A Multi-Agent Coverage Algorithm with Connectivity Maintenance

Sungjoon Choi, Kyungjae Lee, and Songhwai Oh

Abstract— **This paper presents a connectivity control algorithm of a multi-agent system. The connectivity of the multi-agent system can be represented by the second smallest eigenvalue λ_2 of the Laplacian matrix L_G and it is also referred to as algebraic connectivity. Unlike many of the existing connectivity control algorithms which adapt convex optimization technique to maximize algebraic connectivity, we first show that the algebraic connectivity can be maximized by minimizing the weighted sum of distances between the connected agents. We implement a hill-climbing algorithm that minimizes the weighted sum of distances. Semi-definite programming (SDP) is used for computing proper weight $w^\star$. Our proposed algorithm can effectively be mixed with other cooperative applications such as covering an unknown area or following a leader.**

I. Introduction

The problem of cooperative control of a multi-agent system has received a considerable amount of attention in the field of robotics. Advances in autonomous agents such as unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs) have amplified the importance of cooperative control algorithms. In this context, the ability of a multi-agent system to reconfigure its formation is important and useful in many applications such as cooperation work in swarm robotics or floating mobile sensor networks estimating an oil-spill.

In control theory and robotics, a lot of research about maintaining connectivity of multi-agent systems have been conducted since sharing information between agents is often a key component to the success of multi-agent tasks. For controlling the connectivity, we first have to define the measure of connectivity. In a rendezvous case, the term connectivity can be easily defined since the goal of rendezvous case is to make agents to be located at the same place. In most cases, however, we do not want agents to collide each other, and we have to define a methodical measure of connectivity for this purpose.

For a criterion for connectivity, most of the studies represent the configuration of the multi-agent system with a Laplacian matrix and use the second smallest eigenvalue of the Laplacian matrix as a measure of connectivity. The second smallest eigenvalue of the Laplacian matrix is referred to as algebraic connectivity and often used as a measure of robustness and stability of the dynamic system [1]. In this paper, we also use this algebraic connectivity as an optimization criterion and implement a control algorithm to maximize this value.

The remainder of this paper is organized as follows. In Section II we place the current work in the context of previous research, and provide a short state-of-the art survey. In Section III, we state mathematical background for understanding a Laplacian matrix and the quasi-convexity of the second smallest eigenvalue to the distances between agents. Using this quasi-convexity we formulate a control algorithm maximizing the second smallest eigenvalue of the dynamic graph. We also present a SDP (Semi-definite problem) convex formulation which can be used effectively for avoiding local maxima. We also propose a control algorithm for maximizing the connectivity of a multi-agent system. In Section IV, we analyzed the experimental results of the proposed control algorithm.

The space of $n \times n$ real-valued matrix will be designated by $\mathbb{R}^{n\times n}$ and a real-valued symmetric matrix will be designated by $\mathbb{S}^n$. The matrix inequality based on semidefinite cone will denoted by $A \succeq B$, $A \preceq B$ when $A, B \in \mathbb{S}^n$, meaning $x^T(A - B)x \geq 0$, $x^T(A - B)x \leq 0$ for all $x \in \mathbb{R}^n$, respectively.

II. Related Work

A dynamic graph and its connectivity have been widely studied by a number of researchers. These researches can be divided into two parts: a centralized approach [1]–[4] and decentralized approach [5]–[9].

For a centralized approach, Zavlanos et. al. [4] deals with the problem of maintaining the connectivity of a time-varying dynamic graph. In this paper, k-hop connectivity is introduced for a multi-agent control algorithm to preserve the k-hop connectivity. Olfati-Saber et. al. [10] noted that the second smallest eigenvalue of a Laplacian matrix plays a crucial role in analysis of a consensus problem as well as a consensus of a networked dynamical system. In [1], [2], the authors proposed a control algorithm maximizing the second smallest eigenvalue corresponding to the multi-agent system using a convex formulation. However, due to the fact that the second smallest eigenvalue is non-convex with respect to the configuration, the authors separated configuration of the system and distances between agents to form a convex formulation.

In a decentralized approach, most of the papers used distributed convex optimization techniques such as distributed orthogonal iteration (DOI) [8] or decentralized power iteration [7] for estimating the second smallest eigenvalue of the multi-agent system. Zavlanos et.al. [9] considered the problem of controlling a group of agents so that the resulting motion always preserve the connectivity. De Gennaro et. al. [8] used DOI to estimate the Laplacian matrix of the overall system in a distributed manner and based on this estimation, they implemented a gradient controller for maximizing the second smallest eigenvalue. Meng Ji et. al. [6] proposed a

This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (NRF-2017R1A2B2006136).

S. Choi, K. Lee, and S. Oh are with the Department of Electrical and Computer Engineering and ASRI, Seoul National University, Seoul 151-744, Korea (emails: {sungjoon.choi, kyungjae.lee, and songhwai.oh}@cpslab.snu.ac.kr).

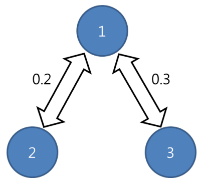

Fig. 1: A simple multi-agent system consists of three agents and two links

formation control algorithm over dynamic interaction graphs, and by adding appropriate weights to the edges in the graphs, they guaranteed that the graph stays connected. They defined an energy function based on distances between agents and formulated a control law minimizing this energy function. It is similar to our proposed algorithm except that our proposed algorithm utilizes specific weight heuristics to avoid local maxima. Schuresko et.al. [5] proposed a distributed algorithm which allows robots to know whether a desired cooperative motion breaks connectivity, and based on this, they proposed a distributed connectivity maintenance algorithm in robotic networks.

Our approach is related to the centralized approaches and differs from the previous approaches in a sense that we used quasi-concavity of the second smallest eigenvalue with respect to the distance between adjacent agents and changed original formulation into a gradient-based method for implementing a control algorithm. We use a SDP formulation for calculating weights of the cost function to avoid the local maxima.

III. Problem Formulation

A. Laplacian Matrix

A Laplacian matrix is a matrix representation of a graph used in the mathematical field of graph theory. Once an undirected graph $G = (V, E)$ is given, where V denotes a set of nodes and E denotes a set of edges, we can easily represent G into a Laplacian matrix $L \in \mathbb{R}^{n\times n}$. The Laplacian matrix is often used in multi-agent systems in that it can effectively represent its configuration. In a multi-agent system, each node V_i denotes an agent, and each edge E_{ij} denotes the strength of connectivity between the i-th and j-th agent. For a weighted symmetric Laplacian matrix, the weight $w_{ij} = w_{ji}$ of an edge varies between 0 to 1. The off-diagonal term $L_{i,j}$ is a minus weight between agent i and agent j where the diagonal term $L_{i,i}$ is the sum of weights connected to the agent i,

$$[L_G]_{i,j} = \begin{cases} -w_{ij} & \text{if } i \neq j \\ \sum_{k\neq i} w_{ik} & \text{if } i = j. \end{cases} \tag{1}$$

For example, if a multi-agent system is given as Figure 1, the corresponding Laplacian matrix is constructed as follows.

$$L_G = \begin{bmatrix} 0.5 & -0.2 & -0.3 \\ -0.2 & 0.2 & 0 \\ -0.3 & 0 & 0.3 \end{bmatrix} \tag{2}$$

The same matrix can be represented using an incidence matrix $A \in \mathbb{R}^{n\times m}$ and weight vector $w \in \mathbb{R}^m$ where n is the number of nodes and m is the number of links in the graph. This incidence matrix A indicates the connectedness between nodes. If an agent i and j are connected, we define $a_l \in \mathbb{R}^n$ as $(a_l)_i = 1, (a_l)_j = -1$, and since there are m connections among a multi-agent system, the incidence matrix is composed m a_l vectors, $A = [a_1, \cdots, a_m]$. Each element of w indicates the weight of the corresponding edge. Finally the weighted Laplacian matrix can be defined as

$$L_G = A \cdot \text{diag}(w) \cdot A^T \tag{3}$$

If we go back to the simple multi-agent system in Figure 1, we can easily calculate A, w and L_G.

$$A = \begin{bmatrix} 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}, w = \begin{bmatrix} 0.2 \\ 0.3 \end{bmatrix}$$

$$L_G = A \cdot \text{diag}(w) \cdot A^T = \begin{bmatrix} 0.5 & -0.2 & -0.3 \\ -0.2 & 0.2 & 0 \\ -0.3 & 0 & 0.3 \end{bmatrix} \tag{4}$$

We can see that the two Laplacian matrices calculated from different methods are equal. In general, the Laplacian matrix can be computed by (3) when w and A are given. By using this representation, we can formulate L_G as an affine function of w once A is given.

There are some important properties about this Laplacian matrix L_G that we are going to use throughout this paper. If the weights are positive, the L_G is positive semi definite. From this, we can use a convex formulation maximizing the second smallest eigenvalue of L_G using SDP. Also, the smallest eigenvalue of L_G is always zero and has column vector whose elements are all 1 as a corresponding eigenvector. The second smallest eigenvalue of the L_G indicates the connectivity of the graph and also referred to as an algebraic connectivity. If the graph is fully connected, i.e., there is no isolated node, the algebraic connectivity is always bigger than 0 and less than the number of nodes. This algebraic connectivity will be used as a measure of connectivity throughout this paper.

B. Primary Convex Formulation

As we have seen above, the formation of the multi-agent system can be represented as a Laplacian matrix and its second smallest eigenvalue λ_2 indicates its algebraic connectivity. Since our objective is to maximize the connectivity of the multi-agent system, our problem can be changed into maximizing the algebraic connectivity under certain conditions, and this optimization problem can be solved via a semi definite problem (SDP) solver.

Since λ_2 can be defined as $\min(\lambda_2, \cdots, \lambda_n)$, and $\min$ is a concave function, λ_2 is a concave function of weight vector w [11]. This concavity allows us to apply a convex optimization to maximize λ_2. If we allocate w with a fixed total weight, i.e., $\sum(w) = 1$, maximize the algebraic connectivity of the Laplacian matrix is a convex problem where the maximized algebraic connectivity is often referred to as an absolute algebraic connectivity [12].

However, in the field of control, a control parameter is the configuration of the formulation not w and the function that maps the configuration to w is usually non-convex and

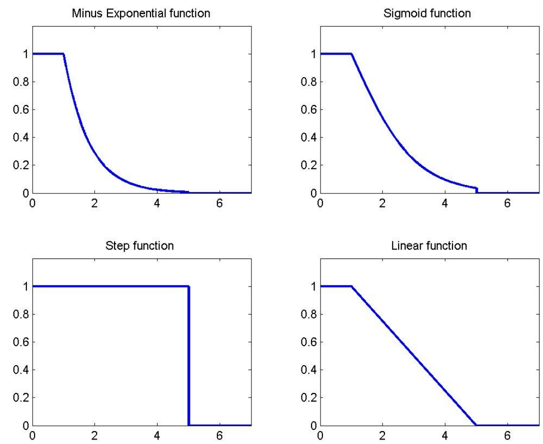

Fig. 2: x-axis indicates the distance between two agents and y-axis indicates the weight between two agents. Four candidates of weight functions. In all cases, d_{min} is 1, d_{max} is 5. Note that all of these functions are non-increasing function of d.

multimodal. Because of this non-convexity most of the work used approximated algorithms and evaluates the algorithms with simulations only [4]–[6], [11], [13].

In most cases, w_{ij} is computed by the function U of the distance d_{ij} between the agent i and agent j. The function U has many forms as shown in Figure 2 [2], [5]. All of them, however, are quasi-concave and non-increasing function of $d_{ij} - d_{min}$ when $d_{ij} > d_{min}$ and constant when $d_{ij} < d_{min}$ where d_{min} is a threshold distance, i.e., $w_{ij} = U(d_{ij} - d_{min})$. It is natural in that the connectivity will decrease as the distances between agents elongates and become 0 if the distance exceeds a certain threshold. Since λ_2 is an increasing function of w, we can easily see that λ_2 is a non-increasing quasi-concave function of $d'_{ij} = d_{ij} - d_{min}$ for $d_{ij} > d_{min}$. For the rendezvous algorithms, we can simply set d_{min} to be 0.

Proposition 1: λ_2 is as a quasi-concave function of d' where

$$d'_{ij} = d_{ij} - d_{min} \text{ where } d_{ij} \geq d_{min}$$

Proof: λ_2 is an increasing concave function of w and w_{ij} is quasi-concave function of d'_{ij}. Let $\lambda_2 = g(w)$ and $w = U(d')$, then $\lambda_2 = g(U(d'))$ where g is an increasing function and U is a quasi-concave function. We can easily show the quasi-concavity of the λ_2 using following property of quasi-concave function.

f is quasi-concave if and only if for every x and y and every λ with $0 \leq \lambda \leq 1$, if $f(x) \geq f(y)$ then $f((1-\lambda)x + \lambda y) \geq f(y)$

Suppose that

$$f(x) = g(U(x))$$

$$f(y) = g(U(y))$$

$$f(x) \geq f(y)$$

Since g is a increasing function, $U(x) \geq U(y)$ and by quasi-concavity of U, $U((1-\lambda)x + \lambda * y) >= U(y)$, again using the fact that g is increasing,

$$g(U((1-\lambda)x + \lambda * y)) \geq g(U(y)) \tag{5}$$

We have shown that if $g(U(x)) > g(U(y))$ then $g(U((1-\lambda)x + \lambda * y)) \geq g(U(y))$, proving that $g(U(x))$ is quasi-concave. ■

Proposition 2: λ_2 of L_G is maximized when d_{sum} is minimized.

$$d_{sum} = \sum_{i,j} w_{ij}(d_{ij} - d_{min})^2, \;\; w_{ij} \geq 0 \tag{6}$$

Proof: By proposition 1, we know that λ_2 is a quasi-concave function of each d_{ij}. If a function is quasi-concave it can either be monotonic or unimodal, and since w_{ij} is a non-increasing function of d_{ij} and λ_2 is a non-decreasing function of w_{ij}, λ_2 must be monotonic and in this case it is a non-increasing function of d_{ij}.

Minimizing d_{sum} is equivalent to minimizing each d'_{ij} when weights w_{ij} are positive. By this we can easily see that λ_2 is maximized when d_{sum} is minimized. ■

For maximizing the connectivity of a multi-agent system, our objective has now changed to minimize d_{sum} in (6) by controlling the configuration of a multi-agent system. However, since w_{ij} and d_{ij} are a function of a configuration x of multi-agent system, d_{sum} is a multi-modal function of the states of a multi-agent system, i.e., $d_{sum}(x)$. Therefore, if we directly apply gradient method to this d_{sum} with uniform weights, it is likely to fall into local minima. In this context, we separate the minimization of d_{sum} into two steps: finding the best allocation of edge weights and reconfiguration of the multi-agent system, in order to avoid the local minima.

We can use the concept of the absolute algebraic connectivity for this purpose. Let us define $w^\star$ as the optimal edge weights which achieves the absolute algebraic connectivity for given incidence matrix A [12]. The $w^\star$ can be obtained by solving a following convex optimization problem in (7) using SDP solvers such as SeDuMi [14] or CVX [15]. Note that $w^\star$ is the optimal allocation of edge weights and is different from the edge weights w of current configuration x.

$$\begin{array}{ll} \underset{\gamma, w}{\text{maximize}} & \gamma \\ \text{subject to} & L = A \cdot \text{diag}(w) \cdot A^T \\ & \sum_{i=1}^{m} w_i = 1 \\ & P^T \cdot L \cdot P \geq \gamma \cdot I_{n-1} \end{array} \tag{7}$$

where P is a matrix spanned by the nullspace of a vector whose elements are all 1. After computing $w^\star$, we reconfigure the location of each agent to reach $w^\star$.

Since the meaning of the $w^\star$ is the best allocation of finite resource w for maximizing the connectivity of the multi-agent system, we can regard that $w^\star_{ij}$ indicates the importance of the corresponding edge connecting node i and j. By adding more weights to corresponding nodes, we can reduce the time needed for reaching the equilibrium as well as chances falling into the local maxima. The performance of gradient-based method using the weighted cost function is verified by experiments.

C. Multi-agent Connectivity Maintaining Algorithm

We implemented a hill-climbing algorithm maximizing the multi-agent connectivity defined by the algebraic connectivity λ_2 of a Laplacian matrix L_G of the dynamic multi-agent system. Using Proposition 2, maximizing λ_2 of L_G is equivalent to minimizing the weighted sum

$$d_{sum} = \sum_{i,j} w_{ij}(d_{ij} - d_{min})^2 \text{ with proper weight } w \geq 0. \tag{8}$$

We first represent given multi-agent system with an incidence matrix A containing connectedness information. Using A, we calculate $w^\star$ which achieves the absolute algebraic connectivity. Then, we define the cost function as a weighted squared sum of all the connected distance minus d_{min}. The control input can be derived from the normalized gradient of this cost function.

After controlling the agents, the weights of edges are computed for measuring the connectivity of multi-agent system. For calculating the weights of edges, we used a squared exponential function. Since we have shown that every non-increasing quasi-concave function satisfies Proposition 2, other types of functions can be freely used to define the connectivity of system.

$$w_{ij} = \begin{cases} 1 & \text{if } d_{ij} < d_{min} \\ 0 & \text{if } d_{ij} > d_{max} \\ \exp(\frac{-(d_{ij}-d_{min})^2}{\sigma}) & \text{otherwise} \end{cases} \tag{9}$$

It is worth noting that we don't directly use λ_2 of L_G to implement the control algorithm, but we used the equivalent formulation which maximizes the algebraic connectivity based on Proposition 2. The overall algorithm is summarized in Algorithm 1.

Algorithm 1 Multi-agent Connectivity Control Algorithm

Given Initial multi-agent formulation $x \in \mathbb{R}^{2n}$
repeat
1. Calculate $w^\star$ which achieves absolute algebraic connectivity using (7).
2. $x_{control} = \frac{\nabla_x CostFunction(x,w^\star)}{||\nabla_x CostFunction(x,w^\star)||}$
3. **If** $max(x_{control}) > control_{max}$
   $x_{control} = \frac{x_{control}}{max(x_{control})} control_{max}$
4. $x = x + x_{control}$

until Algebraic connectivity λ_2 of the multi-agent system converges.

Function output = $CostFunction(x, w^\star)$
  $d_{sum} = 0$
  for every i, j connected
    $d_{ij} = ||x_i - x_j||$
    $d_{sum} = d_{sum} + w^\star_{ij}(d_{ij} - d_{min})^2$
  end
  output = d_{sum}
end

Moreover, in practice, controlling multi-agent systems based on gradient information using the proposed cost function has several advantages.

First, as we have seen above, the optimization parameter λ_2 is non-convex with respect to the configuration x of a multi-agent system. Because of this non-convexity, several approximation techniques, such as linearizion [4] or separating optimization variables [2], are applied. However, these approximations and convex formulation make it hard to adapt to actual mobile robots since it is hard to add real physical constraints such as maximum velocity or sampling period. Gradient-based method has an advantage on actual implementation in that it is possible to incorporate such physical constraints by setting a proper step size or scaling the gradient based on the maximum velocity.

Second, if we control the multi-agent systems based on distances between agents, we can implement a fully distributed control algorithm maintaining connectivity. Most of the existing distributed algorithms assume that an agent receives information from its neighbors within certain communication radius [5]–[7]. While these are undoubtedly meaningful researches, this assumption does not fit into low-cost indoor mobile robots since it is hard to implement ad-hoc communication between agents for these robots. Most of the indoor robots are only aware of their relative location, and it is the reason why it is hard to adapt existing multi-agent algorithms to mobile robots [3]. If we implement a control algorithm based on distance information, each agent only requires the distance and relative angle of other agents in its field of view (FOV). One of the few papers that actually perform experiments using real robots adapts similar decentralized algorithms [16].

IV. Experiments

We performed experimental simulations with 10, 20, 30 agents in a walled environment of 2 by 2 unit distance. Initialized with random positions inside the environment, we compared our proposed control algorithm with uniform weight control algorithm with respect to trajectory of the agents, time required for reaching equilibrium, and temporal change in algebraic connectivity. d_{min} and d_{max} are set to 0.6 and 1.2, respectively. The two trajectories are shown in Figure 3.

First of all, the algebraic connectivity of the multi-agent system increases over time as illustrated in Figure 5. This results show that the time required for reaching the equilibrium is decreased when using proper weights $w^\star$. The red and blue lines indicate the temporal change of algebraic connectivity of the multi-agent system using the proposed algorithm and simple distance algorithm, respectively. Note that trembling at the equilibrium state is due to the step size of the gradient. This trembling also verifies our formulation was correct. Since we minimized d_{sum} in Algorithm 1 for maximizing the λ_2 of the Laplacian matrix L_G, some could argue the inconsistency of the optimization variables. This trembling of the second smallest eigenvalue inversely validate our proposed hill-climbing algorithm can maximize λ_2.

The second quantity of interest is avoiding local optima. Since the trajectory must be continuous and multi-modality of the mapping function between configuration x and d, it is likely to fall into local maxima. Two final configurations and temporal changes of λ_2 started from the same configuration are plotted in Figure 4. We can see that the our proposed control algorithm can effectively avoid falling into poor local maxima. We checked the performance of our proposed algorithm with respect to avoiding local maxima

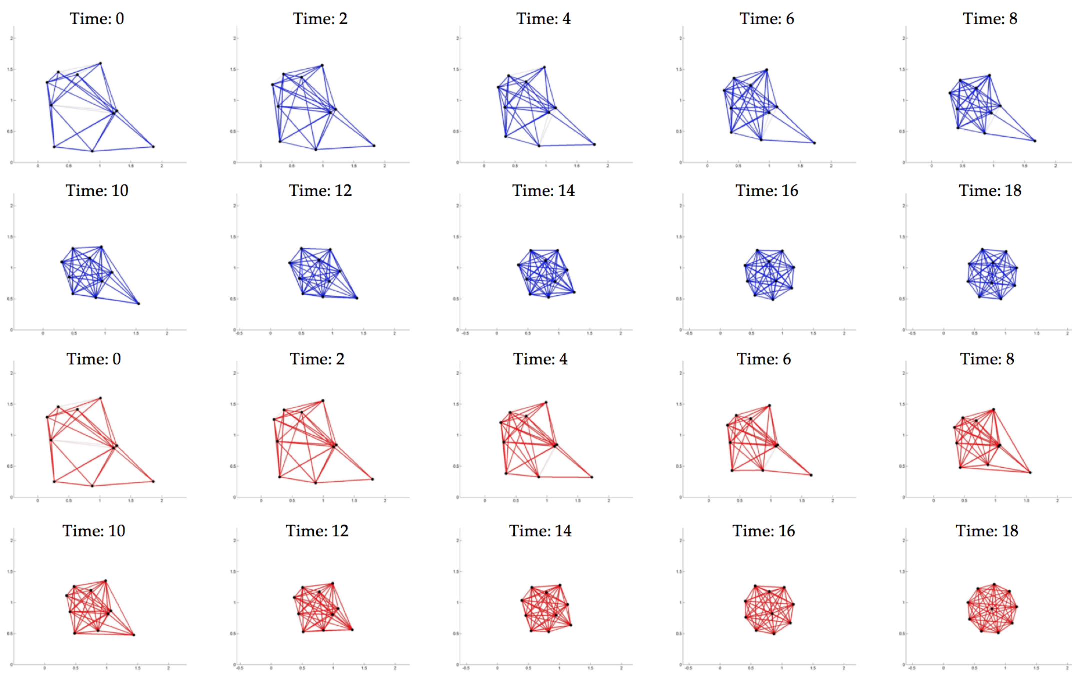

Fig. 3: Two chronological trajectories resulted from our proposed algorithm and distance-only algorithm. Upper blue trajectory indicates results from distance-only algorithm and lower red trajectory indicates results from our proposed algorithm. The figures are sorted in chronological order, left-to-right, and up-to-down.

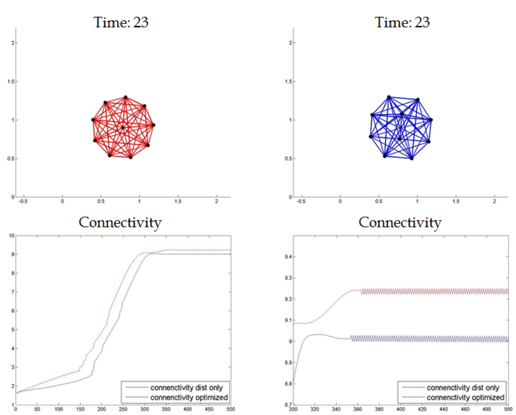

Fig. 4: Comparison of two final configuration using distance-only algorithm and our proposed algorithm. Red and blue line indicates results from our proposed algorithm and distance-only algorithm, receptively. Bellow figures are temporal change of λ_2 of two algorithms. The lower-right figure enlarged the equilibrium state of lower-left figure.

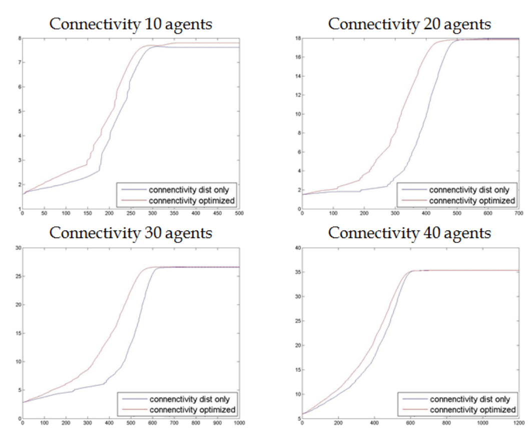

Fig. 5: Four temporal varying algebraic connectivity λ_2 with different number of agents, 10, 20, 30, 40 from left-to-right, from up to down.

by additional 100 independent comparative experiments with 10 agents started from random initial configuration. While distance-only control falls into local maximum for 21 times, our proposed algorithm falls for 7 times.

We should point out that d_{min} is not an actual minimum distance but a desired distance between agents. This can be seen as a trade-off between using the proposed hill-climbing algorithm and a usual convex optimization solver. When using a convex optimization, the initial state must satisfy inequality constraints, in context of multi-agent control, all distances between agents must be bigger than or equal to minimum distance. Because of this, if the distances are less than the minimum distance, convex solvers cannot work properly. Our gradient formulation, however, can handle this situation effectively. One heuristic method to satisfy d_{min} is that when agents reached equilibrium state, we increase d_{min} until the the minimum distance between agents exceeds desired d_{min}.

V. Future Works

One of the advantages of using the gradient method based on a relative simple cost functio is that it can easily be

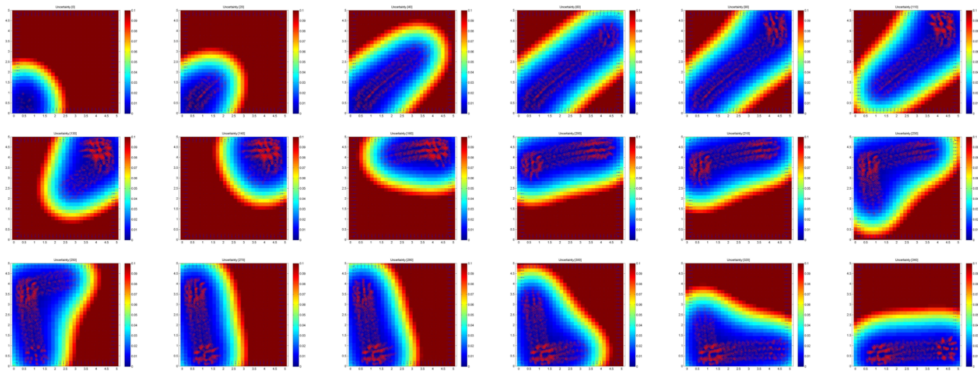

Fig. 6: Chronological trajectory of multi-agent system covering unknown area while maintaining connectivity. Colors indicates the estimated uncertainty (variance) predicted by Gaussian processes. The initial configuration is the upper-left figure and the figures are sorted in chronological order, left-to-right, and up-to-down. Each agent contains its latest 20 positions for calculating uncertainty using Gaussian processes.

combined with other gradient methods. For example, the proposed control algorithm can be combined with a coverage algorithm to implement cooperative coverage control algorithms maintaining connectivity as shown in Figure 6. We combined two normalized gradient term; gradient of cost function in Algorithm 1 and gradient of uncertainty predicted by Gaussian processes [17]. For a convex optimization formulation, however, it is not easy to add additional constraint to existing formulations.

With respect to implementing our proposed algorithm to actual mobile agents, our algorithm has several advantages. Since both localization and ad-hoc communication between agents are hard to be implemented for indoor mobile agents, most of the existing connectivity control algorithms are difficult to be realized. However, as mentioned in Section III-C, by controlling the agents based on the distance d_{ij} between agent i and j, we can implement the control algorithm in a fully distributed way.

VI. Conclusions

In this paper, we have shown that the algebraic connectivity of a Laplacian matrix is a non-increasing quasi-concave function of distances between agents in a multi-agent system. Using this property, we were able to convert a complex SDP formulation maximizing the second smallest eigenvalue of the Laplacian matrix to relatively simple gradient method using the cost function based on distances d_{ij} between agents. Because of multi-modal relations between control variable x, configuration of agents, and distances d_{ij}, simple gradient-based method is likely to fall into local maxima. For avoiding this, we proposed the concept of absolute algebraic connectivity and used the allocated weight $w^{\star}$ for formulating the weighted cost function. We have shown that the gradient control algorithm with the weighted cost function is three times less likely to fall into poor local maxima than that of using a simple cost function.

References

[1] R. Saber and R. Murray, "Consensus protocols for networks of dynamic agents," in *American Control Conference, 2003. Proceedings of the 2003*, vol. 2, 4-6, 2003, pp. 951 – 956.

[2] Y. Kim and M. Mesbahi, "On maximizing the second smallest eigenvalue of a state-dependent graph laplacian," *Automatic Control, IEEE Transactions on*, vol. 51, no. 1, pp. 116 – 120, Jan. 2006.

[3] W. Ren, R. Beard, and E. Atkins, "A survey of consensus problems in multi-agent coordination," *American Control Conference, 2005. Proceedings of the 2005*, pp. 1859–1864 vol. 3, June 2005.

[4] M. Zavlanos and G. Pappas, "Controlling connectivity of dynamic graphs," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, Dec. 2005, pp. 6388 – 6393.

[5] M. Schuresko and J. Cortes, "Distributed motion constraints for algebraic connectivity of robotic networks," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, Dec. 2008, pp. 5482 –5487.

[6] M. Ji and M. Egerstedt, "Distributed coordination control of multiagent systems while preserving connectedness," *Robotics, IEEE Transactions on*, vol. 23, no. 4, pp. 693 –703, Aug. 2007.

[7] P. Yang, R. Freeman, G. Gordon, K. Lynch, S. Srinivasa, and R. Sukthankar, "Decentralized estimation and control of graph connectivity in mobile sensor networks," in *American Control Conference, 2008*, June 2008, pp. 2678 –2683.

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

[9] M. Zavlanos and G. Pappas, "Distributed connectivity control of mobile networks," *Robotics, IEEE Transactions on*, vol. 24, no. 6, pp. 1416 –1428, Dec. 2008.

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

[11] S. Boyd, "Convex optimization of graph laplacian eigenvalues," in *International Congress of Mathematicians*, pp. 1311–1319.

[12] M. Fiedler, "Absolute algebraic connectivity of trees," *Linear and Multilinear Algebra*, vol. 26, no. 1-2, pp. 85–106, 1990. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/03081089008817967

[13] K. Srivastava and M. Spong, "Multi-agent coordination under connectivity constraints," in *American Control Conference, 2008*, June 2008, pp. 2648 –2653.

[14] J. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11–12, pp. 625–653, 1999, version 1.05 available from `http://fewcal.kub.nl/sturm`.

[15] I. CVX Research, "CVX: Matlab software for disciplined convex programming, version 2.0 beta," http://cvxr.com/cvx, Sep. 2012.

[16] N. Michael, M. M. Zavlanos, V. Kumar, and G. J. Pappas, *Maintaining Connectivity in Mobile Robot Networks*, 2008.

[17] M. Jadaliha, Y. Xu, J. Choi, N. Johnson, and W. Li, "Gaussian process regression for sensor networks under localization uncertainty," *Signal Processing, IEEE Transactions on*, vol. PP, no. 99, p. 1, 2012.