2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Connectivity Management in Mobile Robot Teams

Ethan Stump, Ali Jadbabaie, Vijay Kumar
GRASP Laboratory, University of Pennsylvania
Philadelphia, PA 19104, USA
{estump,jadbabai,kumar}@seas.upenn.edu

***Abstract*— We develop a framework for controlling a team of robots to maintain and improve a communication bridge between a stationary robot and an independently exploring robot in a walled environment. We make use of two metrics for characterizing the communication: the Fiedler value of the weighted Laplacian describing the communication interactions of all the robots in the system, and the k-connectivity matrix that expresses which robots can interact through k or less intermediary robots. At each step, we move in such a way as to improve the Fiedler value as much as possible while keeping the number of intermediary robots between the two robots of interest below a desired value. We demonstrate the use of this framework in a scenario where the hop-count constraint cannot be satisfied, but show that communication quality is maintained anyways.**

I. Introduction

When deploying robots to accomplish tasks in potentially unknown environments, one challenge to overcome is the lack of a global communication medium. Point-to-point wireless radio communication is available, but when dealing with robots spread over a large area, range and line-of-sight are both important considerations when determining whether a remote robot agent is able to receive commands and send back data. Although this challenge could be overcome by using better radios, an alternate strategy would be to deploy more robots to form a communication bridge between the user and the task-performing robot.

Control strategies for groups of locally communicating robots have been developed for systems with both first-order [1] and second-order [2] dynamics using models considering a pair of robots connected if their separation distance does not exceed some communication threshold. Both of these works create a graph with edges corresponding to the communication links in the network and then search for control inputs that preserve graph properties such as, in this case, graph connectedness; these works have created a strong precedent for the use of graph theoretic concepts when discussing control of networked robots. In [3], the restrictions on connectedness are loosened to k-connectedness to capture the fact that when looking at connectedness between agents k-hops away in the communication graph, there may be multiple redundant paths that preserve the connection and so some can be safely broken. Differential constraints on the control inputs are defined combinatorially to preserve the k-connectivity of the network. These papers were developed in an open-field setting with only range limitations being considered, and work with line-of-sight considerations in a cluttered environment is much rarer because it is trickier to prove stability and convergence. In [4], line-of-sight constraints were added in as an additional potential field on top of a navigation function, but convergence was only conjectured. For the related problem of maintaining line-of-sight to an unpredictably-moving target in a cluttered environment, there has been success using computational geometry techniques for both a single target/robot [5] and multiple targets/robots [6].

Returning to consideration of range-based connection models, the previously mentioned works deal with finding controls to strictly preserve graph properties but do not consider how to do so to form system configurations that are more robust or further from losing connectedness. One attempt to do so was covered in [7]. First the full system communication graph is reduced to just a spanning tree to reduce its complexity. Then the distances that agents would need to move in order to break their relative range constraints are computed, and controls are found to increase this margin. An alternate approach is given in [8] where the quality of the system connectedness is abstracted by computing the Fiedler value of the weighted graph Laplacian of the communication graph. By choosing controls to increase the Fiedler value, the connectedness is indirectly improved, and this was accomplished using a distributed subgradient algorithm. These ideas have much in common with [9], but in that work the Fiedler value maximization is cast as a semi-definite program. In both cases, one major complication is that the graph Laplacian depends nonlinearly on the controllable system state.

In this paper, we continue with the methodology of [8] in considering how to develop a framework for solving the problem of controlling a team of robots to enable and improve the system communications. However, we are interested in doing so in an environment with obstacles, and so we have incorporated the constraints of [3] and cast the resulting combination as a convex optimization problem solved at each time step. We go on demonstrate the use of this framework in a simulated environment with rooms in order to explore its effectiveness.

II. Problem and Theory

We consider a team of N point robots in a walled environment, where the position of robot i in the global frame is denoted by the vector $\boldsymbol{x}_i$ and the collection of all robot positions is $\boldsymbol{x} = \left[\boldsymbol{x}_1^T \cdots \boldsymbol{x}_N^T\right]^T$.

The task is simple: there is a stationary robot ($\boldsymbol{x}_\alpha$) acting as base station, and there is a mobile robot ($\boldsymbol{x}_\beta$) exploring the environment independently. The rest of robots in the team must move in such a way as to enable and improve communications between the mobile and the stationary robots. We split this task up by first exploring how communication quality is defined and improved, and then exploring how the communication link is enabled and maintained.

A. Connectivity Metric

Following the setup of [8], we define the communication quality for the network. The robot team is equipped with radio systems that allow point-to-point communications between individual robots. We model this link as range-dependent, with a quality that varies between 0 and 1. The quality of the link between robot i and robot j is set to be:

$$f_{ij}(\boldsymbol{x}) \triangleq \begin{cases} 1 & \|\boldsymbol{x}_i - \boldsymbol{x}_j\| < \rho \\ 0 & \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \geq R \\ \exp\left(\frac{-5(\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - \rho)}{R-\rho}\right) & otherwise \end{cases} \quad (1)$$

where R defines a cutoff distance where the signal becomes unusable and ρ defines a saturation distance where the communication between agents does not change as they get closer together (see Fig. 1). Such a model allows for there to be some notion of "better" communication between two agents so that we can ask questions regarding optimization later. An exponentially-decaying communication strength is a reasonable fit to measurements from physical systems [10].

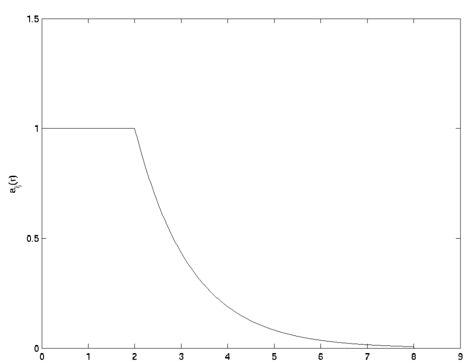

Fig. 1. Plot of Eq. 1, modeling the distance-based link quality between robots.

In stepping from treatment of quality of a single pair-wise connection to quality of the entire ensemble of pair-wise connections, we make use of some of the ideas of Graph Theory [11]. In particular, we construct a *weighted Laplacian*, $L(\boldsymbol{x})$, whose entries are:

$$[L(\boldsymbol{x})]_{ij} \triangleq \begin{cases} -f_{ij}(\boldsymbol{x}) & i \neq j \\ \sum_{k=1,k\neq i}^{N} f_{ik}(\boldsymbol{x}) & i = j \end{cases} \quad (2)$$

This is a Laplacian because it can be written as $D - A$, where A is the *adjacency matrix* whose entries are the pair-wise interactions between graph nodes, and D is a diagonal matrix consisting of the row-sums of A. It is denoted as weighted because the entries of A can take values other than zero or one, reflecting degrees of adjacency. Since L is symmetric, its eigenvalues are all real, and its special structure ensures that the smallest eigenvalue is zero, corresponding to the eigenvector of all ones, $\mathbf{1}$.

The second-smallest eigenvalue of L is known as the *Fiedler value*, denoted as λ_2, and it has an important interpretation for many of the problems modeled using a weighted Laplacian system [12]. In our system, we take λ_2 to signify an overall system connectedness: if it becomes zero, then the system has multiple connected components (a well known result from Graph Theory), and if it becomes higher, we interpret this as meaning that nodes are more tightly connected.

The function $\lambda_2(L)$ is a concave function of a Laplacian L, as clearly demonstrated by expressing it as:

$$\lambda_2(L) = \inf_{\boldsymbol{v}\in\mathbf{1}^{\perp}} \frac{\boldsymbol{v}^T L \boldsymbol{v}}{\boldsymbol{v}^T \boldsymbol{v}}$$

Concavity follows because this is the point-wise infimum of a family of linear functions [13]. Although this function is concave, it is not smooth and so does not have a gradient at all points. But notice that if λ_2 is distinct, i.e. the third-smallest eigenvalue, λ_3, is different from λ_2, then the only choice of $\boldsymbol{v}$ that completes the infimum is $\boldsymbol{v}_2$: the eigenvector corresponding to the second eigenvalue. When this is true, there is only one linear function active and so its derivative must be the derivative of the infimum as a whole. Therefore, as long as λ_2 is distinct, then we can state that:

$$\frac{\partial \lambda_2(L)}{\partial L} = \frac{\boldsymbol{v}_2 \boldsymbol{v}_2^T}{\boldsymbol{v}_2^T \boldsymbol{v}_2} \quad (3)$$

We do not know the significance of λ_2 being non-repeated. Even if this condition is broken, Eq. 3 still provides a *supergradient* belonging to the *superdifferential* [14], as shown in [8].

But while λ_2 may be concave with respect to L, L is a nonlinear function of $\boldsymbol{x}$. We can, however, use the chain rule to linearize λ_2:

$$\frac{\partial \lambda_2(L(\boldsymbol{x}))}{\partial \boldsymbol{x}_\alpha} = \left\langle \frac{\partial \lambda_2(L)}{\partial L}, \frac{\partial L}{\partial \boldsymbol{x}_\alpha} \right\rangle \quad (4)$$

where $\boldsymbol{x}_\alpha$ is a single element of $\boldsymbol{x}$ (rather than the subvector corresponding to a particular robot), and $\langle A, B\rangle \triangleq tr(A^T B)$, an inner product for the space of matrices. Note that $\frac{\partial L}{\partial \boldsymbol{x}_\alpha}$ will be sparse, with entries only on the diagonal and in indices corresponding to robots currently interacting with the robot being partially described by $\boldsymbol{x}_\alpha$.

As demonstrated in [8], the computation of $\boldsymbol{v}_2\boldsymbol{v}_2^T$ can be performed in a distributed way. Moreover, since the entries of $\frac{\partial L}{\partial \boldsymbol{x}_\alpha}$ are only non-zero when local interactions are occuring, then the computation of $\frac{\partial \lambda_2}{\partial \boldsymbol{x}_i}$, the sensitivity of λ_2 to the motion of robot i, can be computed locally to robot i without relying on a central computer.

Clearly, knowledge of this derivative gives local directions to move to improve or at least maintain connectivity of the network as measured by λ_2. But in practice, this direction is not enough because two of the nodes are not being directly controlled and the desired directions could indicate that the robots should move through walls in improve the communications. As the mobile node moves away or robots slide along walls, the network could possibly be forced to go through phases of lower connectivity and this simple gradient scheme will not ensure that the network maintains connectivity, necessitating a further addition to the system.

B. Connectivity Maintenance

Since part of the task is to maintain connectivity between the stationary and mobile nodes at all times, we need a means to derive constraints that will prevent the robot motions from breaking connectivity. For this, we make use of the ideas of *k-connectivity* and differential constraints to preserve it, as derived by [3].

We use a new adjacency matrix, $\hat{A}(\boldsymbol{x})$, whose entries no longer reflect degrees of adjacency as before. Instead, the entries $\hat{a}_{ij}$ are composed of sharp *sigmoid* functions, reflecting an on/off relationship where pairs of nodes either have communication or they do not but allowing for derivatives to be taken near the transition:

$$\hat{a}_{ij}(\boldsymbol{x}) \triangleq \hat{u}(R - \|\boldsymbol{x}_i - \boldsymbol{x}_j\|) \tag{5}$$

where R is the cutoff distance again, and $\hat{u}$ is the sigmoid:

$$\hat{u}(y) = \frac{1}{1 + e^{-\omega y}}$$

This function is shown with various values of ω in Fig. 2.

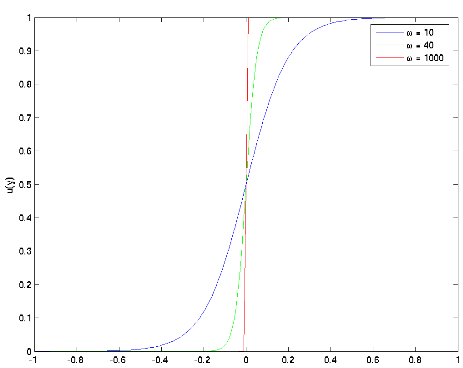

Fig. 2. Sigmoid function $\hat{u}(y)$ for various sharpness values ω.

The *k-connectivity matrix* is defined as:

$$C_k(\boldsymbol{x}) \triangleq I_N + \hat{A}(\boldsymbol{x}) + \hat{A}^2(\boldsymbol{x}) + \cdots + \hat{A}^k(\boldsymbol{x}) \tag{6}$$

and the $\{i, j\}$ entry can be interpreted as the number of communication paths of k-hops or less that connect robot i to robot j. As shown in [3], this matrix can be differentiated with respect to the robot state $\boldsymbol{x}$ to find $\nabla C_k(\boldsymbol{x})$ and complete the expression:

$$\dot{C}_k(\boldsymbol{x}) = (\nabla C_k \boldsymbol{x})(I_N \otimes \dot{\boldsymbol{x}})$$

where $\otimes$ denotes the Kronecker product (see kronecker reference). $\nabla C_k(\boldsymbol{x})$ is an $N \times 2N^2$ matrix, but the expression could be rearranged by stacking the elements of C_k into a vector:

$$[\dot{c}_{ij}(\boldsymbol{x})]_{ij} = (\nabla c_{ij}(\boldsymbol{x}))\dot{\boldsymbol{x}} \tag{7}$$

Note that because C_k is symmetric and we can ignore the diagonal (since we do not care about the node's connectivity to itself), we actually only have $\frac{n(n-1)}{2}$ derivatives to deal with. Each $\nabla c_{ij}(\boldsymbol{x})$ has dimension of $1 \times 2N$.

Although we now know how the k-connectivity between any two robots changes as the robots move, we are only interested in the connection between the stationary robot and the exploring robot, allowing us to use only the derivative of Eq. 7 that corresponds to this pair of indices (i.e. $\nabla c_{\alpha\beta}(\boldsymbol{x})$).

As in [3], we are only interested in whether $c_{\alpha\beta}$ is non-zero, and so pass $c_{\alpha\beta}$ through the sigmoid $\hat{u}$ and require that its change always be non-negative, ensuring that the robots of interest stay connected at all times:

$$\hat{u}'(c_{\alpha\beta}(\boldsymbol{x}))(\nabla c_{\alpha\beta}(\boldsymbol{x}))\dot{\boldsymbol{x}} \geq 0 \tag{8}$$

where $\hat{u}'$ is the derivative of the sigmoid with respect to its single argument.

The computation of $c_{\alpha\beta}(\boldsymbol{x})$ and $\nabla c_{\alpha\beta}(\boldsymbol{x})$ has not yet been made decentralized, although some distribution of this operation seems possible.

C. Solving the Task

Now we have at hand the two pieces we need to solve the task of improving and maintaining communication between the exploring and stationary nodes. Computation of $\frac{\partial \lambda_2(\boldsymbol{x})}{\partial \boldsymbol{x}}$ gives us a direction to move the robot system to improve the connectivity measure, and the differential constraint of Eq. 8 gives us motion limits to prevent communication from being lost. All that remains is to deal with the walled environment and combine all the pieces into one framework.

We assume that each robot has some local sensing capabilities and can recognize nearby rectilinear walls. If a wall is close by, then we find its outward normal $\boldsymbol{n}$ and add the differential constraint:

$$\boldsymbol{n}^T \boldsymbol{x}_i \geq 0 \tag{9}$$

Finally, we formulate the task as a bounded linear program to choose an input $\boldsymbol{u} = [\boldsymbol{u}_1^T \cdots \boldsymbol{u}_N^T]^T$ to make the change in λ_2 as positive as possible while keeping the differential constraints:

$$\max_{\boldsymbol{u}} \frac{\partial \lambda_2(\boldsymbol{x})}{\partial \boldsymbol{x}} \boldsymbol{u} \tag{10}$$

$$\|\boldsymbol{u}_i\|_\infty \leq d_i \tag{11}$$

$$\hat{u}'(c_{\alpha\beta}(\boldsymbol{x}))(\nabla c_{\alpha\beta}(\boldsymbol{x}))\boldsymbol{u} \geq 0 \tag{12}$$

$$W\boldsymbol{u} \geq 0 \tag{13}$$

$$\boldsymbol{u}_\beta = \dot{\boldsymbol{x}}_\beta(t) \tag{14}$$

where W is a matrix containing all of the wall constraints like Eq. 9, d_i is step-limit for robot i, and the motion of the exploring robot, $\dot{\boldsymbol{x}}_\beta$, is known. Although using a 2-norm limit on $\boldsymbol{u}_i$ would be more natural, the choice of the ∞-norm is a relaxation that keeps the problem as a linear program.

Note that since the step limit is based on an ∞-norm, the wall detection occurs in a square-shaped sensing region.

Although it may be possible to solve this program in a decentralized fashion, we have made no attempts to do so here.

III. Simulation

We have performed a simulated experiment with 8 robots in a walled environment, seen in Fig. 3; the map dimensions are 20 meters by 20 meters. There is one exploring robot (the diamond) that travels through the rooms and one stationary robot (the square) sitting in the starting area. The k-connectivity limit of Eq. 8 was set as 6, and the step limit d_i was set to 0.1 m. The sigmoid sharpness ω was set to 40, the saturation radius ρ was set to 2 meters, and the cutoff radius R was set to 8 meters (as seen in Fig. 1). In Fig. 3, the circles indicate the communication cutoff radius.

As the experiment proceeded, the configuration in the lower-left room stretched out to keep connection with the exploring robot as it moved to the upper-left corner, seen in panel 2 of Fig. 3. When the exploring robot went into the hallway on the right (panel 3), the connecting robots became trapped and the link was lost (panels 4 and 5), and finally when the exploring robot looped back around, it reconnected with the other robots (panel 6). This test was quantified in two different ways, as seen in Figs. 4 and 5. In both, the dotted lines indicate at what times the snapshots in Fig. 3 were taken. The initial snapshot was at time 0.

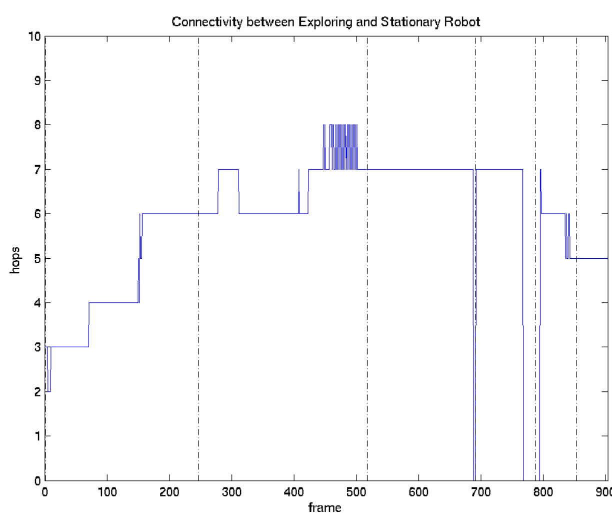

Fig. 4. Hop count of the robot network over time. The hop count limit was intended to be 6 or less at all times, but this limit was exceeded in the later stages as the lead robot got far away from the stationary one. Surprisingly, connectivity was still maintained afterwards.

First of all, the hop count between the stationary robot and the exploring one is plotted in Fig. 4. The constraint of Eq. 8 was supposed to keep this quantity less than or equal to 6 at all times, but clearly it exceeds that number and increases up to 8 at times and even loses connection (hop count zero) as the exploring robot moves in such a way as to trap the intermediary robots against the walls. This is discussed further in Sec. IV.

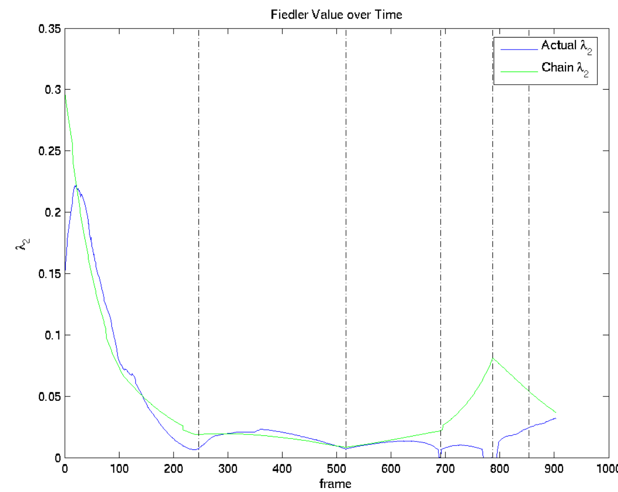

Fig. 5. Actual Fiedler value and idealized "chain" value of the robot network over time. The chain value is the Fiedler value that would be obtained if the robots were evenly spaced on a straight line between the lead robot and the stationary one, ignoring walls. By comparing the actual Fiedler value to the chain value, the effect of distance is in some sense normalized out, and the effect that the terrain displacement has on the connection quality is made apparent by the divergence of the two values.

The second quantity of interest is the Fiedler value, describing communication quality, and this is shown in Fig. 5. The main line shows the actual Fiedler value as it changes. Note that it is zero at snapshots 4 and 5, indicating that connectivity in the network was lost as the lead robot went out of range of the trapped intermediary robot. The second line shows the connection for an ideal chain configuration with robots evenly distributed on a straight line between the stationary and lead robot. It is intended to show that although the actual Fiedler value drops as the network stretches out, it is close to an idealized value, effectively normalizing it. When the two lines diverge substantially, the effect of the walls has become pronounced, causing the chain to stretch.

IV. Discussion

In this paper we used two measures of graph connectivity to develop a new framework for dealing with the task of moving robots to maintain communication between a stationary and exploring robot. First, the quality of the connectivity is measured using the second eigenvalue of the weighted Laplacian describing the network, and the derivative of this value provides information about how to move the robots to improve the connectivity. Second, the k-connectivity of the system is enforced using differential constraints on the motion. The two are combined to form a linear program that is solved at each step of motion. Although some parts have been made decentralized so far, the entire procedure is not yet decentralized.

Except for the periods when robots were prevented from maintaining connections because of intervening walls, controlling to increase the Fiedler value seems to have had an

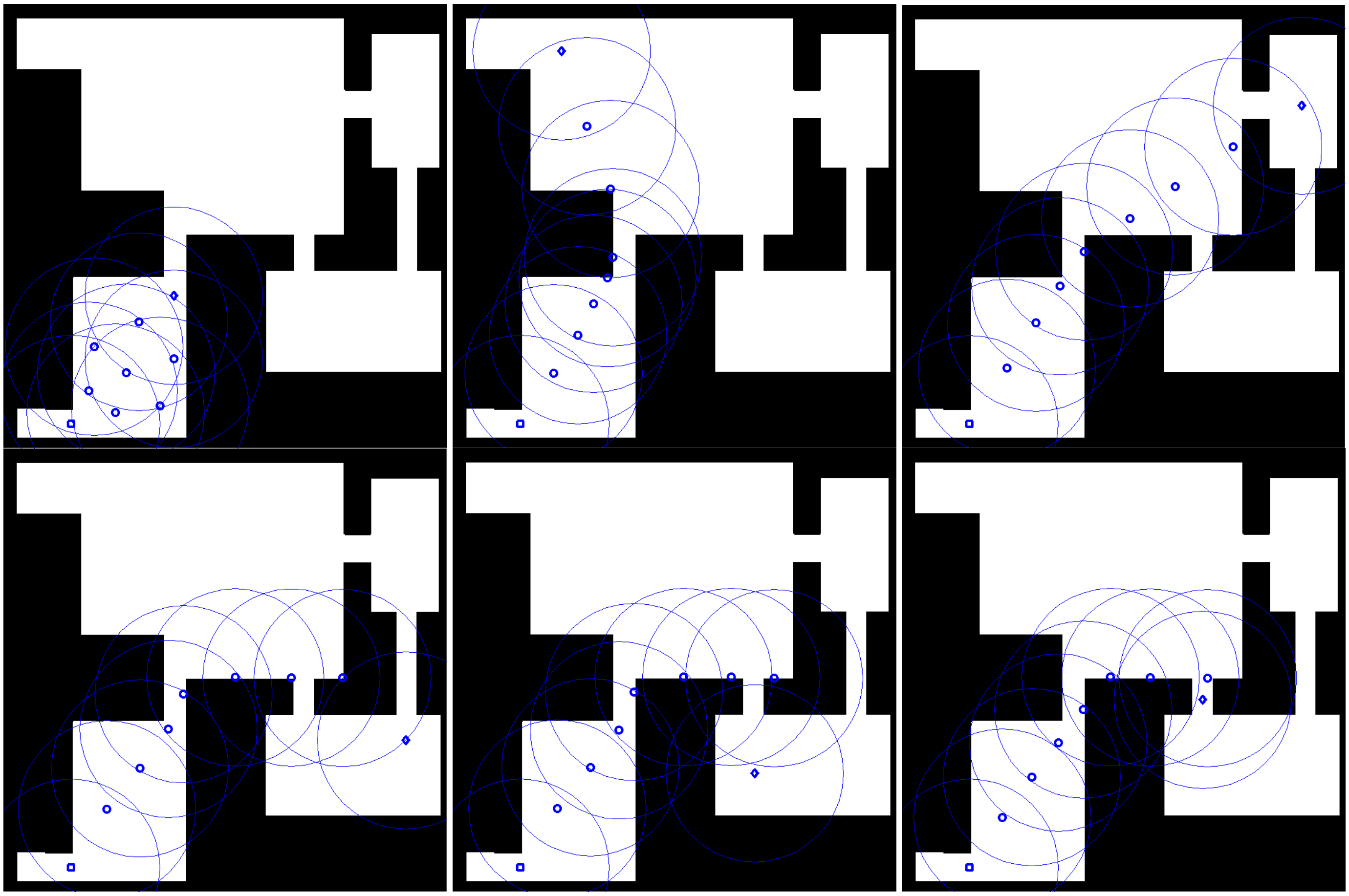

Fig. 3. Six snapshots of the robot team motion, starting from the initial configuration in the upper-left panel and proceeding from left-to-right in chronological order. Note that in panels 4 and 5, connectivity was lost because of the robots getting stuck against walls.

impact. As Fig. 5 shows, the actual Fiedler value was close to the idealized chain value for most of the run. Near snapshot 5, the robot nearest to the lead robot got caught in the corner and was unable to close the gap when the lead robot started moving back towards the group, causing the Fiedler value to diverge greatly from the chain value. However, this entire scheme is purely reactive and has no planning component that would allow the robots to recognize where paths exist to close the gaps. Adding in this planning could be a useful future direction.

The resulting positions of the robots during motion are close to locally optimal for the given positions of the stationary and lead robot. For the configurations shown in snapshots 2, 3, and 6 in Fig. 3, the lead robot was frozen and the controller was allowed to continue to improve the Fiedler value. The resulting Fiedler value changes are shown in Fig. 6. In all cases, the value does not change much at all.

One striking feature of the experiment is that at times the hop count exceeded the limit set at 6. This is actually to be expected given the setup because 6 hops is not enough to form an unbroken chain from one corner of the map to the other. However, the program kept moving along and providing motions to take that still kept the Fiedler value high and

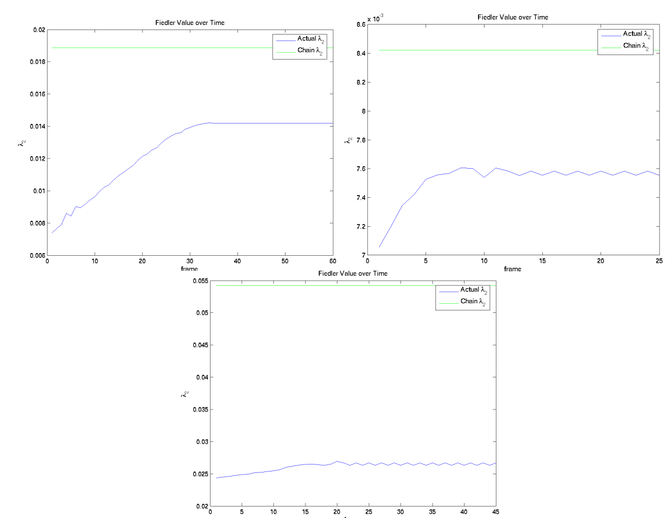

Fig. 6. Fiedler value changes after freezing the lead robot and letting the robot system stabilize; the lead robot was frozen in snapshots 2, 3, and 6 of Fig. 3, corresponding to the three frames above.

kept connectivity of higher orders than 6. It is likely that the contributions from the two measures are more or less important at different times.

One thing that we tried was to remove the k-connectivity constraint and control the formation solely with the derivative of the Fiedler value; the resulting motion was not able to maintain connection with the lead robot. With only Fiedler value considerations, we are trying to optimize a single value by changing several others, and so there is natural tradeoff between enhancing the value by adjusting robots near the stationary robot and enhancing connection with the lead robot as it moves away. At a certain point, the lead robot moved too fast for the other robots to react to its departure, and connection was lost.

When the k-connectivity is thrown in, the resulting motion keeps connectivity, and so these constraints seem to pushing the system away from the "lazy" solution of tweaking the regions with high connectivity already and instead forcing it to stretch out to pursue the lead robot. Even if the constraint becomes meaningless at a certain point because the intended hop count has been exceeded, it is possible that the robot chain has been stretched out to a point where it is no longer just as effective to adjust the robots near the start point.

All of these points lead to many directions for future research on this method of solving our connectivity-maintenance task. First of all, it would be useful to make the entire solution decentralized; doing so has advantages for deployment on physical robots and for scaling the solution to larger teams. Second, it would be worth exploring what effect the k-connectivity constraint is actually having on the solution once the fixed hop-count limit has been exceeded. If there is a subtle interplay between the two components of the program, it would be useful to discover it. Finally, with this purely reactive scheme, the problems with robots getting stuck against walls when trying to improve connectivity cannot be avoided. Adding in a planning component based off global *a priori* or local online maps would be an important step to take in making this a practical approach.

References

[1] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, pp. 988–1001, June 2003.

[2] G. Notarstefano, K. Savla, F. Bullo, and A. Jadbabaie, "Maintaining limited-range connectivity among second-order agents," in *American Control Conference*, (Minneapolis, MN), pp. 2124–2129, June 2006.

[3] M. M. Zavlanos and G. J. Pappas, "Controlling connectivity of dynamic graphs," in *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference*, pp. 6388–6393, 2005.

[4] J. M. Esposito and T. W. Dunbar, "Maintaining wireless connectivity constraints for swarms in the presence of obstacles," in *Proceedings of IEEE Conference on Robotics and Automation*, pp. 946–951, 2006.

[5] H. González-Baños, C.-Y. Lee, and J.-C. Latombe, "Real-time combinatorial tracking of a target moving unpredictably among obstacles," in *Proceedings of IEEE Conference on Robotics and Automation*, vol. 2, pp. 1683–1690, 2002.

[6] R. Murrieta-Cid, H. González-Baños, and B. Tovar, "A reactive motion planner to maintain visibility of unpredictable targets," in *Proceedings of IEEE Conference on Robotics and Automation*, vol. 4, pp. 4242–4248, 2002.

[7] D. Spanos and R. M. Murray, "Robust connectivity of networked vehicles," in *Proceedings of the 43rd IEEE Conference on Decision and Control*, December 2004.

[8] M. C. De Gennaro and A. Jadbabaie, "Decentralized control of connectivity for multi-agent systems," in *45th IEEE Conference on Decision and Control*, pp. 3628–3633, 2006.

[9] Y. Kim and M. Mesbahi, "On maximizing the second smallest eigenvalue of a state-dependent graph laplacian," in *Proceedings of the 2005 American Control Conference*, pp. 99–103, 2005.

[10] M. A. Hsieh, V. Kumar, and C. J. Taylor., "Constructing radio signal strength maps with multiple robots," in *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, (New Orleans, LA), pp. 4184–4189, April-May 2004.

[11] C. Godsil and G. Royle, *Algebraic Graph Theory*. New York: Springer-Verlag, 2001.

[12] J. Sun, S. Boyd, L. Xiao, and P. Diaconis, "The fastest mixing markov process on a graph and a connection to a maximum variance unfolding problem," *SIAM Review*, vol. 48, no. 4, pp. 681–699, 2006.

[13] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

[14] N. Z. Shor, *Minimization Methods for Non-Differentiable Functions*. Berlin: Springer-Verlag, 1985.