Sensing-based Shape Formation on Modular Multi-Robot Systems: A Theoretical Study

Chih-Han Yu
School of Engineering & Applied Sciences
Harvard University
Cambridge MA 02138 USA
chyu@fas.harvard.edu

Radhika Nagpal
School of Engineering & Applied Sciences
Harvard University
Cambridge MA 02138 USA
rad@eecs.harvard.edu

ABSTRACT

This paper presents a theoretical study of decentralized control for sensing-based shape formation on modular multi-robot systems, where the desired shape is specified in terms of local sensor constraints between neighboring robot agents. We show that this problem can be formulated more generally as *distributed constraint-maintenance* on a networked multi-agent system. It is strongly related to a class of multi-agent algorithms called *distributed consensus*, which includes several bio-inspired algorithms such as flocking and firefly synchronization. By exploiting this connection, we can theoretically analyze several important aspects of the decentralized shape formation algorithm and generalize it to more complex multi-agent scenarios. We show that the convergence time depends on (a) the number of robot agents and agent connection topology, (b) the complexity of the user-specified goal, and (c) the initial state of the robots. Using these results, we can provide precise statements on how the approach scales, and how quickly the system can adapt to perturbations. These results provide a deeper understanding of the contrast between centralized and decentralized multi-agent algorithms.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Algorithm, Theory

Keywords

Decentralized Control, Modular Robot, Collective Dynamics, Shape Formation, Convergence, Graph Laplacian

1. INTRODUCTION

Modular robots are a class of robotic systems composed of many identical, physically connected, programmable modules that can coordinate to change the shape of the overall robot. By transforming its shape, a modular robot can adapt to many tasks, from different modes of locomotion like crawling and climbing [12], to forming temporary robotic structures such as stairs or bridges [2, 7, 13]. A decentralized modular robot framework can be viewed as a multi-robot or multi-agent system, since the overall function and shape change is achieved by coordinating the independent actions of the individual modules. A key challenge is developing decentralized multi-agent algorithms that can form complex shapes in a scalable, robust and analyzable manner.

In our recent work [13], we have proposed a new type of shape formation task for modular robots: instead of specifying the absolute configuration of the whole desired shape, the shape is expressed as a set of *relative* orientation constraints between neighboring modules. This allows the desired shape to be formed irrespective of environmental conditions, and adapt as environmental conditions change. We provided a decentralized solution to the problem in [13], where each agent adapts its shape based on neighborhood sensing, and we showed a partial convergence proof for that specific set of robot configurations. However, many important questions remain open: Does the approach work for agents connected in complex and irregular topologies? How does the algorithm scale as the number of agents increases? How quickly does the system adapt, given different goals?

In this paper, we present a theoretical analysis for this class of multi-agent tasks and algorithms. We show that this problem can be formulated more generally as a problem of *distributed constraint-maintenance* on a *networked multi-agent system*. This class of problems relates strongly to a class of problems in decentralized control theory called *distributed consensus*, which includes several canonical bio-inspired algorithms such as flocking and firefly synchronization. By exploiting this connection, we can theoretically analyze several important aspects of the decentralized shape formation algorithm, and we can extend the algorithmic approach to new types of sensor-actuator agents and arbitrary multi-agent network topologies.

Our contributions are as follows: we show that this class of algorithms can be formulated as collective linear dynamical systems. We prove that the algorithms converge to the correct user-described goal, for any undirected connected graph of agents. We derive the convergence rate and show how different factors of the agent network topology, e.g. number of agents and network diameter, determine the convergence time. This allows us to analytically reason about the scalability of the decentralized approach. We also quantify the impact of initial state and goal state, and show that the convergence time depends logarithmically on the distance between the two. This allows us to bound the worst case

Cite as: Sensing-based Shape Formation on Modular Multi-Robot Systems: A Theoretical Study, Chih-Han Yu and Radhika Nagpal, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. 71-78.

number of iterations and compute how quickly the system reacts to small perturbations. Thus we can understand how this approach scales to the number of agents, generalizes to different shapes and adapts to perturbations. We show that this approach is generalizable to a large class of feedback functions. Finally, we use these results to show that decentralized algorithms have an advantage over tree-based centralized algorithms when it is important to adapt quickly to constant environment changes.

The rest of the paper is as follows: Section 2 describes related work. Section 3 describes the networked multi-agent model and Sections 4, 5, and 6 present the decentralized algorithm and theoretical properties of the algorithm. We show a more general class of the control algorithm in Section 7 and contrast decentralized and centralized approaches in Section 8.

2. RELATED WORK

Several groups have designed centralized and decentralized shape formation algorithms for modular robots [2, 10]. Centralized algorithms suffer from scalability problems, but decentralized solutions have been shown to be more scalable as system size increases. Decentralized algorithms often utilize nearest neighbor rules where the control parameters of a module depend on relationships between neighboring modules. For example, in [10] a module uses its connection configuration with neighboring modules to compute an action for the next time step. In [13], the modules' control parameters are computed based on relative orientation feedbacks from neighboring modules. While various decentralized approaches have been proposed, it is often difficult to prove basic properties of these algorithms, including correctness. Several important properties of the algorithms are rarely studied, such as how the speed of shape formation process is related to the complexity of the shape or the connection topology of the modules.

On the other hand, nearest neighbor rules have been widely studied in the control theory community, including in consensus problems [9], flocking [8], and multi-vehicle formation [4]. In addition, there are many results developed in graph Laplacian and matrix theory that can be utilized to study such algorithms [6, 11]. In this paper, we bridge the results from these areas. We use our previous framework [13] as an example to show how control problems in decentralized modular robot systems can be formulated as networked multi-agent systems. By exploiting connections with the above theories, many important properties can be characterized.

3. NETWORKED MULTI-AGENT MODEL

Sensing-based Shape Formation

Here we briefly review an abstract version of the modular robot from our previous work [13]. As shown in the Figure 1 (top), modules are connected to form a flexible surface on the top with supporting legs attached to the surface. In each local structure, a pivot module (A) connects to a supporting group (B) and one or more surface groups (C). Pivot modules play the coordination role in the decentralized algorithm. Supporting groups are thought of as legs that can compress and extend their heights. Each supporting group is composed of multiple modules that are linearly connected, and can vary its length via modules' actuation.

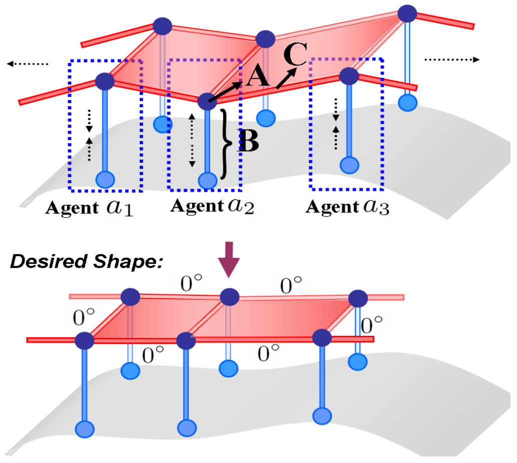

Figure 1: Top: A skeleton view of the modular robot. A: Pivot module; B: Supporting group. A and B are jointly viewed as an agent. Surface group C is viewed as an inter-agent link. Bottom: The desired shape (level surface) is specified with a set of local desired tilt angles (constraints).

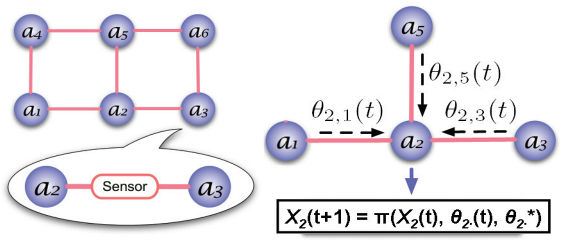

Figure 2: Left: The modular robot can be viewed as a graph with nodes represent agents and edges represent inter-agent links. There is a sensor on each edge that provides inter-agent tilt angle. Right: Each agent's control algorithm takes all neighboring sensor readings as inputs to compute the agent's new state.

Surface groups, which are also composed of multiple modules, provide sensor information and and act as communication channels between pivot modules.

As shown in the bottom diagram of Figure 1, the desired goal shape is specified with a set of desired tilt sensor angles (constraints) on surface groups that connect two neighboring pivot modules. Each pivot can only observe its neighboring surface groups' sensory output and each of them has multiple constraints to satisfy. When the robot is placed in an unknown environment (e.g. on a rough terrain), pivots coordinate supporting groups to change their heights in such a way that all desired constraints are satisfied.

Formal Model

In this section, we model this system as a network of agents consisting of the following components:

1. Agent: An agent is defined as a unit that has independent *computation*, *communication*, and *actuation* capabilities. In the modular robot of Figure 1, we consider each pivot module (A) plus its underlying supporting group (B)

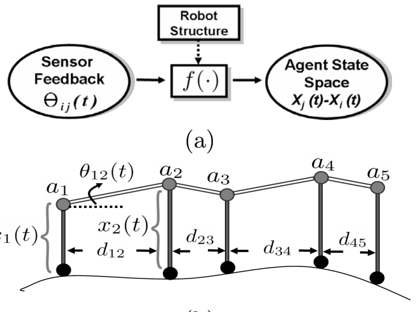

Figure 3: (a) A mapping function between sensor space and agent state space can be constructed by exploiting the local structure of the robot. (b) An example of exploiting the robot structure.

as a single agent a_i. The robot R is composed of n such agents: $R = \{a_1, a_2, \cdots, a_n\}$.

2. Cooperation Graph: In our networked agent model, agents have only a one-hop *local view*. They achieve global tasks by cooperating and communicating with their immediate neighbors. We represent cooperation relationships as a graph G in which vertices represent agents and edges represent cooperation between an agent and its neighbors. For example, in Figure 2 (left) edges correspond to the surface groups in Figure 1. We use N_i to denote the set of one hop neighbors of agent a_i. The neighbor relationship between agents is symmetric, so the edges in G are *undirected*.

3. Sensor: There is a sensor available to measure the relationship between each pair of the neighboring agents. We denote the sensor reading between neighboring agent a_i and a_j at t time step by $\theta_{ij}(t)$. In the example of Figure 1, $\theta_{ij}(t)$ measures the tilt angle of the surface group connected to two neighboring agents a_i and a_j.

4. Desired Shape: We assume that the desired global shape of the robot can be decomposed into the desired sensor reading for each pair of neighboring agents. We denote by θ^*_{ij} the desired sensor reading between neighboring agent a_i and a_j. For example, in Figure 1 the sensors measure tilt relative to the ground, and one example of a task is to maintain a level surface which means that $\theta^*_{ij} = 0$ for all i, j.

More generally, this type of task specification can be viewed as a constraint-based specification. The constraint between neighboring agents a_i and a_j is satisfied when the current relationship $\theta_{ij}(t)$ equals the goal relationship θ^*_{ij}.

5. Control Algorithm: Each agent is running an identical control algorithm π that controls the agent's state. We denote agent a_i's state at time t by $x_i(t)$. In the modular robot of Figure 1, $x_i(t)$ represents the height of the agent. At each time step, each agent updates its state based on the current state of the agent, current sensor readings, and the desired sensor readings (as shown in Figure 2 (right)). We assume that the agents are synchronized and update their state in rounds. Agent a_i's state update equation:

$$x_i(t+1) = \pi(x_i(t), \theta_{i\cdot}(t), \theta^*_{i\cdot}) \qquad (1)$$

where $\theta_{i\cdot}(t) = \{\theta_{ij}(t), \forall a_j \in N_i\}$ and $\theta^*_{i\cdot} = \{\theta^*_{ij}, \forall a_j \in N_i\}$.

This model both simplifies and generalizes the original model from [13]. Here, we also allow the agents to be connected in an *arbitrary* undirected graph as opposed to a mesh as shown in Figure 2. We also do not require tilt-height but allow any sensor-actuator relationship. This model captures a class of modular and mobile robot applications. For example, the environmentally-adaptive structures (table, bridge) described in [13] and a team of mobile robots whose overall shape can be specified as inter-robot relationships.

4. THE ALGORITHM

In this section, we describe a simple multi-agent control algorithm for achieving a desired shape. This control law exploits the structure of the robot and is briefly described in [13]. A generalized feedback form of the control law will be described later in Section 7.

Specifically, each a_i senses $\theta_{ij}(t)$ from each of its neighbors a_j and controls its own state $x_i(t)$. In our networked multi-agent model, the states of an agent's neighbors are not directly observable; instead the sensor values acts as a proxy for inter-agent relationships. There is a mapping function $f(\cdot)$ that transforms the sensory input into an actual difference between neighbor agent states (Figure 3 (a)):

$$f(\theta_{ij}(t)) = x_j(t) - x_i(t)$$

Suppose this mapping is known. For example, consider the modular robot model of Figure 3 (b). An agent's state represents its height, while the sensor provides the tilt angle between an agent and its neighbor. We assume that the distance between an agent and its neighbor a_j, d_{ij}, is known ahead of time. In this case,

$$f(\theta_{ij}(t)) = d_{ij}\tan(\theta_{ij}(t)) = x_j(t) - x_i(t)$$

DEFINITION: **Control Law**

$$\begin{aligned} x_i(t+1) &= \pi(x_i(t), \theta_{i\cdot}(t), \theta^*_{i\cdot}) \\ &= x_i(t) + \sum_{a_j \in N_i} \alpha(f(\theta_{ij}(t)) - f(\theta^*_{ij})) \qquad (2) \end{aligned}$$

If we expand the right hand side of Eq. 2, we see that the control law is operating in the agent state space. Let the desired state difference be called $\Delta^*_{ij} = f(\theta^*_{ij}) = x^*_j - x^*_i$. Each agent is summing feedbacks from all of its neighbors and then acting *linearly* towards it to reduce error with step size α, where $0 \le \alpha \le 1/|N_i|$:

$$x_i(t+1) = x_i(t) + \sum_{a_j \in N_i} \alpha(x_j(t) - x_i(t) - \Delta^*_{ij}) \qquad (3)$$

One can expect that different kinds of sensors or different agent's prior knowledge about robot structure will lead to different mappings $f(\cdot)$. As long as agents have sufficient information to map sensor information $\theta_{ij}(t)$ to agent state difference $x_j(t) - x_i(t)$,[1] then this control law can be used.

5. CORRECTNESS & CONVERGENCE RATE

In the control law we presented, an agent sums the feedback from its local neighbors and acts in some fashion towards that feedback. Since the system is decentralized with

[1]The control law can be used even though the mapping is to $\sigma(x_j(t) - x_i(t))$ and the constant σ is unknown. An example of this case is when the agents only know that they are equally apart from their neighbors, but the exact distances are unknown.

many agents acting on local information in parallel, a key question is whether these actions always will always produce the correct desired global goal (convergence from all initial states) and if so, how fast it will take to achieve the goal (convergence rate). In this section we show these properties of the algorithm can be characterized for arbitrary connected topologies and desired goals.

We first demonstrate how to aggregate all agent update equations to become *collective dynamics*. This allows us to study the emerged global behavior of all agents via analyzing a single dynamical system. We then show that the collective dynamics bears an important resemblance to a class of networked multi-agent problems called *distributed consensus*. By leveraging results from control theory and spectral graph theory, we prove the convergence property for the shape formation algorithm.

Collective Dynamics: Let $X(t)$ represent the ensemble of all agents' states at time t:

$$X(t) = (x_1(t), x_2(t), \cdots x_n(t))'$$

We can write the collective dynamics of all agents as:

$$X(t+1) = A \cdot X(t) + \tilde{b} \quad (4)$$

where $A = [\mathbf{a}_{ij}]$, an $n \times n$ matrix with element $\mathbf{a}_{ij}$ defined by:

$$\mathbf{a}_{ij} = \begin{cases} \alpha & \text{if } a_j \in N_i \text{ and } i \neq j \\ 1 - \alpha \cdot |N_i| & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

and where $\tilde{b}_i = \alpha \cdot \sum_{a_j \in N_i} \Delta^*_{ij}$ is a *bias vector*. We note that A is a row stochastic matrix since each row sums to 1. In addition, $\sum_i \tilde{b}_i = 0$, since $\Delta^*_{ij} = -\Delta^*_{ji}$ for all i, j.

Distributed Consensus: This is a process by which a network of agents can come to an agreement. A canonical example is the *average consensus problem*, in which agents must compute the average of their initial states. The dynamics of this agreement process can be formulated as:

$$X(t+1) - X(t) = -\alpha \cdot LX(t) \Rightarrow X(t+1) = A \cdot X(t)$$

where $A = I - \alpha L$, and matrix L is a special matrix called *graph Laplacian*. $L = [l_{ij}]$, and

$$l_{ij} = \begin{cases} -1 & \text{if } a_j \in N_i \text{ and } i \neq j \\ |N_i| & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

The properties of this solution have been well-explored, and convergence and convergence rates for this have been shown over arbitrary and even time-varying graphs. Olfati-Saber et al. [9] provide an overview of the different types of consensus problems and distributed solutions, and show that the alignment problem in flocking (agreeing on a single heading) and the synchronization problem (agreeing on a single phase) are consensus problems.

In the example of Figure 1, if the goal is to create a level surface $\tilde{b} = 0$, then the collective dynamics is exactly the *average-consensus problem*. This also tells us that the height of the level surface will be the average of the heights of the agents' initial heights. For a more complex desired shape goal, the collective dynamics differs by a bias term $\tilde{b}$. We will show that $\tilde{b}$ actually plays no role in the convergence analysis of the system. It allows the system to achieve *distributed constraint maintenance*; i.e. agreement on a solution that maintains a set of local constraints.

Convergence Property: The key result of [9] relates the eigenvalues of A and L to convergence of the local rules defined by these matrices.[2] The key point for us is that having a nonzero bias vector $\tilde{b}$ does not affect the eigenvalue analysis (proof in Appendix); thus, the results of [9], apply here as well. Specifically, let λ_i denote the i^{th} *smallest* eigenvalue of the graph Laplacian L, and let μ_i be the i^{th} *largest* eigenvalue of A. Then

$$\mu_i = 1 - \alpha\lambda_i.$$

L has a simple eigenvalue $\lambda_1 = 0$ with associated eigenvector $\mathbf{1}$ (an all ones vector), and A has as its largest eigenvalue $\mu_1 = 1$. As shown in [6], when the graph G is *connected* (with arbitrary topology), the *second* smallest eigenvalue of L, λ_2, is strictly larger than 0. Thus, the second largest eigenvalue μ_2 of A is strictly smaller than 1.

We can then prove that under the iteration of Eq. 4,

$$\|X(t) - X^*\|^2 \leq \mu_2^{2t} \|X(0) - X^*\|^2.$$

Thus, since $\mu_2 < 1$, we have:

THEOREM 1. *Let X^* be the desired shape state that $x^*_j - x^*_i = \Delta^*_{ij}$ $\forall a_i$ and $a_j \in N_i$, the collective dynamics in Eq. 4 converge to X^* for all initial conditions with exponential rate μ_2.*

PROOF. see Appendix. □

Because μ_2 does not depend on the bias vector $\tilde{b}$ at all, this convergence analysis is independent of desired system shape. From the convergence proof, we can also see that

$$\|Y(t+1)\| \leq \mu_2 \|Y(t)\| \quad (5)$$

where $Y(t) = \|X(t) - X^*\|$ represents the current distance from the desired goal.

6. FACTORS AFFECTING PERFORMANCE

In the previous section, we prove convergence and derived convergence rates. In this section we expand these results to answer several important questions about the shape formation process.

1. Scalability and impact of topology: How does the convergence time increase as we increase the number of agents and the diameter of the cooperation graph?

2. The effect of shapes: How do the initial shape and the desired shape affect the convergence time?

3. Reactivity: How do the agents react to perturbations from the desired state?

We provide precise answers to these questions. In doing so, it is useful to first derive an inequality for the number of iterations required to create a given shape within a certain error tolerance. By error tolerance, we mean:

[2]We can show that A and L have the same eigenvectors. Let v_i be the i^{th} eigenvector of L. $Lv_i = \lambda_i v_i \Rightarrow Av_i = (I - \alpha L)v_i = (1 - \alpha\lambda_i)v_i = \mu_i v_i$.

Definition: **ϵ-approximation.** The shape formed by the agents at state $X(t)$ is $\epsilon-$approximation of the desired shape if $X(t)$ satisfies: $Y(t) = \|X(t) - X^*\| \leq \epsilon$

The error tolerance ϵ represents the fact that agents have finite resolution in controlling their actuation, so some level of inaccuracy must be tolerated. From Theorem 1 , we know that the agents approach the desired state X^* with exponential rate. We can further express the number of time steps required to achieve the goal as a function of convergence rate μ_2, ϵ, the initial condition $X(0)$, and the goal condition X^*:

$$\begin{aligned} \|Y(t_{\max})\| &\leq \mu_2^{t_{\max}} \|Y(0)\| \leq \epsilon \\ \Rightarrow t_{\max} &\leq \left\lceil \log_{\mu_2} \left(\frac{\epsilon}{\|Y(0)\|} \right) \right\rceil \end{aligned} \qquad (6)$$

We can see from Ineq. 6 that the number of iterations required, $t_{\max}$, depends on two main factors: the connectivity of the cooperation graph G (as reflected by its second eigenvalue μ_2) and $\|X(0) - X^*\|$, the *distance* from the initial state $X(0)$ to the desired goal X^*. The first factor is only dependent on G, and is independent of the desired shape X^*, and vice versa for the second factor. We discuss these two factors in the following two subsections.

Scalability and Topology

Assuming that the initial distance from the desired goal is fixed, the number of iterations depends on μ_2. We show in Section 5 that $\mu_2 = 1 - \alpha\lambda_2$ where λ_2 is the the second eigenvalue of the graph Laplacian. This second eigenvalue has a special significance in graph theory and is called the *algebraic connectivity* because it encodes how well the graph is connected. While algebraic connectivity has been studied extensively in graph theory, its use in understanding decentralized algorithms is relatively new. Here we show that there are several important bounds on λ_2 that will provide us with a concrete understanding of how the algorithm scales as we scale up the size of the multi-agent system.

Theorem 2 (Mohar, 1991[5]). *Let L be the graph Laplacian of G. Then the second eigenvalue λ_2 of L satisfies*

$$\lambda_2 \geq \frac{4}{n \cdot diam(G)}$$

where n and $D = diam(G)$ are the number of agents and diameter of cooperation graph G respectively. Note that this theorem implicitly provides an upper bound on μ_2, which gives us the *worst case* convergence rate.

$$\mu_2 \leq \mu_2^+ = 1 - \frac{4\alpha}{n \cdot D} \qquad (7)$$

For example, assume that the multi-agent system consists of n agents connected in a line, e.g. Figure 3 (b). The worst-case λ_2 is bounded by $O(1/n^2)$. However, in reality, the convergence rate can be much superior than this upper bound. Figure 4 (a) shows the case when the multi-agent topology is an n by n grid where n is varied from 2 to 15. We fix $Y(0) = 50\epsilon$ to only examine the effect of topology. The upper curve shows t_{bnd}, the estimated $t_{\max}$ of Ineq. 6 with μ_2^+, the middle curve shows t_{est}, the estimated $t_{\max}$ with real μ_2, and t_{sim} is the average number of iterations in simulation computed from 1000 different initializations of $X(0)$ for each topology. We can see that although the worst case $t_{\max}$ is quite high, the convergence time in reality can be much faster.

Note that there are many forms of upper and lower bounds on λ_2 that can provide estimates of both worst and best case convergence time (see supplementary information in [1]). Also one can directly compute μ_2 for a given topology, providing a more precise prediction on convergence time.

Effect of Shape

As we can see from Ineq. 6, a bound on the effect of shape on convergence is given by the *euclidean distance* of the final state from the initial state. We can see that $t_{\max}$ increases *logarithmically* with $\|Y(0)\|$, which implies time changes slowly as deviation $\|Y(0)\|$ increases. The following result also allows us to bound the worst-case number of iterations for *any* shape:

Theorem 3. *Let the system's convergence rate be μ_2. Assume $x_i(t) \in R^+ \; \forall i, t$. Let $C = \sum_i x_i(0)$. Then the number of iterations required to achieve ϵ-approximation:*

$$t_{\max} = \left\lceil \log_{\mu_2} \left(\frac{\epsilon}{\sqrt{2} \cdot C} \right) \right\rceil$$

Proof. Expanding out the definition of 2-norm, we have:

$$||Y(0)||^2 = \left(\sum_i x_i^2(0) \right) + \left(\sum_i (x_i^*)^2 \right) - 2 \sum_i x_i(0) x_i^*.$$

Since $x_i(t) > 0$ for all t,[3] both $x_i(0)$ and x_i^* must be non-negative. Thus $\sum_i x_i(0)x_i^*$ is non-negative. Similarly, $\sum_i x_i^2(t) \leq \left(\sum_i x_i(t)\right)^2$ for all t. Hence:

$$||Y(0)||^2 \leq \left(\sum_i x_i(0) \right)^2 + \left(\sum_i x_i^* \right)^2.$$

Since A is stochastic and $\sum_i \tilde{b}_i = 0$, the sum of all agents' states is conserved. Therefore, $\sum_i x_i^* = \sum_i x_i(0) = C$, and we have $||Y(0)||^2 \leq 2 \cdot C^2$. Taking square-roots of both sides yields the result. $\square$

This indicates that if the agents' *connection topology* and *initial states* are known, we can calculate the number of iterations *guaranteed* to achieve ϵ-approximation of *any* desired shape.

Reactivity

Another important criteria is how the networked multi-agent system reacts to perturbations. From Ineq. 6, we can see that if $\|Y(0)\|$ is small, then only a few iterations will be sufficient to achieve ϵ-approximation. Furthermore, even as the number of agents increases, the number of iterations remains low.

Figure 4 (b) and (c) show the system's reactivity for two types of topologies: (b) square grid topologies and (c) random connected topologies that have the same number of edges and nodes as the corresponding grid graphs, but the connections between vertices are randomly generated.[4] Red lines indicate larger perturbations ($Y(0) = 250\epsilon, 500\epsilon, 1000\epsilon$)

[3]If the agents' operation range is $x_i(t) \geq x_{\min}$, $x_{\min}$ finite, we can shift the coordinates to $\tilde{x}_i(t) \geq 0$ and $C = \sum_i \tilde{x}_i(0)$.
[4]We remove the graph with ≥ 2 connected components and restart the process until the generated graph is connected.

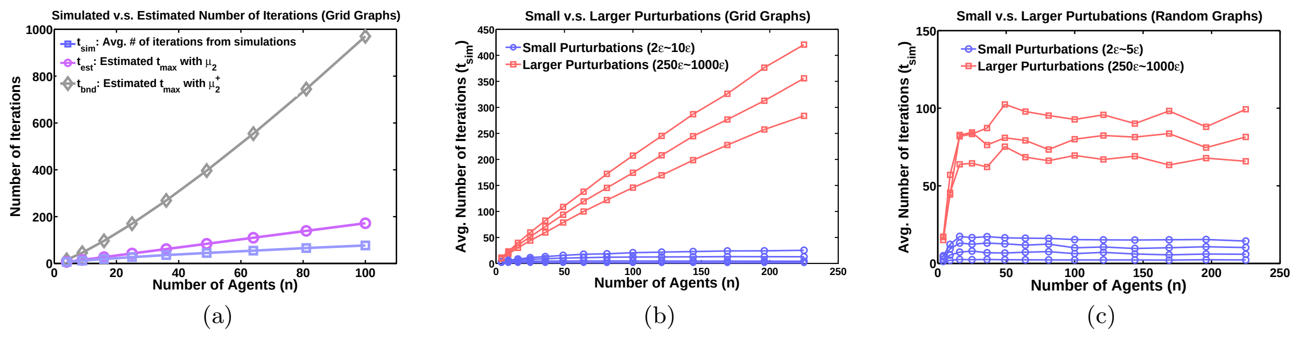

Figure 4: (a) Number of modules vs average number of iterations in simulations (t_{sim}) and estimated Bounds on number of iterations (t_{est} and t_{bnd}). (b) Comparison of t_{sim} between small and large perturbations in grid graphs. (c) Comparison of t_{sim} in randomly generated connected graphs.

and blue lines indicate smaller perturbations ($Y(0) = 2\epsilon$, 5ϵ, 7ϵ, and 10ϵ). Each point on Figure 4 (b) represents mean number of simulation rounds required of 1000 random initial $X(0)$ that satisfies the given $Y(0)$. For Figure 4 (c), each point represents the mean from 20 random topologies and 500 different initializations.

We can see from the figures that the networked multi-agent system's reactivities toward small perturbations (blue lines) scale well with the number of agents in both *regular* and *irregular* topologies. If the environment changes smoothly, then even large changes will appear like small perturbations over time. This shows why the algorithm performs particulary well in adaptation tasks like the self-adaptive structures shown in [13].

7. GENERALIZED FEEDBACK ALGORITHM

In many cases, the mapping f between sensor space and agent states is not precisely known. For example, when the distances between agents are unknown or have changed over time, the mapping described in the previous subsection cannot be computed.

Even in this situation, it may be possible to create an agent control law that uses the sensor feedback to directly control the agent state. To see why, we present a slightly generalized form of the feedback. Let θ be sensor measurement and θ^* be desired sensor reading, and $g(\theta, \theta^*)$ be *any* function such that the following conditions hold:

1. $g(\theta, \theta^*) = 0 \Leftrightarrow \theta = \theta^*$.
2. $\text{sign}(g(\theta, \theta^*)) = \text{sign}(f(\theta) - f(\theta^*))$.
3. $g(-\theta, -\theta^*) = -g(\theta, \theta^*)$.

Intuitively, condition 1 means that g only "thinks" the system is solved when it actually is; condition 2 means that when not solved, each sensor measurement at least points the agent at the correct *direction* to satisfy the local constraint; and condition 3 means that g is *anti-symmetric*

DEFINITION: **Generalized Feedback Control Law**

$$x_i(t+1) = x_i(t) + \sum_{a_j \in N_i} \alpha \cdot g(\theta_{ij}(t), \theta^*_{ij}). \qquad (8)$$

The second control law presented in [13], in which $g(\theta, \theta^*) = \theta - \theta^*$, belongs to this class of control law. In this case, Eq. 8 becomes:

$$x_i(t+1) = x_i(t) + \sum_{a_j \in N_i} \alpha \cdot (\theta_{ij}(t) - \theta^*_{ij}).$$

This means that we are performing biased local averaging directly on the tilt angles between the modules. (This may be a very natural situation, since it is often easy to supply robots with tilt or directional sensors even when it is difficult to measure and control inter-robot distances) As shown in Figure 3 (b), the actual mapping is $f(\theta_{ij}(t)) = d_{ij}\tan(\theta_{ij}(t))$. This choice of g satisfies the three properties required above, since: (1) tilt angle: $\theta_{ij}(t) = \theta^*_{ij} \Leftrightarrow \theta_{ij}(t) = \theta^*_{ij}$. (2) $\text{sign}(\theta_{ij}(t) - \theta^*_{ij}) = \text{sign}(d_{ij}\tan(\theta_{ij}(t)) - d_{ij}\tan(\theta^*_{ij})))$, $0 \le \theta_{ij}(t), \theta^*_{ij} \le \pi/2$. (3) $-\theta_{ij}(t) + \theta^*_{ij} = -(\theta_{ij}(t) - \theta^*_{ij})$.

To further compare the difference between the control law with generalized feedback and the control law in Eq. 3, let

$$\phi_{ij}(t) = \alpha \frac{g(\theta_{ij}(t), \theta^*_{ij})}{x_j(t) - x_i(t) - \Delta^*_{ij}}.$$

Then we can express Eq. 8 as

$$x_i(t+1) = x_i(t) + \sum_{a_j \in N_i} \phi_{ij}(t)(x_j(t) - x_i(t) - \Delta^*_{ij}). \qquad (9)$$

Comparing Eq. 8 with Eq. 3, the agent a_i's *step size* towards the feedback from a_j is now a *state dependent variable* $\phi_{ij}(t)$ instead of a constant factor α.[5] In other words this variable varies with the changing state of the agents. The relationship between the step size and the actual agent state differences may be non-linear. For example, in the case of the choice of feedback function $g(\theta, \theta^*) = \theta - \theta^*$ and $\theta^* = 0$,

$$\phi_{ij}(t) = \alpha \tan^{-1}\left(\frac{x_j(t) - x_i(t)}{d_{ij}}\right) / (x_j(t) - x_i(t))$$

which is clearly non-linear (as shown in Figure 5).

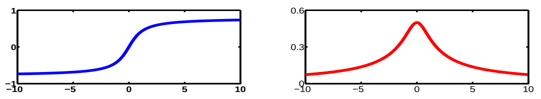

Figure 5: Left: $x_j(t) - x_i(t)$ (x axis) vs $\theta_{ij}(t)$ (y axis). Right: $\phi_{ij}(t)$ (y axis) varies with different $x_j(t) - x_i(t)$.

[5]unless $g(\theta_{ij}(t), \theta^*_{ij})$ is linear in $x_j(t) - x_i(t) - \Delta^*_{ij}$

Collective Dynamics: We can rewrite the system dynamics as

$$X(t+1) = A(t) \cdot X(t) - \tilde{b}(t) \tag{10}$$

where $A(t) = [\mathbf{a}_{ij}(t)]$ is an $n \times n$ matrix with element $\mathbf{a}_{ij}(t)$ defined by:

$$\mathbf{a}_{ij}(t) = \begin{cases} \phi_{ij}(t) & \text{if } a_j \in N_i \text{ and } i \neq j \\ 1 - \sum_{a_j \in N_i} \phi_{ij}(t) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

and $\tilde{b}_i(t) = \sum_{a_j \in N_i} \phi_{ij}(t) \Delta^*_{ij}$. We note that since $\phi_{ij}(t)$ is a state-dependent (and thus time-varying) variable, so $A(t)$ is also state-dependent.

Convergence Property: This control law has the same equilibria as the rule defined earlier (Eq. 3). The style of convergence proof we used in the original case required the generating matrix $A(t)$ to be row stochastic and symmetric. That is, for all t: (1) $\mathbf{a}_{ij}(t) \geq 0$ for $i \neq j$; (2) $\mathbf{a}_{ii}(t) \geq 0$ for all i; and (3) $\mathbf{a}_{ij}(t) = \mathbf{a}_{ji}(t)$ for all i, j. Because of the properties of the function g, $A(t)$ is actually row stochastic and symmetric for all t. To see why, notice that for *off-diagonal* elements, $\mathbf{a}_{i \neq j}(t) = \phi_{i \neq j}(t)$, for all a_i, and $a_j \in N_i$. Because of condition 2 on g, the numerator in the definition of ϕ_{ij} (which is g itself) always has the same sign as the denominator, so their ratio is non-negative. For the *diagonal* elements, notice that $\mathbf{a}_{ii}(t) = 1 - \sum_j \phi_{ij}(t) \geq 0$ for all i. But this can always be ensured as long as α is chosen small enough. Finally, since $g(-x, -y) = -g(x, y)$, we have $\phi_{ij}(t) = \phi_{ji}(t)$ so the symmetry condition is ensured.[6]

The guarantee of stochasticity and symmetry allows us to prove the following:

THEOREM 4 (GENERALIZED FEEDBACK). *Let X^* be the desired shape state that $x^*_j - x^*_i = \Delta^*_{ij}$. The collective dynamics defined by Eq. 8 will ensure that $X(t)$ converges to X^* for all initial conditions with convergence rate at least:*

$$\mu^*_2 = \max_t \mu_2(A(t)).$$

PROOF. see Appendix. □

Intuitively, what this result says is that the analog of original result holds, except that the convergence rate guarantee becomes somewhat looser, as we now have to maximize $\mu_2(A(t))$ over all t. The analog of Theorem 2 can also be shown:

THEOREM 5 (GENERALIZED FEEDBACK). *Let $\phi_{\min}$ be $\min_{i,j,t} \phi_{ij}(t)$ and μ^*_2 be the upper bound on the convergence rate. Then*

$$\mu^*_2 \leq 1 - \frac{4 \cdot \phi_{\min}}{n \cdot D}$$

PROOF. see Appendix. □

[6]We note that the assumption that $A(t)$ is symmetric (condition 3) can be relaxed, but the upper bound on convergence rate is less tight. The proof of this case is based on the theory of nonhomogenous stochastic matrix products [11], the product $A(t) \cdots A(2)A(1)$ will converge to a rank 1 matrix with exponential rate. The recent result in [3] explicitly determines an upper bound on convergence rate.

8. DECENTRALIZED VS CENTRALIZED

An important question in networked multi-agent systems is whether to use a *decentralized approach*, such as the one described here where agents iteratively communicate and react to arrive at a solution, or to use a *centralized tree-based* approach where a root agent collects all the information from other agents. This question is not only relevant to modular robots, but also to robot swarms and sensor networks. It also applies to many problems from shape formation to time synchronization. Using our results, we can describe the tradeoffs between these two approaches.

For the centralized algorithm, we assume that a root agent collects all the information from *all* other agents using a spanning tree, computes a final state for every agent, and then disseminates the results back to them. This results in two costs: (a) a communication cost of collecting/disseminating information and (b) a computation cost for the root node. In most homogenous multi-agent systems, each agent has fixed communication and computation power. For the kinds of tasks we consider here, communication is often a more severe bottleneck: if an agent can only collect a constant amount of information per unit time, then the time to collect all the agents's states is $O(n)$ (n: number of agents) and not $O(\mathrm{D})$ (D: diameter). This cost is paid for every shape change, regardless of the distance between the initial and desired states. This results in poor performance in the case of small perturbations, where information must travel all the way to the root agent before it can be resolved.

In contrast, the communication cost of the decentralized algorithm described here is $O(t)$ (t: iteration number) which depends on both topology and distance from goal. The relationship between topology and performance in some cases is worse than the centralized case. However, if the distance from goal is small – e.g., a small perturbation or slowly changing environment – then the system reacts rapidly in only a few iterations even when n is large (Figure 4 (b)(c)).

This suggests that for consensus-like problems while decentralized algorithms may pay a significant start-up cost to achieve a steady state, they are extremely reactive to perturbations. Thus they are more appropriate when the goal is to "maintain" constraints over long periods of time under uncertain and changing conditions, rather than produce a solution once. Finally, they are more robust, and less complex to implement, in situations where agent errors and topology changes are common.

9. CONCLUSIONS

We have presented and analyzed a class of distributed sensing-based shape formation algorithms. We prove the convergence properties and characterize how various factors, such as the topology of the agent cooperation graph, are related to the performance of the algorithms. In comparison with a centralized approach, this approach has a strong advantage in robustness and reactivity. This framework can be applied to many other multi-agent problems, e.g. a team of mobile robots, where the goal of the agents are expressed as inter-agent relationships and agents must constantly and quickly adapt to the uncertainties of the environment.

Acknowledgement

We thank R. Olfati-Saber and D. Yamins for their helpful comments. This work is funded by NSF Grant No. 0523676.

10. REFERENCES

[1] http://www.eecs.harvard.edu/∼chyu/aamas08-appdx.pdf.

[2] H. Bojinov, A. Casal, and T. Hogg. Emergent structures in modular self-reconfigurable robots. In *Proc. of ICRA*, 2000.

[3] M. Cao, A. S. Morse, and B. D. O. Anderson. Reaching a consensus in a dynamically changing environment: A graphical approach. *To appear in SIAM Journal on Control and Optimization.*

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

[5] B. Mohar. Eigenvalues, diameter, and mean distance in graphs. *Graphs and Combinatorics*, 7:53–64, 1991.

[6] B. Mohar. The laplacian spectrum of graphs. *Graph Theory, Combinatorics, and Applications*, 2, 1991.

[7] S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji. M-tran: Self-reconfigurable modular robotic system. *IEEE/ASME Trans. Mechatron*, 7(4):431–441, 2002.

[8] R. Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Trans. on Automatic Control*, 51(3):401–420, 2006.

[9] R. Olfati-Saber, J. Fax, and R. Murray. Consensus and cooperation in networked multi-agent systems. In *Proc. of IEEE*, 2007.

[10] D. Rus, Z. Butler, K. Kotay, and M. Vona. Self-reconfiguring robots. *Communications of the ACM*, 45(3):39–45, 2002.

[11] E. Seneta. *Non-negative Matrices and Markov Chains.* Springer Verlag, 1981.

[12] W.-M. Shen, M. Krivokon, H. Chiu, J. Everist, M. Rubenstein, and J. Venkatesh. Multimode locomotion for reconfigurable robots. *Autonomous Robots*, 20(2):165–177, 2006.

[13] C.-H. Yu, F.-X. Willems, D. Ingber, and R. Nagpal. Self-organization of environmentally-adaptive shapes on a modular robot. In *Proc. of IROS*, 2007.

APPENDIX

Proof of Theorem 1

We first show that analyzing Eq. 10 is equivalent to analyzing a linear dynamical system without the bias vector. The optimality condition:

$$X^* = A \cdot X^* + \tilde{b} \tag{12}$$

can be rewritten as: $\alpha L \cdot X^* = \tilde{b}$. We use the graph Laplacian property [6] that when G is connected, $\mathrm{rank}(L) = n-1$ with $\mathrm{null}(L) = \mathbf{1}$. We can add an additional constraint to the system based on *conservation* property of the agent state: $\sum_i x_i(0) = \sum_i x_i^* = C$ (since A is a row stochastic matrix, and $\sum_i \tilde{b}_i = 0$). The new linear system with the additional constraint becomes $\alpha L' \cdot X^* = \tilde{b}'$. Since the new constraint lies in the null space of L, $\mathrm{rank}(L') = n$ and there exists a unique X^* for every initial condition $X(0)$. We subtract the optimality condition from Eq. 4:

$$Y(t+1) = A \cdot Y(t) \tag{13}$$

where $Y(t) = X(t) - X^*$. The following proof follows the procedure of [9]. Since L is a real symmetric matrix, the well-known Courant-Fischer Theorem yields:

$$\lambda_2(L) = \min_{\|x\|=1, x\perp\mathbf{1}} \frac{x^T L x}{x^T x}$$

As $Y(t)^T \cdot \mathbf{1} = 0$ for all t and the relationship between λ_2 and μ_2, we can utilize the results from [9] and show that:

$$\max_{Y(t)} \frac{Y(t)^T A Y(t)}{Y(t)^T Y(t)} = \mu_2(A) < 1 \tag{14}$$

$\mu_2(A)$ denotes the 2nd largest eigenvalue of A. Define a Lyapunov function $V(t) = \sum_i (x_i(t) - x_i^*)^2 = Y(t)^T Y(t)$. Now, following Eq. 14, we get:

$$\begin{aligned} V(t+1) &= (AY(t))^T (AY(t)) = Y(t)^T A^2 Y(t) \\ &< (\mu_2(A))^2 Y(t)^T Y(t) = (\mu_2(A))^2 V(t) \end{aligned} \tag{15}$$

Thus, $Y(t)$ converges to zero ($X(t)$ converges to X^*) with exponential rate at least $\mu_2(A) < 1$.

Proof of Theorem 4

The $A(t)$ matrix in Eq. 10 can be written as: $I - L_w(t)$ where $L_w(t)$ is a *weighted* graph Laplacian matrix. The properties of the weighted graph Lalpacian are similar to those of the standard Laplacian [6]. When G is connected, $\mathrm{rank}(L_w(t)) = n-1$ with $\mathrm{null}(L_w(t)) = \mathbf{1}$. Since $\sum_i \tilde{b}(t) = 0$ and $A(t)$ is stochastic for all t, the agent state conservation constraint still applies. We can solve X^* with the same procedure as the proof of Theorem 1 w.r.t. a particular $A(t)$. We note that the obtained X^* satisfies $x_j^* - x_i^* = \Delta_{ij}^*$ for all $a_i, a_j \in N_i$, the optimality condition $X^* = A(t)X^* + \tilde{b}(t)$ will hold for all t.[7] We can again obtain the new dynamics system:

$$Y(t+1) = A(t)Y(t)$$

Since $L_w(t)$ is a real symmetric matrix for all t. Applying the Courant-Fisher Theorem to the analagously-defined Lyapunov function, a similar derivation as Eq. 14 and 15 yields:

$$V(t+1) \le (\mu_2(A(t)))^2 V(t) \le (\max_t \mu_2(A(t)))^2 V(t)$$

Therefore, the convergence rate is at least the maximal value of the second largest eigenvalues among the $A(t)$.

Proof of Theorem 5

This is an extension to the original proof of [5]. Let v be the $L_w(t)$'s eigenvector associated with λ_2. Based on Courant-Fischer, we can get: $\sum_{a_i,a_j\in N_i} \phi_{ij}(t)(v_i - v_j)^2 = \lambda_2 \sum_i v_i^2$. In addition, $\sum_i v_i = 0$ (orthogonal to $\mathbf{1}$), we obtain: $2n \sum_{a_i,a_j\in N_i} \phi_{ij}(t)(v_i - v_j)^2 = \lambda_2 \sum_i \sum_j (v_i - v_j)^2$. In [5], it is shown that $\lambda_2 \sum_i \sum_j (v_i - v_j)^2 \le \lambda_2 \cdot diam(G) \cdot \frac{n^2}{2} \sum_{a_i,a_j\in N_i} (v_i - v_j)^2$. Since $\phi_{\min} \le \sum_{a_i,a_j\in N_i} \phi_{ij}(t)(v_i - v_j)^2 / \sum_{a_i,a_j\in N_i}(v_i - v_j)^2$, we reach the conclusion: $\lambda_2 \ge \frac{4\cdot\phi_{\min}}{n\cdot diam(G)}$.

[7]The i^{th} element of X^*: $x_i^* + \sum_{a_j\in N_i} \phi_{ij}(t)(x_j^* - x_i^*) - \sum_{a_j\in N_i} \phi_{ij}(t)\Delta_{ij}^* = x_i^*$