisfies $f^{\circ}_{2-\text{conn}}(\mathcal{P}; \{u_i\}_{i\in\{1,\ldots,n\}}) \geq 0$. •

Given some $\tilde{m} \geq m$, we can conclude from Proposition 4.5 that any M satisfying $M \bullet (ww^T) \geq 0$ for all $w \in U_A(\tilde{m})$ wins GRAPH PICKING GAME on $A, B, \lambda_-, \lambda_+$. To determine whether a given M satisfies this property, it is sufficient to find the vector $u \in \mathbf{u}_{\text{span-}A}(\tilde{m})$ which minimizes $M \bullet (uu^T)$ or equivalently $u^T M u$ (and then tack a fudge factor based on $\epsilon_A(\tilde{m})$ onto this minimum).

Let d be the dimension of $\mathbf{u}_{\text{span-}A}(\tilde{m})$. Let $M_{u(\tilde{m})} \in \mathbb{R}^{n\times\tilde{m}}$ be a matrix whose column vectors are an orthonormal basis of $\mathbf{u}_{\text{span-}A}(\tilde{m})$. Any vector in $\mathbf{u}_{\text{span-}A}(\tilde{m}) \cap \mathbb{S}^n$ can be expressed as $M_{u(\tilde{m})}x$ for some $x \in \mathbb{S}^{\tilde{m}}$ and any $x \in \mathbb{S}^{\tilde{m}}$ satisfies $M_{u(\tilde{m})}x \in \mathbf{u}_{\text{span-}A}(\tilde{m}) \cap \mathbb{S}^n$.

Proposition 4.7: Finding the vector $u \in \mathbf{u}_{\text{span-}A}(\tilde{m}) \cap \mathbb{S}^n$ which minimizes $M \bullet (uu^T)$ is equivalent to finding the vector $x \in \mathbb{S}^d$ which minimizes $x^T M^T_{u(\tilde{m})} M M_{u(\tilde{m})} x$. This corresonds to the smallest eigenvalue of $M^T_{u(\tilde{m})} M M_{u(\tilde{m})}$. •

The next results provides a sufficient criterion to check if a matrix is a winning solution to GRAPH PICKING GAME.

Proposition 4.8: $(1 - \epsilon_A(\tilde{m})^2)M \bullet (uu^T) + \epsilon_A(\tilde{m})^2 \min(\min(\text{eigs}(M)), 0) \geq 0$ for all $u \in \mathbf{u}_{\text{span-}A}(\tilde{m})$ only if $M \bullet (ww^T) \geq 0$ for all $w \in U_A(\tilde{m})$.

C. DIRECTION CHECKING ALGORITHM

We introduce DIRECTION CHECKING ALGORITHM in Table I. Given $A, B \in \text{LAP}(n)$, and a lower bound, $X \in \text{LAP}_{\pm}(n)$ of the candidate instantaneous rate of change of the Laplacian matrix, $Y \in \text{LAP}_{\pm}(n), Y \geq_{\text{LAP}} X$, the algorithm returns a value $S_{\text{check}} \geq 0$ if it can verify that any $Y \geq_{\text{LAP}} X$ wins GRAPH PICKING GAME on A and B, and returns $S_{\text{check}} < 0$ otherwise.

The following result shows that DIRECTION CHECKING ALGORITHM is successful in determining if we win GRAPH PICKING GAME.

Theorem 4.9: DIRECTION CHECKING ALGORITHM returns $S_{\text{check}} \geq 0$ only when each Y having $Y \geq_{\text{LAP}} X$ satisfies $Y \bullet M \geq 0$ for $M \in \partial f_{\lambda_2}(L)$ with $L \in [A, B]_{\text{LAP}}$.•

D. Information dissemination of robot positions

In order to execute DIRECTION CHECKING ALGORITHM, robots first need information about the past states of the network to come up with reasonable bounds on the Laplacian matrix. Before specifying the protocol to disseminate information about each node throughout the network, we first address what it means for each node to hold information which is consistent with the real world.

Definition 4.10 (Consistency of stored network information): Let $\mathcal{P}_{\text{truth}} \in \mathbb{R}^{n\times n}$ be the actual position of the robots at time t_{curr}, and let $v_{\max}$ be a bound on the maximum velocity of each individual robot. A tuple, $(\mathcal{P}, T, D)$, $\mathcal{P} \in \mathbb{R}^{d\times n}, T \in \mathbb{R}^n, D \in \mathbb{R}^{n\times n}$, is called CONSISTENT with $\mathcal{P}_{\text{truth}}$ at time t_{curr} if the following hold:

(i) For $i \in \{1, \ldots, n\}$, $\mathcal{P}_i \in B(\mathcal{P}_{\text{truth}\,i}, (t_{\text{curr}} - T_i)v_{\max})$.

(ii) For $i, j \in \{1, \ldots, n\} \times \{1, \ldots, n\}$, $\|\mathcal{P}_{\text{truth}\,i} - \mathcal{P}_{\text{truth}\,j}\| \in [D_{i,j} - v_{\max}(t_{\text{curr}} - T_i + t_{\text{curr}} - T_j), D_{i,j} + v_{\max}(t_{\text{curr}} - T_i + t_{\text{curr}} - T_j)]$. •

Name:	DIRECTION CHECKING ALGORITHM
Goal:	Let Y be the (unknown) instantaneous rate of change of the Laplacian matrix of the communication graph of a robotic network. Given X (known) such that $Y - X$ is known to be positive semidefinite, determine whether Y can be proved to win GRAPH PICKING GAME on A and B and eigenvalue bounds λ_- and λ_+
Inputs:	• Matrices $A, B \in \text{LAP}(n)$ • Eigenvalue bounds $\lambda_- \leq \lambda_+ \in \mathbb{R}$ • Lower bound, $X \in \text{LAP}_{\pm}(n)$, on candidate direction in matrix space, $Y \in \text{LAP}_{\pm}(n)$
Outputs:	$S_{\text{check}} \in \mathbb{R}$. $S_{\text{check}} \geq 0$ means each $Y \geq_{\text{LAP}} X$ wins GRAPH PICKING GAME on A, B and $[\lambda_-, \lambda_+]$

Let $\lambda_+ \leftarrow \min(\lambda_+, \lambda_2(B))$
Let $\lambda_- \leftarrow \max(\lambda_-, \lambda_2(A))$
if $\lambda_- > \lambda_+$ **then**
  return 0
end if
Let $\lambda_{\min} \leftarrow \min(\text{eigs}(X))$
Let $m_{\min} \leftarrow \min\{m \mid \lambda_m \in \text{eigs}(A),\ \lambda_m > \lambda_+\}$
Initialize $S_{\text{check}} \leftarrow -1$.
for all $\tilde{m} \in \{m_{\min} - 1, \ldots, n\}$ **do**
  if $\tilde{m} < n$ **then**
    Let $\epsilon_A(\tilde{m}) \leftarrow \sqrt{\frac{\lambda_+ - \lambda_2(A)}{\lambda_{\tilde{m}+1}(A) - \lambda_2(A)}}$ and $\mathbf{u}_{\text{span-}A}(\tilde{m}) \leftarrow \text{span}(u_j, j \in \{1, \ldots, m\})$
  else
    Let $\epsilon_A(\tilde{m}) \leftarrow 0$
  end if
  Let $d \leftarrow \dim(\mathbf{u}_{\text{span-}A}(\tilde{m}))$
  Let $M_{u(\tilde{m})} \in \mathbb{R}^{n\times\tilde{m}}$ whose columns are orthogonal basis of $\mathbf{u}_{\text{span-}A}(\tilde{m})$
  Let $S \leftarrow (1 - \epsilon_A(m)^2)\min(\text{eigs}(M^T_{u(\tilde{m})} X M_{u(\tilde{m})})) + \epsilon_A(\tilde{m})^2 \min(\lambda_{\min}, 0)$ */*Does current $\tilde{m}$ verify X is safe?*/*
  Let $S_{\text{check}} \leftarrow \max(S, S_{\text{check}})$ */*Does any $\tilde{m}$ checked so far verify X is safe?*/*
end for
return S_{check} */*Does any $\tilde{m}$ verify X is safe?*/*

TABLE I
DIRECTION CHECKING ALGORITHM.

In other words, a set of information is CONSISTENT with an actual state of the world, 1) if the position of each robot, i, is within the range it could have reached by traveling with speed $v_{\max}$ starting from $\mathcal{P}_i$ for time $t_{\text{curr}} - T_i$ and 2) $D_{i,j}$ stores the distance between $\mathcal{P}_i$ and $\mathcal{P}_j$.

We achieve the problem of providing each robot with consistent information via ALL-TO-ALL BROADCAST ALGORITHM: a randomized algorithm in which, for each $i, j \in \{1, \ldots, n\}$ there is some finite probability i will receive an update on j's position at any given round. A formal description of the algorithm can be found in [15]. A simple description follows.

Under ALL-TO-ALL BROADCAST ALGORITHM, robots store a position estimate and a timestamp for each other robot. At each round, each robot transmits to its neighbors:

- Its own position, its UID, and the current time as a timestamp.
- The position, UID and timestamp of a randomly selected robot.

For each timestamp and position it receives, it compares the timestamp with the stored one for the associated UID. If the

time is more recent, it replaces its entry for that UID. The timestamp allows us to bound the current position of any robot, given the robot's maximum velocity and the current time. Because this algorithm is randomized, we discuss its expected performance.

Theorem 4.11: For any robot j the following holds: the expectation, for a randomly selected robot, $i \in \{1, \ldots, n\}$ of $t_{\text{curr}} - T_j^{[i]}$ never exceeds $(n^2+1)\delta T$. Likewise, the expected maximum, over all $i \in \{1, \ldots, n\}$ of $t_{\text{curr}} - T_j^{[i]}$ never exceeds $n(n^2+1)\delta T$. •

Additionally the information stored by the network is consistent with the actual robot positions in the sense of Definition 4.10, as we state next.

Theorem 4.12: Assume each robot moves with velocity at most $v_{\max}$. At all times, each robot holds values of $T, \mathcal{P}, D$ which are CONSISTENT, in the sense of Definition 4.10, with the state of the network at time t_{curr}. •

V. MOTION TEST ALGORITHM

Here we synthesize a motion coordination algorithm that solves the SPECTRAL CONNECTIVITY DECISION PROBLEM. With the information provided by the ALL-TO-ALL BROADCAST ALGORITHM, the network can compute lower $A \in \text{LAP}(n)$ and upper $B \in \text{LAP}(n)$ bounds on the Laplacian matrix of the communication graph. An explicit algorithm that does this is proposed in [15]. Combining these ingredients with the DIRECTION CHECKING ALGORITHM to verify winning solutions to GRAPH PICKING GAME, we synthesize the MOTION TEST ALGORITHM presented in Table II. The next result shows that this algorithm returns a value of $f_{\text{safe}} \geq 0$ only if the instantaneous change in the Laplacian due to motion in direction v wins GRAPH PICKING GAME.

Theorem 5.1: Assuming that each robot moves with velocity at most $v_{\max}$, MOTION TEST ALGORITHM solves SPECTRAL CONNECTIVITY DECISION PROBLEM. •

The next result shows that solutions to SPECTRAL CONNECTIVITY DECISION PROBLEM keep the algebraic connectivity of the robotic network above the desired threshold.

Corollary 5.2: If each robot runs an algorithm which solves SPECTRAL CONNECTIVITY DECISION PROBLEM, and never takes an "unsafe" motion, then λ_2 never drops below λ_-. •

A. Analysis under perfect information

We wish to show that MOTION TEST ALGORITHM exhibits reasonable behavior as δT becomes small. To do so, we compare it to an idealized variant of MOTION TEST ALGORITHM under which each robot has perfect information.

We let IDEALIZED MOTION TEST ALGORITHM be the algorithm defined by executing MOTION TEST ALGORITHM in continuous time, with $\delta T = 0$, and with perfect information about the state of the network available to each robot. We expound on how this is an idealized variant of MOTION TEST ALGORITHM in the following result.

The next result shows that, as δT approaches zero, the behavior of MOTION TEST ALGORITHM approaches that of IDEALIZED MOTION TEST ALGORITHM.

Name:	MOTION TEST ALGORITHM
Goal:	Solve SPECTRAL CONNECTIVITY DECISION PROBLEM.
Inputs:	• Current time $t_{\text{curr}} \in \mathbb{R}$ • Maximum velocity of any robot, $v_{\max}$ • Maximum time between communication rounds, δT • Proposed direction of motion, v • Eigenvalue bounds $\lambda_- \leq \lambda_+ \in \mathbb{R}$
Persistent data:	• $T \in \mathbb{R}^n$, last recorded time information • $\mathcal{P} \in \mathbb{R}^{d \times n}$, last recorded position information • $D \in \mathbb{R}^{n \times n}$, inter-robot distances • $A, B \in \text{LAP}(n)$ • id $\in \{1, \ldots, n\}$, identifier of current robot
Outputs:	• $f_{\text{safe}} \in \mathbb{R}$ such that $f_{\text{safe}} \geq 0$ if, for any time $t \in [t_{\text{curr}}, t_{\text{curr}} + \delta T]$, the instantaneous change in the Laplacian matrix due to motion in the direction v wins GRAPH PICKING GAME at time t

1: Initialize $X_{\text{upper}} \leftarrow \mathbf{0}$
2: Initialize $X_{\text{lower}} \leftarrow \mathbf{0}$
3: **for all** $i \in \{1, \ldots, n\}$ **do**
4: $\quad X_{\text{lower}_{\text{id},i}} \leftarrow -\min_{p \in B(\mathcal{P}_i, v_{\max}(t - T_i + \delta T))} g'_{\text{wgt}}(p, \mathcal{P}_{\text{id}}; v, \mathbf{0})$ */*Compute bounds on direction matrix*/*
5: $\quad X_{\text{upper}_{\text{id},i}} \leftarrow -\max_{p \in B(\mathcal{P}_i, v_{\max}(t - T_i + \delta T))} g'_{\text{wgt}}(p, \mathcal{P}_{\text{id}}; v, \mathbf{0})$
6: $\quad X_{\text{upper}_{\text{id,id}}} \leftarrow X_{\text{upper}_{\text{id,id}}} - X_{\text{upper}_{\text{id},i}}$
7: $\quad X_{\text{lower}_{\text{id,id}}} \leftarrow X_{\text{lower}_{\text{id,id}}} - X_{\text{lower}_{\text{id},i}}$
8: **end for**
9: $\lambda_- \leftarrow \max(\lambda_-, \lambda_2(A))$
10: $\lambda_+ \leftarrow \min(\lambda_+, \lambda_2(B))$
11: **if** $\lambda_- \geq \lambda_+$ **then**
12: $\quad$ return 0 */*There are no possible matrices with eigenvalues in the disallowed range*/*
13: **end if**
14: $f_{\text{safe}} \leftarrow$ DIRECTION CHECKING ALGORITHM $(A, B, X_{\text{lower}}, \lambda_-, \lambda_+)$
15: return f_{safe}

TABLE II
MOTION TEST ALGORITHM.

Theorem 5.3: Assume g''_{wgt} is bounded and $g'_{\text{wgt}}(0) = 0$. Then, for any configuration and any proposed direction of motion v for robot j, permitted under IDEALIZED MOTION TEST ALGORITHM, there exists a time step, δT, such that when communication happens every δT time units, with high probability robot j is allowed to move in direction v. •

B. Simulations

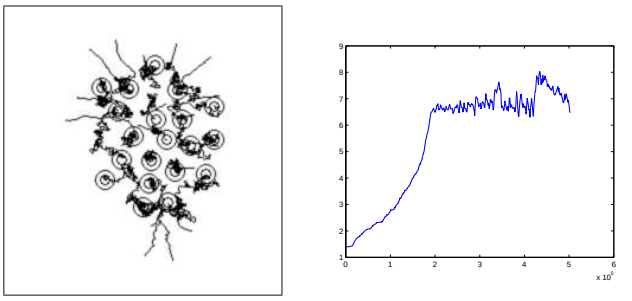
Here, we present simulations of an algorithm synthesized using MOTION TEST ALGORITHM. Assume we are given an underlying coordination algorithm that specifies a desired network motion at each configuration. The MOTION PROJECTION ALGORITHM evaluates MOTION TEST ALGORITHM on directions in a range about the one specified by the underlying law, and returns the closest direction among those for which MOTION TEST ALGORITHM returns "safe." Further details are presented in [15]. We have developed a custom Java-based simulation platform for robotic networks, which is available at [13]. This platform is a software implementation of the modeling framework for robotic networks proposed in [11].

We simulated the MOTION PROJECTION ALGORITHM with the r-disk graph for the actual communication network, and the nonconvex weight function

$$g_{\text{wgt}}(s) = \begin{cases} 1 - 3(\frac{s - r_{\text{mn}}}{r_{\text{mx}} - r_{\text{mn}}})^2 + 2(\frac{s - r_{\text{mn}}}{r_{\text{mx}} - r_{\text{mn}}})^3, & r_{\text{mn}} \leq s \leq r_{\text{mx}}, \\ 1 \text{ if } (s > r_{\text{mn}}) \text{ else } 0, & \text{otherwise}, \end{cases}$$

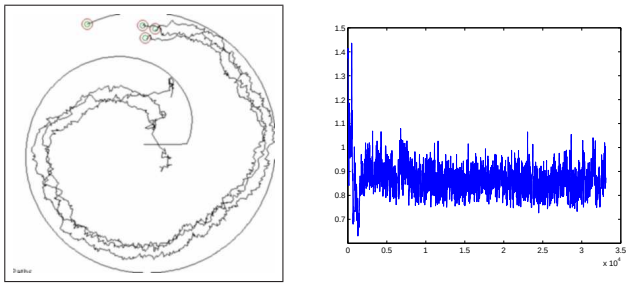
where $r_{\text{mx}}, r_{\text{mn}} \in \mathbb{R}$ satisfy $0 < r_{\text{mn}} < r_{\text{mx}} < r$. Note that the function g_{wgt} satisfies the conditions of Theorem 5.3.

We ran the simulation with two sets of underlying control laws. In the first simulation, we specify random desired motion for each agent, subject to a connectivity threshold. An example of this execution is shown in Fig. 1. In the second simulation, one robot attempts to follow a fixed trajectory while the others move randomly subject to the constraint of maintaining connectivity. Here we bound the angle each robot can deviate from its target direction by $\theta_{\max-i} = 0.2$. Fig. 2 shows a sample execution.



(a) (b)

Fig. 1. Execution of MOTION PROJECTION ALGORITHM with 18 robotic agents. The underlying control law for each agent is random motion. Plot (a) shows the paths taken by the robots and plot (b) shows the evolution of the algebraic connectivity. The threshold λ_+ is 6.



(a) (b)

Fig. 2. Execution of MOTION PROJECTION ALGORITHM with 4 robotic agents. The underlying control law corresponds to one leader following a fixed trajectory and the remaining agents moving randomly. Plot (a) shows the paths taken by the robots and plot (b) shows the evolution of the algebraic connectivity. The threshold λ_+ is .4 for the randomly-moving agents and .5 for the leader agent.

VI. CONCLUSIONS AND FUTURE WORK

We have studied the problem of connectivity maintenance in robotic networks. In our approach, the edge weights of the connectivity graph need not be convex functions of the inter-robot distances. We have proposed a distributed procedure to synthesize motion constraints on the individual robots so that the algebraic connectivity of the network is above a threshold. This algorithm works even though individual robots only have partial information about the network state due to communication delays and network mobility. We have shown that as the communication rate increases, the performance of the proposed algorithm approaches the ideal centralized solution to this problem. Future work will study the communication complexity of the proposed coordination algorithm. We are interested in calculating lower bounds on the communication complexity required to compute the gradient of $f_{2-\text{conn}}$. We also plan to study the relationship between the rate of information transmission and the rate of robot motion in terms of the number of robots and the exact value of $f_{2-\text{conn}}$. Finally, we plan to combine the proposed approach with algorithms for deployment and exploration.

ACKNOWLEDGMENTS

This research was supported in part by NSF CAREER Award ECS-0546871.

REFERENCES

[1] S. Boyd. Convex optimization of graph Laplacian eigenvalues. In *Proceedings of the International Congress of Mathematicians*, volume 3, 2006.

[2] F. Bullo, J. Cortés, and S. Martínez. *Distributed Control of Robotic Networks*. Princeton University Press, Princeton, NJ, June 2008. Manuscript preprint. Electronically available at http://www.coordinationbook.info.

[3] F. H. Clarke. *Optimization and Nonsmooth Analysis*. Canadian Mathematical Society Series of Monographs and Advanced Texts. John Wiley, 1983.

[4] J. Cortés, S. Martínez, and F. Bullo. Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *IEEE Transactions on Automatic Control*, 51(8):1289–1298, 2006.

[5] M. C. de Gennaro and A. Jadbabaie. Decentralized control of connectivity for multi-agent systems. In *IEEE Conf. on Decision and Control*, pages 3628–3633, San Diego, CA, December 2006.

[6] C. D. Godsil and G. F. Royle. *Algebraic Graph Theory*, volume 207 of *Graduate Texts in Mathematics*. Springer Verlag, New York, 2001.

[7] J. W. Jaromczyk and G. T. Toussaint. Relative neighborhood graphs and their relatives. *Proceedings of the IEEE*, 80(9):1502–1517, 1992.

[8] M. Ji and M. Egerstedt. Distributed control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics*, 23(4):693–703, 2007.

[9] Y. Kim and M. Mesbahi. On maximizing the second smallest eigenvalue of a state-dependent graph Laplacian. *IEEE Transactions on Automatic Control*, 51(1):116–120, 2006.

[10] A. S. Lewis. Nonsmooth analysis of eigenvalues. *Mathematical Programming*, 84:1–24, 1999.

[11] S. Martínez, F. Bullo, J. Cortés, and E. Frazzoli. On synchronous robotic networks – part i: Models, tasks, and complexity. *IEEE Transactions on Automatic Control*, 52(12):2199–2213, 2007.

[12] K. Savla, G. Notarstefano, and F. Bullo. Maintaining limited-range connectivity among second-order agents. *SIAM Journal on Control and Optimization*, 2007. Special issue on "Control and Optimization in Cooperative Networks." Submitted.

[13] M. D. Schuresko. CCLsim. a simulation environment for robotic networks, 2008. Electronically available at http://www.soe.ucsc.edu/~mds/cclsim.

[14] M. D. Schuresko and J. Cortés. Safe graph rearrangements for distributed connectivity of robotic networks. In *IEEE Conf. on Decision and Control*, pages 4602–4607, New Orleans, LA, 2007.

[15] M. D. Schuresko and J. Cortés. Distributed motion constraints for algebraic connectivity of robotic networks. In *Journal of Intelligent and Robotic Systems*, 2008. Special issue on "Special Issue on Unmanned Autonomous Vehicles." Submitted.

[16] D. P. Spanos and R. M. Murray. Motion planning with wireless network constraints. In *American Control Conference*, pages 87–92, Portland, OR, June 2005.

[17] C. W. Wu. Algebraic connectivity of directed graphs. *Linear and Multilinear Algebra*, 53(3):203–223, 2005.

[18] M. M. Zavlanos and G. J. Pappas. Controlling connectivity of dynamic graphs. In *IEEE Conf. on Decision and Control and European Control Conference*, pages 6388–6393, Seville, Spain, December 2005.

[19] M. M. Zavlanos and G. J. Pappas. Flocking while preserving network connectivity. In *IEEE Conf. on Decision and Control*, New Orleans, LA, 2007.