NORTHWESTERN UNIVERSITY

Multi-agent Coordination by Decentralized Estimation and Control

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Mechanical Engineering

By

Peng Yang

EVANSTON, ILLINOIS

December 2008

Acknowledgments

First of all I would like to thank my advisors Professor Randy A. Freeman and Professor Kevin M. Lynch for their careful guidance on my research project and intellectual development. I am very grateful for the research freedom I was given and more so for the necessary corrections along my road of inquiry. I feel fortunate to work closely at the same time with two advisors that have different expertise, style and interest. I also want to thank Professor Randall Berry, Professor J. Ed Colgate and Professor Adilson E. Motter for being on my committee; their generous feedback in various occasions and diverse background helped me gain a more complete understanding of my project. I also want to thank Professor Siddhartha Srinivasa for the fruitful collaboration and Professor Michael A. Peshkin for his valuable feedback during my LIMS presentations.

I would like to thank my previous and current LIMS labmates, especially Michael Taylor, David Weir, Tom Vose and Peng Pan. Their kindness makes the lab an enjoyable place to stay. I want to thank my Chinese friends here at Northwestern, especially Lan Ge, Junzhao Ma, Rui Qiao and Yao Zhao for coloring my extracurricular life.

I take the last chance to thank my parents for their unconditional love.

ABSTRACT

Multi-agent Coordination by Decentralized Estimation and Control

Peng Yang

This thesis contributes in two ways. First it describes a new framework for the systematic design of collective behaviors and solves a key stability issue under this design framework. In this thesis we apply this framework to solve three tasks in the swarm robotics field: connectivity maintenance, formation control and target tracking. Second, in the process of solving one problem, the connectivity maintenance task, we design a decentralized power iteration algorithm. Given any connected graph, this generic algorithm allows each agent to estimate its corresponding component of the Fiedler eigenvector, the eigenvector corresponding to the second smallest eigenvalue of the graph Laplacian [**45**]. The Fiedler eigenvector and eigenvalue have proved to be useful in many areas, including Google's pagerank system, graph segmentation algorithms and connectivity maintenance in mobile sensor networks.

Given a group of mobile agents, this thesis describes a framework for the systematic design of collective behaviors. The approach is based on decentralized simultaneous estimation and control, where each agent communicates with neighbors and estimates the

global performance properties of the swarm needed to make a local control decision. Steps of the approach include designing a control law with desired convergence properties, assuming each agent has perfect global knowledge; designing an estimator that allows each agent to make correct estimates of the global properties needed to implement the controller; and possibly modifying the controller to recover desired convergence properties when using the estimates of global performance. In addition, the performance of this design framework can be optimized by increasing the convergence speed of the estimator through tuning the communication weights.

We apply this framework to three different problems: (1) estimation and control of graph connectivity, (2) controlling the moment statistics describing the location and shape of a swarm, and (3) cooperative target localization. For the first task, we design a decentralized power iteration algorithm allowing each agent to estimate its component of the Fiedler eigenvector of the graph. For the last two tasks, we derive small-gain conditions which, if satisfied, guarantee that the system falls into a stable equilibrium set, even in the presence of a changing network topology and the addition and deletion of robots. We validate our approach through computer simulation and physical experiments.

Table of Contents

List of Figures

CHAPTER 1

Introduction and Thesis Overview

1.1. Introduction to Mobile Sensor Network

Recent advances in wireless communications and electronics have enabled the development of low-cost, low-power, multi-functional sensor nodes that are small in size and communicate untethered in short distances, e.g. the Berkeley Mote [**34**], the Intel Mote [**2**] and the BTnodes [**1**]. These tiny sensor nodes, consisting of sensing, data processing and communicating components, enable the idea of sensor networks. Sensor networks are envisioned to revolutionize our daily life by ubiquitously monitoring our environment and adjusting to suit our needs. Exemple applications include managing inventory, monitoring disaster areas and monitoring product quality. Among many design challenges, here are some from the controller design perspective [**3**], [**20**]:

Limited Resources: Sensor nodes are limited in power, computational capacity and memory. E.g.: The Berkeley Mote [**34**] has a 4 Mhz processor and 8k RAM, and powered by a single CR2450 battery. The system runs 30 hours at peak load, 200 hours at the idle mode and over a year at the inactive mode.

Highly Distributed: Such networks are pictured to be deployed in large numbers, and therefore it is important that all algorithms are scalable. This should include both the system layer services (localization, time synchronization, routing) and the application layer services. It is also important to make sure these distributed

algorithms are robust to individual agent failures/new agent coming in due to the nature of such sensor networks.

Deployment and Coverage: One of the initial challenges to use large sensor networks is deployment. Since the number of nodes being deployed is very high, it is difficult to carefully handplace each of the nodes. Current approach often requires dense deployment of them to guarantee the connectedness of the network in a probabilistic sense [**85**].

Network Dynamics: Variation in communication range, frequent death of nodes due to lack of energy and other environmental causes result in large variations in network topology. Accordingly, applications should be able robust to variations in network topology.

Controlled mobility enables a whole new set of possibilities in sensor networks. The ability of actively changing locations can be used to solve many design challenges outlined above [**20**], and adding new functionalities to make the sensor networks more adaptive and robust. With the mobility fully leveraged, we can imagine a *mobile sensor network* being able to self-deploy into the environment while cooperatively localizing themselves and building a map of the environment. Besides all the data-aggregation abilities of the static sensor network, such mobile sensor networks can adaptively monitor a dynamic environment by self-aggregating around areas of interest. With additional actuation devices, they can also perform cooperative transportation of an object and dynamically allocate tasks. At the same time, the system connectivity is being monitored continuously and the network self-reconfigures to maintain connectivity when some sensor nodes run out of energy or newly charged sensor nodes rejoin the network. The active research effort into the

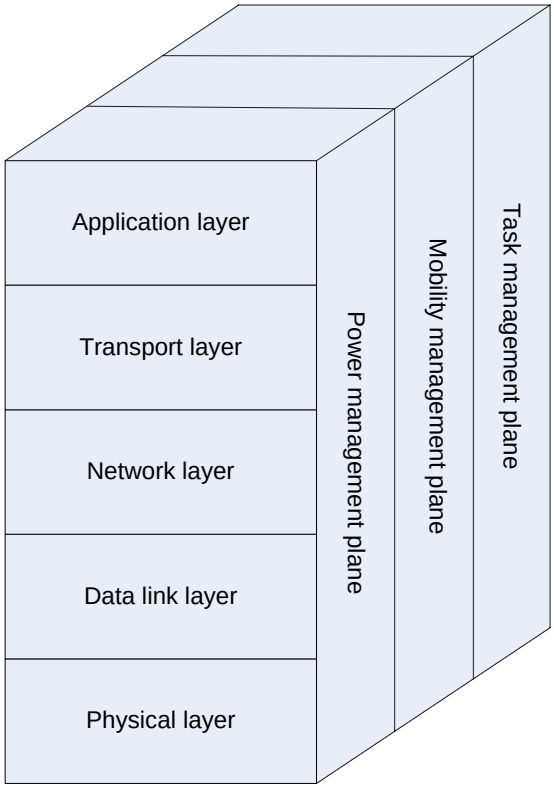


Figure 1.1. The sensor networks protocol stack proposed in [**3**].

existing mobile sensor networks, e.g. swarm-bots [**22**], robomote [**20**], Icosystem/Irobot swarm [**68**] and underwater glider networks [**58**], are turning this vision into reality.

Next we examine the mobile sensor networks in greater detail, specifically reviewing some common application layer tasks (Fig. 1.1) and current research effort on them. These tasks differ in nature: Some tasks, like maintaining connectivity, act like a low-level system service, and the corresponding solution should put constraints on the higher level tasks like deployment, coverage or localization. Our review focus is biased toward coordinated tasks, in contrast to individual tasks which each agent can solve alone without sensing or communicating with other agents.

1.1.1. Deployment and Coverage

One of the initial design challenges in mobile sensor networks is how to automatically deploy the sensors over the environment. As can be imagined, the availability of a precise environment map greatly affects the difficulty of the problem. When this information is known as a priori, *Voronoi partition* of the environment is used to develop a coverage metric and in the deployment phase each sensor moves to maximize the sum of the information gain from its own Voronoi partition and its neighbors [**18**], [**71**]. In environments of uniform interestingness, this results in an even distribution of the agents. In environments with hot spots of information, agents achieve a higher density in these regions. In fact, it is easy to conceive a family of algorithms for deployment that optimize various metrics like coverage area [**18**], [**15**] or required connectivity [**61**]. In contrast, only heuristic algorithms exist when the environment map is dynamic (e.g. a search and rescue task) or totally unknown (e.g. exploring a hostile or unknown environment). One way to spread out the sensors is to design simple inter-agent and agent-obstacle repulsive potential fields [**36**]. Another approach deploys the sensors sequentially: each next-in-line sensor calculates the optimal position to maximize the existing coverage measure while maintaining line of sight visibility constraint and move to the calculated spot [**37**]. A good heuristic for the art gallery deployment problem is also proposed, where each agent moves towards the furthest vertex in its Voronoi partition of the overall visibility polygon [**37**].

1.1.2. Maintain Connectivity

Usually it is desirable for a mobile sensor network to achieve a task while maintaining the overall network connectivity. One appropriate connectivity measure is the second smallest

eigenvalue λ_2 of its Laplacian matrix, $\lambda_2 > 0$ being necessary and sufficient for connectivity [**32**]. There is another measure built on the following fact: Given A as the adjacency matrix of an undirected and unweighted graph, the (i, j)-th entry of A^k is the number of paths of length k for node i to node j. The matrix $C_k(A) = I + A + \cdots + A^k$ can be used to measure the *k-hop connectivity* of the graph [**87**]: Every entry of $C_k(A) > 0$ if every pair of nodes is connected with a path of length no longer than k. This gives a nice scalable measure: the simple and restrictive $C_1(A)$ can be computed locally, and to the other end $C_{n-1}(A)$ measures connectivity in the usual sense. With a measure of connectivity, it is natural to enforce the connectivity constraint by deriving the motion constraints for each agent using the Control Lyapunov Function approach [**87**]. For each agent, the motion constraint in general needs central computation and filters the control effort given by the original motion planner. The simplest way to have a distributed implementation is to use a restrictive approach: designing a motion controller while maintaining every existing communication link [**51**], [**43**].

1.1.3. Collaborative Localization and Mapping

Due to the inherent odometer error and other sensor measurement noises, self localization and environmental mapping are fundamental estimation problems for mobile robots. Besides devising separate estimation procedures [**28**], the SLAM (Simultaneous Localization and Mapping) approach aims to solve these two problems at the same time [**21**]. For mobile sensor networks, the multi-robot SLAM (Simultaneous Localization and Mapping) problem offers new challenges and opportunities. Existing centralized solutions use Extended Kalman Filters or Extended Information Filters, and the computational efficiency

can be improved by using sparse matrix computations [**77**] or by limiting the frequency with which each robot exchanges its local map information [**83**]. An alternative to the EKF approach is the sample-based approach, e.g. using particle filters [**35**]. The EKF approach remains to be decentralized and more importantly it doesn't take advantage of the new multi-agent scenario. For example, there is a possibility of better localization abilities in multi-agent systems, where agents can localize themselves not only by measuring distances to the landmarks [**28**], but also by measuring distances to its nearby agents and maintaining motion models of them [**69**]. The current EKF approach is *passive*, meaning there is no controlled motion during the SLAM estimation procedure. Instead, motion controllers can be designed to improve the performance of the estimation procedures [**28**].

1.1.4. Formation Control

Formation control is one of the most-studied problems in decentralized control due to its wide range of applicability. For a given task, for simplicity it is often desirable to decouple the problem into trajectory design and formation control [**58**]. For example to collaboratively monitor an environment, centralized algorithms are developed to move mobile sensors in a fixed shape along the estimated gradient [**52**] or contour plot [**89**] of the environment (although the environmental monitoring problem can also be formulated as a coverage problem where the goal is to spread the sensors over the environment). Similarly in Target Tracking problems, a common setup is to deploy the sensors to maximize some measure of the information gain. Sometimes the resulting optimal/suboptimal sensor configurations can be calculated beforehand, and it can used to ease the motion controller design [**71**, **4**, **89**, **52**].

1.2. Introduction to Consensus Algorithms

Our work is built directly on consensus algorithms. These algorithms are distributed estimation procedures that each agent can use to iteratively calculate the aggregated information of the network. The explicit investigation of these consensus algorithms dates back from the 1980s [**7**, **6**].

One well-known consensus algorithm was thoroughly investigated by Olfati-Saber and Murray [**55**], who studied the convergence of the differential equation

$$\dot{x} = -Lx\,,$$

where x is a decision variable (e.g., an estimator state in our framework) and L is the *Laplacian* of a communication graph of agents (see Section 2.2.1). They showed that each agent's decision variable x_i converges to a common value, and that this common value is the average of the initial values $x_i(0)$ if the communication graph is balanced, i.e., for each node in the graph, the in-degree and out-degree are equal. Average consensus is reached even with switching network topologies and communication delays bounded by a function of the largest eigenvalue of L. The rate of convergence of the consensus estimator as a function of the network topology and weights, as well as heuristic and analytic approaches to choosing network topologies and weights to optimize convergence rates, are studied in [**84**, **54**, **86**, **47**, **49**]. See [**8**, **66**]and [**24**] for extensions to the cases of uniform and non-uniform communication delays and probabilistic dropping of packets.

Consensus problems arise in a variety of engineering tasks (load balancing, clock synchronization, data aggregation in sensor networks). Therefore, other solutions are proposed from different perspectives around the theme of distributed iteration. The Consensus Propagation method [**44**] is an asynchronous distributed averaging method that has guaranteed convergence for a fixed, tree topology, with convergence time as small as the diameter of the graph. Taking advantage of the tree topology, for each node the network can be decomposed into several non-overlapping subgroups based on its neighbors. Each node maintains an estimate of cardinality and average of all its subgroups, and iterate to refine the estimate. In case of a general graph with cycles, additional parameters need to be introduced and tuned to give good heuristic performance. The algorithm [**57**], which is used to solve the load balancing problem, is also similar to the consensus algorithm [**55**]. The only difference is, at each time the distributed averaging is done only along one communication channel which bears the biggest node value differences among all channels in one neighborhood. This modification will slow down the convergence speed compared to the consensus algorithm, but is simpler for digital implementations. Lastly it is worth mentioning the consensus algorithms belong to a much broader class of problems, synchronization using nonlinear coupled-oscillators [**41**].

We refer to this average consensus problem as static, as it incorporates only the initial data $x(0)$. To build an estimator capable of tracking the average of changing inputs, a high-pass *dynamic consensus estimator* was proposed in [**73**, **79**]. In this thesis we develop dynamic consensus estimators that have improved noise rejection properties and steady-state performance in the face of the addition and deletion of agents were introduced

in. Dynamic consensus filters have also been utilized to construct decentralized Kalman filters [**53**].

The true power of the consensus algorithm lies in its robustness to dynamic topologies, which is inherent in sensor network applications [**3**]. Compared to a tree based method to collect data, this robustness property is obtained at a much higher communication cost. In fact, it was recently shown that the consensus algorithm can be accomplished in finite time in case of a fixed topology [**76**].

In the present work we use average consensus estimators to explicitly estimate global properties of the system. These estimates are then used in the control law. Much past work has instead focused on using average consensus algorithms directly as the memoryless control law K. Typically agents sense the states of their neighbors and choose controls based on the average of these states. Such an approach can be applied to the problems of bringing agents to a common meeting point, or rendezvous [**40**, **17**],where and to various types of flocking and formation control [**38**,**65**,**42**]. Extensions of the basic idea to agents with second-order dynamics can be found in [**39**,**64**]. Many of these algorithms are robust to a broad class of switching network topologies [**38**, **46**].

There are other types of consensus as well, for example in minimal-time rendezvous [**50**] where all robots reach consensus and move to the center of their minimal enclosing ball of their starting configurations.

1.3. Thesis Overview

We are studying the following general problem: given a set of mobile agents, design a control law to run on each agent, based on sensor and communication input, to achieve

a desired collective "emergent" global behavior of the system. In other words, the global dynamical system defined by the interaction of the many individual agents' control laws should have the desired collective behavior as an attractor, preferably a global attractor. The performance of the system is judged by the global behavior of the system—it must be evaluated over all the agents.

The key constraints are that each agent has limited sensing, computation, motion, and communication capabilities, and may have significant dynamics. The performance of the group should improve or degrade gracefully as agents are added or deleted; in other words, the approach should be scalable, robust, and require no central controller.

While intelligent collective behavior can emerge even when each agent is ignorant of global properties of the system, as demonstrated in a number of models of schooling and flocking [**67**, **70**, **19**, **59**], there are few tools to guide the design of local control laws to achieve a particular desired collective behavior. Only a limited class of tasks can be solved by reactive controllers using immediate sensory and communication input from neighbors, and some of these require that the agents be able to implement a sensing or communication network of a particular structure. Instead, we are pursuing a design framework for a broader class of tasks, with changing numbers of agents and few requirements on the time-varying communication network. We begin with the desired behavior encoded in a global cost functional J. We then equip each agent with (a) a local controller that chooses an action to minimize the cost based on knowledge of the global performance of the group, and (b) an estimator that provides estimates of the global properties of the group needed to implement the controller. This estimator uses sensory data and information communicated by other agents.

To ensure desired convergence properties when using estimates, we can modify the controller according to the properties of the estimator. For example, we can enforce a small-gain condition or a time-scale separation of the estimator and controller dynamics. These conditions can typically be written as bounds on motion control gains as a function of the estimator's communication network structure. Deriving such conditions is one of the main goals of our work. In Chapter 2, for example, we provide small-gain conditions that express bounds on the aggressiveness of the motion controls as a function of the communication graph Laplacian.

The estimate-and-control approach can be applied to both motion coordination tasks and mobile sensing tasks. This is illustrated by the examples of *formation control*, where the cost function is in terms of the positions of the agents, and *cooperative target localization*, where the cost function is in terms of the uncertainty in the agents' estimates of the moving target. In the formation control problem, the agents' global estimates serve the motion coordination problem; in the target localization problem, motion control serves to improve the agents' estimates.

Example 1 (Formation Control)**.** *We can encode an abstraction of a robot formation using a finite number of geometric moments [**5**]. If the position of agent i in a plane is denoted $p_i = [p_{ix}\ p_{iy}]^T$, then a formation of n robots can be abstracted to an ℓ-vector of moments of the form*

$$f(p) = \frac{1}{n}\sum_{i=1}^{n}[p_{ix}\ \ p_{iy}\ \ p_{ix}^2\ \ p_{iy}^2\ \ p_{ix}p_{iy}\ \ p_{ix}^3\ \ \ldots]^T\,,$$

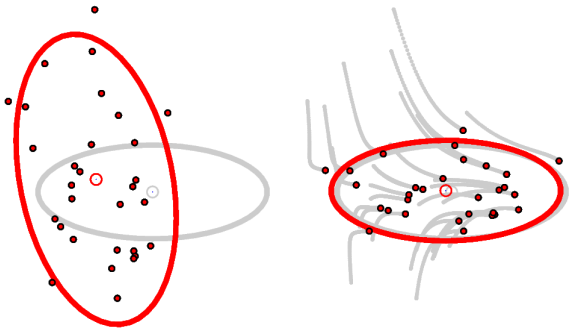


Figure 1.2. (Left) The initial configuration of a swarm, a uniform-density ellipse with the same mass and first- and second- order moments as the swarm, and the goal formation of the swarm represented as another uniform-density ellipse. (Right) The swarm converges to a configuration having the desired first- and second- order moment statistics.

where $p = [p_1^T \dots p_n^T]^T$. *To achieve a desired vector of moments* f^*, *which may be commanded by an exogenous input or derived from environmental variables, we can define a cost function*

$$J = [f(p) - f^*]^T \Gamma [f(p) - f^*] \,,$$

where Γ *is a positive-definite weighting matrix. This is the primary example of the thesis, studied in detail in Section 5 (see Figure 1.2).*

Our primary interest is in large swarms where it is not necessary to specify the exact position of each robot. Instead, the swarm formation can be described by a set of summary moment statistics that form a basis for the space of all formations. These statistics have the property that low-order moments capture much of the essential shape of the swarm, but progressively higher-order moments can be specified until only a single formation is consistent with the statistics. Low-order moments then provide a convenient abstraction of the total swarm formation, allowing, for example, high-level human interaction with a large number of robots. We show that the estimate-and-control approach provides

guarantees on the convergence of the swarm to desired formation statistics in the face of changing communication networks and the addition and deletion of agents.

Example 2 (Cooperative Target Localization)**.** *A group of agents cooperatively tracks the location of a target in the environment. Each agent takes noisy sensor range and bearing readings of the target and maintains an estimate of the target location. Each agent shares its estimate with its neighbors to reach consensus on a belief distribution of the location of the target, represented by a mean and a covariance matrix P. The noise in the information gathered by each agent's sensor is dependent on the relative location of the target, so each agent moves to maximize the expected new sensor information relative to the current uncertainty P. The cost function can be expressed as*

$$J = \det(P)\,.$$

1.4. Summary of the Thesis

Before proceeding with the main exposition, we summarize here the specific technical contribution of the thesis.

Chapter 2: In Section 2.1 we propose a new framework to design algorithms for multi-agent coordination tasks. Each agent is equipped with a controller and estimator in a feedback loop. A systematic design procedure is broken down to four steps: Encoding the desired behavior in a cost function, designing a initial controller, designing a distributed estimator and modifying the controller to guarantee the stability of the feedback system. Then two types of distributed estimators that can dynamically track the average of all sensor readings, the *proportional consensus estimator (P estimator)* and the

proportional-integral consensus estimator (PI estimator), are introduced in Section 2.2. The convergence properties of the two estimators are also characterized. In Section 2.3 we develop a systematic design procedure for the simplest class of problems within this framework. Compared to a centralized design approach, the system with this distributed controller converges to the same equilibrium set as in the centralized control.

Chapter 3: In general the consensus estimation algorithm is built on a weighted graph and this chapter looks at how to choose the weights to maximize the convergence speed of this estimation procedure. This optimization problem is solved in a centralized fashion in Section 3.1 and distributed heuristic weightings are given in Section 3.2. In Section 3.3 it is shown through simulations the *random-rewiring* mechanism on graphs can also speed up the estimation convergence speed by creating long range communication links among networks.

Chapter 4: This chapter studies how to maintain the connectivity of the communication graph in a decentralized way. The key component in our solution is a decentralized power iteration algorithm that estimates the eigenvector corresponding to the second smallest eigenvalue of a weighted Laplacian matrix. This eigenvector estimation procedure is then used to estimate the algebraic connectivity of a graph. We use this estimate in a decentralized controller that maintains the global connectivity of the graph over time.

Chapter 5: This chapter is devoted to the problem of controlling swarm formation by controlling its formation statistics. We solve it through our simultaneous estimation and control approach, and use both the P estimator and the PI estimator. In Section 5.3 and Section 5.4 separate nonlinear damping mechanisms are incorporated into the original

controller to guarantee the stability of the feedback system. The equilibrium set is examined in Section 5.2, and it shows every local maximum of the cost function is unstable. In Section 5.5 and 5.6 we validate our approaches through simulations. In Section 5.7 we describe our experimental testbed and the results from the experiment.

Chapter 6: This chapter studies how to use a mobile sensor network to track one or many moving targets. Initially it is assumed all sensors are of the same type and there is only one target to be tracked. After developing a performance measure for the tracking task in Section 6.1, two methods are proposed to solve the problem in Section 6.2 and Section 6.3 and they are validated through simulations in Section 6.4. Lastly, Section 6.5 extends the methods to heterogeneous sensors and multiple targets cases.

CHAPTER 2

Designing Coordination Algorithms: Decentralized Estimation and Control

2.1. A New Framework for Multi-agent Coordination

The system consists of a collection of n identical mobile agents, each implementing the same control system with no identifying tags or ID numbers. Agents may be added or deleted from the system at any time, i.e., n may change. The (physical) state of agent i is written as the vector p_i, its control is the vector u_i, and its state evolves according to the dynamics

$$\dot{p}_i = F(p_i, u_i, \mathcal{P}_i^{\text{phys}}), \tag{2.1}$$

where $\mathcal{P}_i^{\text{phys}}$ represents the set of states of any nearby agents that may physically impact the motion of agent i (e.g., through contact, fluid wake, hydrophobic effects, etc.). For the problems we consider, $\mathcal{P}_i^{\text{phys}}$ can be omitted from the motion dynamics, yielding $\dot{x}_i = F(p_i, u_i)$.

Each agent takes measurements of the form

$$z_i = C(p_i, \mathcal{P}_i^{\text{sens}}), \tag{2.2}$$

where $\mathcal{P}_i^{\text{sens}}$ represents the states of other agents that affect or contribute to the measurements. The vector z_i may measure absolute or relative positions and velocities of the agents, or environmental variables such as temperature, salinity, or locations of points of interest in the environment. This vector also includes any exogenous reference inputs. The dimension of z_i may change depending on the number of neighbors in the sensing network.

We are interested in the design of dynamic local state feedback controllers of the form[1]

$$u_i = K(p_i, z_i, \eta_i, y_i, \mathcal{S}_i) \tag{2.3}$$

$$s_i = G(p_i, z_i, \eta_i, y_i, \mathcal{S}_i) \tag{2.4}$$

$$\dot{\eta}_i = Q(p_i, z_i, \eta_i, y_i, \mathcal{S}_i) \tag{2.5}$$

$$y_i = R(p_i, z_i, \eta_i, \mathcal{S}_i)\,, \tag{2.6}$$

where s_i denotes the fixed-dimension signal vector agent i sends to its neighbors, $\mathcal{S}_i$ denotes the variable-dimension collection of signals agent i receives from its neighbors $\mathcal{N}_i$ through communication channels, the vector η_i is an estimator state containing information about the global performance of the system, and the vector y_i represents the output of the estimator (Figure 2.1). The formulation above expresses the estimator Q and R using continuous-time dynamics for simplicity, but sampled-data or hybrid dynamics could be substituted as appropriate.

Note that implicit in the formulation are three dynamic interaction networks: a network of physical interaction, represented by $\{\mathcal{P}_i^{\text{phys}}\}$; a network of sensing interaction,

[1]Not all of K, G, Q, R will necessarily use all parameters. Local state observers can be used to provide estimates $\hat{p}_i$ of p_i if required.

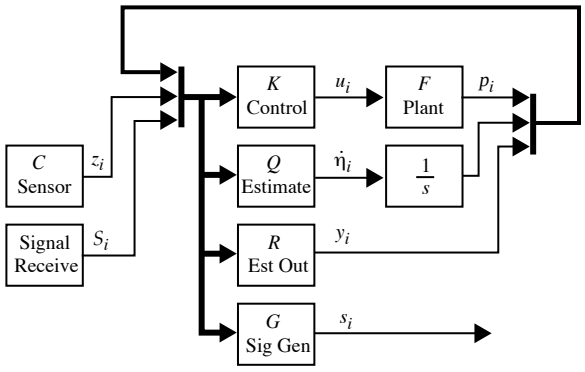


Figure 2.1. Block diagram for agent i.

represented by $\{\mathcal{P}_i^{\text{sens}}\}$; and a network of communication interaction, implicit in $\{\mathcal{S}_i\}$. These networks may be changing over time.

The goal of our work is to design a local controller K, a signal generator G, and a state estimator Q and R to minimize a cost functional J encoding the desired behavior of the system,

$$J = \phi(\mathcal{Y}(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}(\mathcal{Y}(\tau), u(\tau), \tau)\, d\tau\,, \tag{2.7}$$

where $u = [u_1^T \dots u_n^T]^T$ and $\mathcal{Y}$ represents the collection of all p_i, η_i, and y_i. Even when controllers cannot be designed to exactly minimize J, it can be used to measure the quality of a control design. This framework does not include sensing or communication costs, which may be important factors in certain applications.

2.1.1. The Need for Estimation

The need for the global state estimator Q and R is clear for the target localization problem—it is required to maintain estimates of the target location—but perhaps not so clear for the formation control problem. A natural question is whether purely reactive

memoryless controllers can be used to solve the same problem. For example, one common approach to coordinated behavior relies solely on sensing interaction of the agents. In this framework, each agent might sense the positions and velocities of its neighbors and choose its motion based on this information. In this simplification, there is neither an estimator state η_i nor an estimator output y_i, and only a memoryless controller K is to be designed. Simple memoryless gradient control laws are effective when the spatial gradient of J can be shown to be *spatially distributed* over the sensing network—the gradient of J with respect to p_i is a function only of p_i and $\mathcal{P}_i^{\text{sens}}$ [**16**, **14**]. For a particular cost J, the idea is to have the agents implement a sensing network over which the gradient of J is spatially distributed. Cortés et al. [**16**, **14**] describe a number of possible network structures based on different notions of distance to neighbors. When the gradient of J is spatially distributed over none of these networks, we can instead use a cost J' that approximately encodes the desired behavior and whose gradient is spatially distributed over some network.

This approach requires agents to reliably implement a particular distance-based sensing network topology. Even when these networks are implemented reliably, the requirement of a spatially-distributed gradient of J is restrictive. We are interested in a broader class of tasks with fewer restrictions on the network topology. As an example, the following variance-control task, a simplified version of the formation control problem, is only spatially distributed over a fully-connected network. As we see later, a gradient-based controller can be effective even in the face of changing communication networks using simultaneous estimation and control.

Example 3 (Variance Control)**.** *Consider a network of n agents with $p_i \in \mathbb{R}$. Each agent has simple integrator dynamics*

$$\dot{p}_i = u_i \in \mathbb{R}$$

and takes noise-free measurements of its own state as well as the states of its neighbors in the sensing network

$$z_i = \{p_i, \mathcal{P}_i^{sens}\}.$$

(Alternatively, each agent could simply measure the signed distance to its neighbors.) The desired behavior of the system is encoded by

$$J = \left[\frac{1}{n}\sum_{i=1}^{n}(p_i - \mu(p))^2 - \sigma_d^2\right]^2,$$

where $\mu(p) = \frac{1}{n}\sum_i p_i$. The cost is minimized ($J = 0$) when the variance of the agents' states is σ_d^2. The system is clearly controllable to the $(n-1)$-dimensional goal set $\mathcal{M}$. However, the gradient of J is only spatially distributed over the fully-connected sensing network. We conjecture that there is no reactive memoryless controller K that stabilizes the goal exactly for all n and all possible connected sensing networks.[2] *An interesting avenue for further research is to characterize the motion coordination problems that can and cannot be solved using purely reactive controllers.*

[2]Simple reactive controllers can be developed that *approximately* solve the task for arbitrary n and specific network structures, e.g., radius-limited sensing.

2.1.2. Simultaneous Estimation and Control

To solve a broad class of decentralized sensing and control problems, including the problem above, we adopt the following four-step design methodology:

(1) Choose a cost functional J encoding the desired performance of the system.

(2) Design an initial local controller K^{initial} to achieve desired convergence properties. This controller may use global information on the performance of the system (and thus may not be implementable), and it is often based on gradient descent.

(3) Design a signal generator G and a global state estimator Q and R such that each agent can estimate the global information required by the controller K^{initial}. The estimator should provide correct estimates in steady state.

(4) Construct the final control law K by replacing the global variables in K^{initial} by their local estimates, adding terms in the control law (if necessary) to preserve the desired convergence properties.

The design should be robust to changing network topologies and the addition and deletion of agents. It should also be scalable, meaning that the number of signals communicated by each agent $\dim(s_i)$ is independent of n.

The power of this approach rests on the capability of G, Q, and R to provide each agent with necessary estimates of the global performance of the system, subject to scalability constraints. In other words, each agent cannot simply pass around messages from all other agents with unique ID's attached to each message. Quantities that can be estimated in a scalable way include the minimum and maximum of state or sensed variables (each agent simply transmits the max or min value of its own sensor and any signal yet received) as

well as averages of variables. An algorithm for decentralized consensus estimation of the average of static inputs was described in [**55**] and extended to changing inputs in [**73**,**31**].

In this thesis we focus on problems that can be solved using *average consensus estimators*. The class of functions that can be estimated using average consensus estimators is quite large, as we can first apply nonlinear functions to the variables, then pass them through the average estimator, and then apply nonlinear transformations to the result. For example, using a natural log transformation, average consensus estimators can be used to calculate the geometric mean of inputs, as opposed to the arithmetic mean. In the formation control example, formation moments are estimated using nonlinear transformations (powers) of agent position data.

To summarize, we typically choose the controller K in (2.3) based on the gradient of J with respect to x_i, the estimator Q and R in (2.5)–(2.6) to use dynamic average consensus estimation to estimate the quantities needed to implement K, and the signal generator G in (2.4) to broadcast agent estimates.

2.2. Dynamic Average Consensus Estimators

The simultaneous estimation and control framework we proposed in principal applies to very wide range of problems. For the left of the thesis, we focus a class of problems where the estimation task can be performed by *dynamic consensus estimators*. Exemplary problems like this include controlling the center of mass of a robotic swarms and cooperating mobile sensors to do target tracking. In this section we introduce two dynamic consensus estimators and describe their convergence properties. These estimators on constructed on the general directed, weighted graphs (negative weights are allowed).

2.2.1. Proportional Consensus Estimator

Suppose each agent implements a proportional dynamic consensus estimator of the form

$$\dot{w}_i(t) = -\gamma w_i(t) - \sum_{j \neq i} a_{ij}(t)\big[x_i(t) - x_j(t)\big] \tag{2.8}$$

$$x_i(t) = w_i(t) + r_i(t)\,, \tag{2.9}$$

where $r_i(t) \in \mathbb{R}$ is the input, $x_i(t) \in \mathbb{R}$ is the estimator output, $w_i(t) \in \mathbb{R}$ is the internal estimator state, $\gamma \geqslant 0$ is a global estimator parameter, and $a_{ij}(t)$ are piecewise-continuous, time-varying estimator gains. This estimator is modified from the estimator in [**72**] by adding a forgetting factor γ.

We impose the constraint that $a_{ij}(t) = a_{ji}(t) = 0$ if agents i and j cannot communicate with each other at time t. The specific communication model varies in different context, e.g. it can be Euclidean distance based or line-of-sight based. We may write the collection of these n estimators in vector form as

$$\begin{aligned} \dot{w}(t) &= -\gamma w(t) - L(t)x(t) \\ &= -\big[\gamma I + L(t)\big]w(t) - L(t)r(t) \end{aligned} \tag{2.10}$$

$$x(t) = w(t) + r(t) \tag{2.11}$$

where $L(t)$ is the Laplacian matrix for the network graph. In fact the set of all possible Laplacians for such graphs is precisely the set

$$\mathfrak{L} = \big\{L \in \mathbb{R}^{nn} \,:\, L\mathbf{1} = 0\big\} = \big\{L \in \mathbb{R}^{nn} \,:\, L\Pi = L\big\}\,, \tag{2.12}$$

where $\Pi \in \mathbb{R}^{nn}$ denotes the projection matrix

$$\Pi = I - \frac{\mathbf{1}\mathbf{1}^T}{n}. \tag{2.13}$$

For each real parameter ε we define

$$\mathcal{L}^{\varepsilon} = \left\{ L \in \mathcal{L} \,:\, \Pi(L + L^T)\Pi \geqslant 2\varepsilon\Pi \right\} \tag{2.14}$$

and we use

$$\mathcal{L}^{bal} = \left\{ L \in \mathcal{L} \,:\, L^T\mathbf{1} = 0 \right\} \tag{2.15}$$

to represent the set of Laplacians for balanced graphs.

We are interested in achieving average consensus, namely, we would like the error vector

$$e_x(t) = x(t) - \frac{\mathbf{1}\mathbf{1}^T}{n}\, u(t) \tag{2.16}$$

to approach zero as $t \to \infty$. We cannot expect to achieve small steady-state errors when the input vector $u(t)$ is changing too rapidly, because it takes time for the effects of $u(t)$ to flow across the network. For this reason we will assume that both $u(t)$ and its derivative $\dot{u}(t)$ are bounded. The convergence result of this estimator is analyzed in [**31**]:

Theorem 1. *Let $L \in \mathcal{L}_{bal}$ be a piecewise-continuous, time-varying Laplacian matrix, and suppose there exist $\varepsilon, t_0 \in \mathbb{R}$ such that $L(t) \in \mathcal{L}^{\varepsilon}$ for all $t \geqslant t_0$. Let $\gamma \geqslant 0$ be such that $\gamma + \varepsilon > 0$, and suppose there exists $\mu \geqslant 0$ such that the input vector $r(t)$ is absolutely*

continuous and satisfies

$$|\gamma r(t) + \dot{r}(t)| \leqslant \mu \tag{2.17}$$

for almost all $t \geqslant t_0$. *Then the consensus estimator* (2.10)–(2.11) *is such that the error vector* $e_x(t)$ *in* (2.16) *satisfies*

$$\begin{aligned} |e_x(t)| \leqslant \frac{1}{\sqrt{n}} \, |\mathbf{1}^T w(t_0)| e^{-\gamma(t-t_0)} \\ + |\Pi x(t_0)| e^{-\frac{1}{2}(\gamma+\varepsilon)(t-t_0)} + \frac{\mu}{\gamma+\varepsilon} \end{aligned} \tag{2.18}$$

for all $t \geqslant t_0$.

In the theorem, there is no requirement for the parameter ε to be positive; in particular, the estimate (2.18) holds even when $L(t) \equiv 0$, namely, when there is no communication at all between the agents. However, in this case the steady-state value of the bound (2.18) on the error $|e_x(t)|$ will be no smaller than $|r|$ for a constant input r (take $\mu = \gamma|r|$). To achieve small steady-state errors for constant (or slowly varying) inputs, we need $\varepsilon > 0$ and $\gamma \ll \varepsilon$. Furthermore, we see from (2.14) that ε scales with the estimator weights, so when $\varepsilon > 0$ we can simply increase these weights to achieve arbitrarily small steady-state errors for given values of γ and μ. However, in doing so we also increase the maximum eigenvalue of the graph Laplacian and thereby reduce the robustness of the estimator to communication delays between agents [**56**]. Hence the presence of any such delays will limit the bound on the derivative $\dot{u}(t)$ under which we can achieve accurate tracking.

The choice $\gamma = 0$ is appealing for the case $\varepsilon > 0$ because then for constant inputs we have $\delta(t) \to 0$ as $t \to \infty$ (take $\mu = 0$). However, in this case $\mathbf{1}^T w(t)$ is constant and thus

$e_x(t) \to \mathbf{1}\mathbf{1}^T w(t_0)/n$ as $t \to \infty$. In other words, only a correct estimator initialization (say $w(t_0) = 0$) will lead to zero steady-state error. One consequence of this property is that if an agent enters or leaves the network, a simultaneous reinitialization of the estimators will be required to guarantee zero steady-state error for the new consensus system. In contrast, choosing a small but positive γ would introduce a small steady-state error but would also allow the network to slowly "forget" any larger errors introduced by incorrect estimator initializations. To eliminate steady-state errors entirely without the need for correct initializations, at least for the case of constant inputs and Laplacians, we introduce an integral term in the estimator as described in the next section.

We call this estimator (2.10)–(2.11) a high-pass estimator because if the estimator gains a_{ij} were constant, then the resulting LTI system taking the inputs r_i to the outputs y_i would be a high-pass filter with unity high-frequency gain.

2.2.2. Proportional-Integral Consensus Estimator

Suppose each agent implements a proportional-integral dynamic consensus estimator of the form

$$\dot{x}_i(t) = -\gamma x_i(t) - \sum_{j \neq i} a_{ij}(t)\big[x_i(t) - x_j(t)\big] + \sum_{j \neq i} b_{ji}(t)\big[w_i(t) - w_j(t)\big] + \gamma r_i(t) \tag{2.19}$$

$$\dot{w}_i(t) = -\sum_{j \neq i} b_{ij}(t)\big[x_i(t) - x_j(t)\big], \tag{2.20}$$

where $r_i(t) \in \mathbb{R}$ is the input, $x_i(t) \in \mathbb{R}$ is the estimator output, $[x_i(t)\ w_i(t)]^T \in \mathbb{R}^2$ is the internal estimator state, $\gamma > 0$ is a global estimator parameter, and $a_{ij}(t)$ and $b_{ij}(t)$

are piecewise-continuous time-varying estimator gains. We impose the constraint that $a_{ij}(t) = a_{ji}(t) = b_{ij}(t) = b_{ji}(t) = 0$ if agents i and j cannot communicate with each other at time t. We may write the collection of these n estimators in vector form as

$$\begin{bmatrix} \dot{x}(t) \\ \dot{w}(t) \end{bmatrix} = \begin{bmatrix} -\gamma I - L_P(t) & L_I^T(t) \\ -L_I(t) & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix} + \begin{bmatrix} \gamma I \\ 0 \end{bmatrix} r(t)\,, \tag{2.21}$$

where $L_P(t)$ is the *proportional Laplacian* (constructed from the weights a_{ij}) and $L_I(t)$ is the *integral Laplacian* (constructed from the weights b_{ij}).

Unlike the high-pass estimator (2.10)–(2.11), this PI estimator has no direct feedthrough from its input r_i to its output y_i, and thus it may provide better filtering of high-frequency noise. In addition, it approaches a ball around zero whose size is related to the rate of change of the input, with constant input producing errors that decay exponentially to zero [**31**]. For the high-pass estimator [**73**], zero steady-state error requires extra bookkeeping to keep track of which communication links are active. Besides, intermittent communication noise or drops cause the high-pass filters to drift, whereas a "forgetting" factor in the PI filter results in a stable filter from communication noise to errors relative to the solution manifold.

We are again interested in achieving average consensus, namely, we would like the error vector $e_x(t)$ in (2.16) to approach zero as $t \to \infty$.

Theorem 2. *Assuming L_P, L_I are constant Laplacians and let $\varepsilon \in \mathbb{R}$ be such that $L_I \in \mathcal{L}^\tau, L_P \in \mathcal{L}^\varepsilon$. Suppose rank $L_I = n - 1$ and suppose the estimator parameter $\gamma > 0$ is chosen such that $\gamma + \varepsilon > 0$. Then for any constant input $u \in \mathbb{R}^n$ and any initial states*

$x(t_0), w(t_0) \in \mathbb{R}^n$, the trajectories of the system

$$\begin{bmatrix} \dot{x}(t) \\ \dot{w}(t) \end{bmatrix} = \begin{bmatrix} -\gamma I - L_P & L_I^T \\ -L_I & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix} + \begin{bmatrix} \gamma I \\ 0 \end{bmatrix} u \tag{2.22}$$

are such that $x(t)$ and $w(t)$ converge to constant vectors and $e_x(t) \to 0$ exponentially as $t \to \infty$.

2.3. A Simple Case: Kinematic and Holonomic Agents on an Undirected, Unweighted Communication Network

In this section, we describe the detailed design procedures for a simple case within the general framework outlined in the last section. In this simple case, we assume that

(1) The agent model is kinematic, fully actuated and holonomic.

(2) There is no communication delay in the network.

(3) The cost function J only depends on the average type of global information.

(4) The agents share an undirected communication network.

Next we explain the four-step design procedures step by step.

2.3.1. Cost Function

We have n kinematic agents, and the configuration of agent i is written $p_i = [p_i^1 \dots p_i^q]^T \in \mathbb{R}^q$. The total system configuration is $p \in \mathbb{R}^{nq}$. The cost J encoding the desired group behavior is a function of global properties f of the system. Specifically, we consider cost functions that can be written in a special form

$$J(p) = J(f(p), \beta), \tag{2.23}$$

where

$$f(p) = \frac{1}{n}\sum_{i=1}^{n} g(p_i) = [f^1 \cdots f^m]^T \in \mathbb{R}^m \tag{2.24}$$

are m pieces of global information. These are simply the average over each agent's local information $g(p_i)$, obtained from internal state and sensor measurements. Examples of f include the center of mass of a swarm in formation control [**30**], average sensor readings (temperature, gas concentration, etc.) in sensor networks [**11**] and global estimate uncertainty in target localization [**12**]. The constants β are known to each agent, and may represent common goals or controller gains. We require J to be *nonnegative* and *proper* in p (i.e., radially unbounded) so we can use it as a global storage function.

2.3.2. Design the Initial Controller $K^{initial}$

In the initial design, each agent implements a gradient controller

$$\dot{p_i} = u_i = -\frac{\partial J}{\partial p_i}. \tag{2.25}$$

Note that the control gains are present in the definition of the cost J; a cost αJ with $\alpha > 1$ would give agent motions in the same direction, but at a higher speed. Notice that

$$\frac{\partial J}{\partial p_i} = \begin{bmatrix} \frac{\partial f^1}{\partial p_i^1} & \cdots & \frac{\partial f^m}{\partial p_i^1} \\ \vdots & & \vdots \\ \frac{\partial f^1}{\partial p_i^q} & \cdots & \frac{\partial f^m}{\partial p_i^q} \end{bmatrix} \begin{bmatrix} \frac{\partial J}{\partial f^1} \\ \vdots \\ \frac{\partial J}{\partial f^m} \end{bmatrix}$$

$$= \quad D_i^T \begin{bmatrix} \frac{\partial J}{\partial f^1} \\ \vdots \\ \frac{\partial J}{\partial f^m} \end{bmatrix}$$

with $D_i = \frac{\partial f}{\partial p_i} \in \mathbb{R}^{m \times q}$. In general this is a centralized design; although D_i can be obtained locally for each agent i, the value of $\frac{\partial J}{\partial f}$ depends on f. This is in contrast to *spatially distributed* objective function gradients where $\frac{\partial J}{\partial p_i}$ depends only on p_i and its sensed neighbors.

This design guarantees J to be non-increasing along trajectories in forward time. When J is nonnegative and proper, we know every trajectory is bounded. Based on LaSalle's theorem we further conclude that every trajectory converges to an equilibrium set at which $\frac{\partial J}{\partial p} = 0$. In general this set may contain local minima which are not global minima.

2.3.3. Designing the Signal Generator G and the State Estimator Q, R

To estimate f in the initial controller $K^{initial}$, each agent runs a *Proportional Average Consensus Estimator* in section 2.2.1 with local input $\phi_i = g(p_i)$. Given inputs, internal states and outputs ϕ_i, w_i, $x_i \in \mathbb{R}^m$, the P estimator looks like:

$$\dot{w}_i = -\gamma w_i - K_p \sum_{j \in \mathcal{N}_i} \left[x_i - x_j \right] \tag{2.26}$$

$$x_i = w_i + \phi_i. \tag{2.27}$$

Note that we are using an undirected network, and we apply equal weight on each link.

2.3.4. Construct the Final Control Law K

We obtain the final distributed controller K as follows. Each agent estimates the global information f by running a P average consensus estimator with local input $\phi_i = g(p_i)$, then replaces the f in the controller with its estimator output $x_i = [x_i^1 \cdots x_i^m]^T$. After that we scale the control effort by $[I + D_i^T \Lambda_i D_i]^{-1}$, yielding

$$\dot{p_i} = u_i = -[I + D_i^T \Lambda_i D_i]^{-1} \left. \frac{\partial J}{\partial p_i} \right|_{f=x_i} . \tag{2.28}$$

where $D_i = \frac{\partial f}{\partial p_i} \in \mathbb{R}^{m \times q}$, $I \in \mathbb{R}^{m \times m}$ is the identity matrix and $\Lambda_i > 0$.

The foremost concern of this new *feedback* system is stability: in a stable centralized controller, when we replace the global information f with an estimator output x_i, the introduced estimation error $e_i = f - x_i$ may drive the system unstable. The positive-definite matrix $D_i^T \Lambda_i D_i$ in (6) serves to reduce the control gains to guaranteed the overall stability of the feedback-connected estimation and control process. For this reason, we refer to Λ_i as a nonlinear damping matrix. Larger damping, i.e., smaller control gains, improve the stability of the system, possibly at the expense of performance. Our small-gain condition specifies a lower bound on Λ_i that guarantees semi-global stability of the coupled system. The full stability analysis is presented in the next section.

2.3.5. Stability: the Small-gain Condition

We define the estimator input and output variables Z, E as:

$$Z = [z_1 \cdots z_n]$$

$$
\begin{aligned}
&= [\frac{d}{dt}g(p_1)\cdots\frac{d}{dt}g(p_n)] \\
E &= [e_1\cdots e_n] \\
&= [f\cdots f]-[x_1\cdots x_n]
\end{aligned}
$$

Taking $U = \mathrm{tr}(EE^T)$ as the measure of the estimation error, the input-output relationship of the P estimator is given in [**30**]:

$$\dot{U} \leqslant -\varepsilon U + \frac{1}{\varepsilon}\mathrm{tr}(ZZ^T) = \sum_{i=1}^{n}\Big[-\varepsilon|e_i|^2 + \frac{1}{\varepsilon}|z_i|^2\Big] \tag{2.29}$$

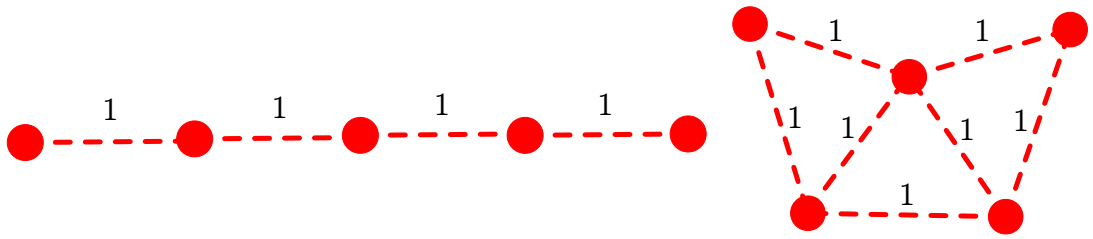
where $\varepsilon(t) = K_p\lambda_2(t)$ and $\lambda_2(t)$ is the algebraic connectivity [**32**] of the underlying communication graph Laplacian. Here we use the undirected graph given by the neighbor relations $\mathcal{N}_i$ with unit weights. For a connected network with n nodes, it is known that λ_2 reaches its minimum value $\lambda_{\min} = 2 - 2\cos(\pi/n)$ [**23**] when the communication topology is a line graph (Fig. 2.2).

Now we characterize the input-output relationship of the controller. First we find out how much the control effort changes when we replace the global information f with x_i in the controller. We can use the Mean Value theorem to obtain:

$$\left.\frac{\partial J}{\partial f^j}\right|_{f=f} - \left.\frac{\partial J}{\partial f^j}\right|_{f=x_i} = \left.\frac{\partial^2 J}{\partial f^j \partial f}\right|_{f=\widetilde{f}_{ij}} e_i \tag{2.30}$$

where each

$$
\begin{aligned}
\widetilde{f}_{ij} &= f - \alpha_j(f - x_i) \\
&= f - \alpha_j e_i \qquad 0 \leq \alpha_j \leq 1
\end{aligned} \tag{2.31}
$$



(a)

$$L = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}, \lambda_2 = 0.382$$

(b)

$$L = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 3 & -1 & 0 & -1 \\ 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & -1 & -1 & -1 & 4 \end{bmatrix}, \lambda_2 = 1.59$$

Figure 2.2. Two communication topologies and graph Laplacians induced from neighborhood relations. Among connected graphs, the line graph in (a) has the smallest algebraic connectivity $\lambda_2 = 2 - 2\cos(\pi/5) = 0.382$.

is on the line segment between f and x_i. We denote the row vector $\frac{\partial^2 J}{\partial f^j \partial f}\Big|_{f=\tilde{f}_{ij}}$ as C_{ij}. Furthermore, we call the assembled matrix $C_i = [C_{i1}^T \ \cdots \ C_{im}^T]^T$ the Hessian matrix of the system. With this notation, the change in control effort is written

$$\begin{aligned} & \frac{\partial J}{\partial p_i}\Big|_{f=f} - \frac{\partial J}{\partial p_i}\Big|_{f=x_i} \\ = \ & \left(\frac{\partial f}{\partial p_i}\right)^T \left(\frac{\partial J}{\partial f}\Big|_{f=f} - \frac{\partial J}{\partial f}\Big|_{f=x_i}\right) \\ = \ & D_i^T C_i e_i. \end{aligned} \tag{2.32}$$

which is the error between the centralized design and the distributed design. Then the input-output relationship of the controller is given as

$$\dot{J} = \sum_{i=1}^{n} \dot{p_i}^T \frac{\partial J}{\partial p_i}$$

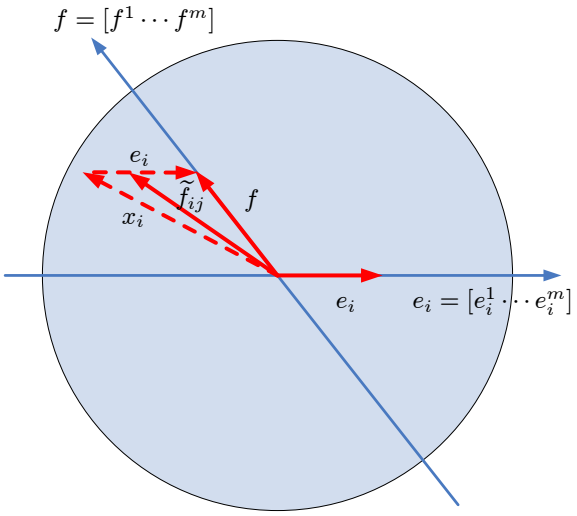


Figure 2.3. At any given time t, $\widetilde{f}_{ij}$ lies on the line segment of f and x_i for every (i, j) pair. It is therefore bounded by a circle of radius $\|f\| + \|e_i\|$.

$$
\begin{aligned}
&= \sum_{i=1}^{n} \dot{p_i}^T [I + D_i^T \Lambda_i D_i][I + D_i^T \Lambda_i D_i]^{-1} (\frac{\partial J}{\partial p_i}\Big|_{f=x_i} + D_i^T C_i e_i) \\
&= -\sum_{i=1}^{n} \dot{p_i}^T [I + D_i^T \Lambda_i D_i] \dot{p_i} + \sum_{i=1}^{n} \dot{p_i}^T D_i^T C_i e_i \\
&\leq -\sum_{i=1}^{n} \dot{p_i}^T (I + D_i^T (\Lambda_i - \frac{C_i C_i^T}{\varepsilon}) D_i) \dot{p_i} + \sum_{i=1}^{n} \varepsilon |e_i|^2 \\
&= -\sum_{i=1}^{n} \dot{p_i}^T \dot{p_i} - \sum_{i=1}^{n} z_i^T (\Lambda_i - \frac{C_i C_i^T}{\varepsilon}) z_i + \sum_{i=1}^{n} \varepsilon |e_i|^2.
\end{aligned}
$$

To combine J and U together, we define a storage function

$$\Upsilon(p, E) = J + (1 + \mu)U \tag{2.33}$$

with $\mu > 0$. The dissipation inequality looks like:

$$\dot{\Upsilon} \leqslant -\sum_{i=1}^{n} \dot{p_i}^T \dot{p_i} - \sum_{i=1}^{n} z_i^T (\Lambda_i - \frac{C_i C_i^T + (1+\mu)I}{\varepsilon}) z_i - \sum_{i=1}^{n} \mu\varepsilon |e_i|^2. \tag{2.34}$$

The following theorem gives a sufficient condition for a stable design of the feedback system.

Theorem 3. *Given task $J = J(f, \beta)$ and we assume its centralized gradient design is stable and the possibly time-varying communication network remains connected. If either one of the following conditions holds:*

(1) *the cost function J is smooth and proper in f, or*

(2) *the global information f is globally bounded,*

Then there exists a corresponding distributed design: Each agent constructs a P estimator to estimate the required global information and closes the feedback loop by adding nonlinear damping with gains

$$\Lambda_i \geq \frac{Q + (1+\mu)I}{\lambda_{min} K_p} \tag{2.35}$$

where $Q = \sup_{(\widetilde{f}_{i1}, \cdots, \widetilde{f}_{im}) \in \Psi^m} C_i C_i^T$. The resulting system is stable and each trajectory converges to the same equilibrium set as in the central design.

Proof. We use Υ_0 to denote the initial value of the storage function Υ. Based on its definition in (2.33), we know

$$\|e_i\| \leq \sqrt{\frac{\Upsilon_0}{1+\mu}}, \quad J(f) \leq \Upsilon_0. \tag{2.36}$$

In either J is proper in f or f is globally bounded, we have

$$\|e_i\| \leq \sqrt{\frac{\Upsilon_0}{1+\mu}}, \quad \|f\| \leq a \tag{2.37}$$

for some scale $a \in \mathbb{R}$. Starting from (2.31), the triangle inequality gives

$$\|\widetilde{f}_{ij}\| \leq \sqrt{\frac{\Upsilon_0}{1+\mu}} + a \tag{2.38}$$

Additionally we have

$$C_i C_i^T \leq Q = \sup_{(\widetilde{f}_{i1},\cdots,\widetilde{f}_{im})\in\Psi^m} C_i C_i^T < \infty \tag{2.39}$$

with $\Psi = [-\sqrt{\frac{\Upsilon_0}{1+\mu}} - a, \sqrt{\frac{\Upsilon_0}{1+\mu}} + a]$ (see Fig. 2.3). Q is a function of the C_i, which are a function of the control gains. Now we choose

$$\Lambda_i \geq \frac{Q + (1+\mu)I}{\lambda_{\min} K_p} \geq \frac{Q + (1+\mu)I}{\varepsilon K_p} \tag{2.40}$$

so that the small gain condition in (2.34)

$$\varepsilon \Lambda_i \geq C_i C_i^T + (1+\mu)I \tag{2.41}$$

is satisfied at $t = 0$. From the Lyapunov stability theorem, $\Upsilon(t) < \Upsilon_0$ and therefore (2.37), (2.38) remain satisfied. Because (2.31) holds over time, we know (2.38) still holds and the domain Ψ will not increase. Therefore by satisfying (2.40), the small gain condition (2.41) is automatically satisfied over time and $\dot{\Upsilon}(t) \leq 0$ from (2.34). By LaSalle's theorem, every trajectory converges to an equilibrium set at which $\dot{p} = 0$ and

$E = 0$. So in steady state the estimation error vanishes and the trajectory will converge to the same equilibrium set as in the centralized design. □

When $C_i C_i^T$ is globally bounded (e.g., J is a quadratic cost function) we get a global stability result, otherwise the result is semi-global. In general, the value of Q can be obtained from a standard nonlinear optimization solver. It is also possible to bound C_i by taking advantage of special problem structures. We use the second approach when we analyze the stability of the target localization algorithm.

CHAPTER 3

Designing Coordination Algorithms: Optimizing the Estimator Convergence Speed

In this chapter we are interested in speeding up the convergence speed of the consensus estimators. Based on the small gain condition 2.41 in Theorem 3, this helps decrease the nonlinear damping gain, and therefore increase the convergence speed of the overall feedback system. As discussed in section 1.2, we can use communication weights as our design freedom to optimize the convergence speed.

In the original static consensus estimator, each agent looks to its neighbors and changes its decision value based on the difference between its own value and the weighted average of its neighbors:

$$\dot{x}_i = \sum_{j \neq i} a_{ij}(x_j - x_i).$$

Collecting the individual agents' estimators together into a continuous-time matrix equation, we get

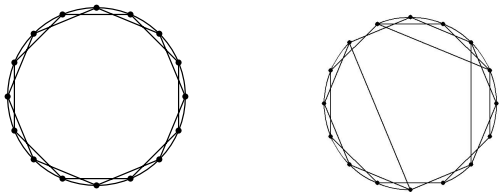
$$\dot{x}(t) = -Lx(t). \tag{3.1}$$

When all the nonzero eigenvalues of the symmetric Laplacian $L(\mathcal{G})$ have strictly positive real parts, this protocol is known to solve the static average consensus problem—the final network decision value is equal to the average of the initial decision values, e.g., the sensed inputs to the network at $t = 0$ [**55**].

The worst-case convergence time of the first-order differential equation in (3.1) is inversely proportional to $\lambda_2(L)$, the second-smallest eigenvalue of L (i.e., other than the one at 0). $\lambda_2(L)$ is called the *algebraic connectivity* of the graph, and $1/\lambda_2(L)$ is the associated time constant.

The sum of the eigenvalues of a symmetric Laplacian L is equal to the trace of its degree matrix D. Therefore, λ_2 can be made arbitrarily large, and the convergence time constant arbitrarily small, by choosing the communication weights a_{ij} to be arbitrarily large positive numbers. As shown by Olfati-Saber and Murray, however, the presence of a communication delay in the communication links may destabilize the estimator for large weights [**55**]. In particular, they showed that stability of the estimation is only assured if the maximum communication delay $T > 0$ is bounded by $T < \pi/(2\lambda_{\max})$, where $\lambda_{\max} = \lambda_n$ is the largest eigenvalue of L. This establishes an upper bound $\lambda^*_{\max}$. With this in mind, we define the (normalized) speed of the network convergence to be $\nu = \lambda_2(L)/\lambda_{\max}(L)$ with associated normalized time constant $\tau = 1/\nu$. Our goal is to minimize τ, i.e., to minimize the convergence time subject to the constraint that the system always remain stable in the presence of intermittent time delays bounded by T. Note that $\tau = 1$ is the minimum time constant possible, and this is achieved for a fully connected network with all weights positive and equal. In this case, $\lambda_2 = \lambda_3 = \ldots = \lambda_n > 0$.

To summarize, we can compare arbitrary weighted symmetric graphs by comparing how closely they resemble a fully connected network with equal weights, the ideal. The optimal edge-weighting problem is to choose edge weights that achieve a τ as close to 1 as possible.



(a) (b)

Figure 3.1. (a) A regular network on S^1 with $n = 16$ and $k = 4$. (b) One possible outcome of the network after random rewiring with $p = 0.1$.

3.1. Optimal Edge Weighting - a Central LMI Approach

As the basic network topology is derived from radius-limited communication, agents evenly distributed over a space will all have the same number of network neighbors, and locally the network will look the same everywhere (except at boundaries). We call this a regular *grid*. If each agent has k neighbors, we call it a *k-regular* grid. To focus our examples, we use the model case of evenly-spaced agents on a one-dimensional metric space, giving rise to a regular grid with n nodes and k neighbors per node. To eliminate boundary effects, the underlying space is the circle S^1. An example regular grid with $n = 16$ and $k = 4$ is shown in Figure 3.1. The adjacency matrix for this undirected graph has the symmetric banded form

$$A = \begin{bmatrix} 0 & a_{12} & a_{13} & 0 & \cdots & 0 & a_{1(n-1)} & a_{1n} \\ a_{21} & 0 & a_{23} & a_{24} & 0 & \cdots & 0 & a_{2n} \\ a_{31} & a_{32} & 0 & a_{34} & a_{35} & 0 & \cdots & 0 \\ & & & & & \vdots & & \end{bmatrix}. \tag{3.2}$$

The graph edges are undirected and we can see that the graph structure has rotation symmetry. Therefore each row of the optimal A matrix, as well as $L(A) = D(A) - A$, will

be a cyclic shift of the row above it. Such a matrix is called a *circulant matrix*. If the top row of a circulant matrix C is $[c_0, \ldots, c_{n-1}]$, where each row below is shifted cyclically one to the right, we can find the n eigenvalues λ_m and eigenvectors $y_m, m = 1 \ldots n$, of C analytically [**33**] as

$$\lambda_m = \sum_{p=0}^{n-1} c_p e^{-2\pi imp/n}, \tag{3.3}$$

$$y_m = [1, e^{-2\pi im/n}, \ldots, e^{-2\pi im(n-1)/n}]^T. \tag{3.4}$$

The circulant matrix $C = L(A)$ also has the property that $c_p = c_{n-p}$ due to reflection symmetry. Therefore we can choose $k/2$ variables $u_1, \ldots, u_{k/2}$, where k is the number of neighbors, and write the top row of $L(A)$ as $[c_0, -u_1, \ldots, -u_{k/2}, 0, \ldots, 0, -u_{k/2}, -u_{(k/2)-1}, \ldots, -u_1]$, where each row below is a cyclic shift. With the constraint $c_0 = 2\sum_{p=1}^{k/2} u_p$, and by cancelation of imaginary components, (3) becomes

$$\lambda_m = 2\sum_{p=1}^{k/2} u_p(1 - \cos(2\pi mp/n)). \tag{3.5}$$

As we uniformly scale the weights up, the eigenvalues of L grow without bound, possibly resulting in instability in the presence of time delays. Therefore we impose the constraint $\lambda_{\max} \leq 1$. The problem now is to find the weights that maximize λ_2 subject to this constraint, thereby minimizing τ. This can be reduced to the following linear program:

$$\begin{aligned} \text{find} \quad & u_1, \ldots, u_{k/2}, \lambda \\ \text{maximizing} \quad & \lambda \end{aligned}$$

$$\begin{aligned} \text{subject to} \quad & \lambda \leq 2\sum_p u_p(1-\cos(2\pi mp/n)) \\ & 2\sum_p u_p(1-\cos(2\pi mp/n)) \leq 1 \\ & m = 1,\ldots,n-1. \end{aligned}$$

Figure 3.2(a,b) gives the results of the optimization for a regular grid with $n = 150$. The optimal edge weights for $n = 150, k = 80$ are given in Figure 3.2(a). Interestingly, for every scenario we tested, we find that optimal weights for nearby agents are very small numbers (even negative), and weight for the most distant agent is large. We have no simple explanation for these optimal weights, but they are heavily dependent on the strict symmetry of the regular grid. Figure 3.2(c) shows the convergence speedup from optimal weighting of the edges. The graph shows the time constant τ associated with the naïve weighting strategy assigning all edges equal weight as compared to the time constant of the optimally weighted graph.

If we restrict all weights to be nonnegative, the optimal time constant increases, as the problem is more constrained. For every scenario we tested with this constraint, we find that neither τ nor the shape of the optimal weights profile changes much. An example is shown in Figure 3.2(b).

The edge-weight optimization problem for the more general case of arbitrary but fixed network topologies can be formulated using linear matrix inequality (LMI) theory. The Laplacian matrix for arbitrary undirected graphs is still symmetric, but not necessarily circulant. If the network has r edges, let $u_p, p = 1 \ldots r$, be the weights associated with those edges, and let $L(u)$ be the associated graph Laplacian. We keep the normalization constraint $\lambda_{\max} \leq 1$. The goal is to find the vector u maximizing $\lambda_2(L)$.

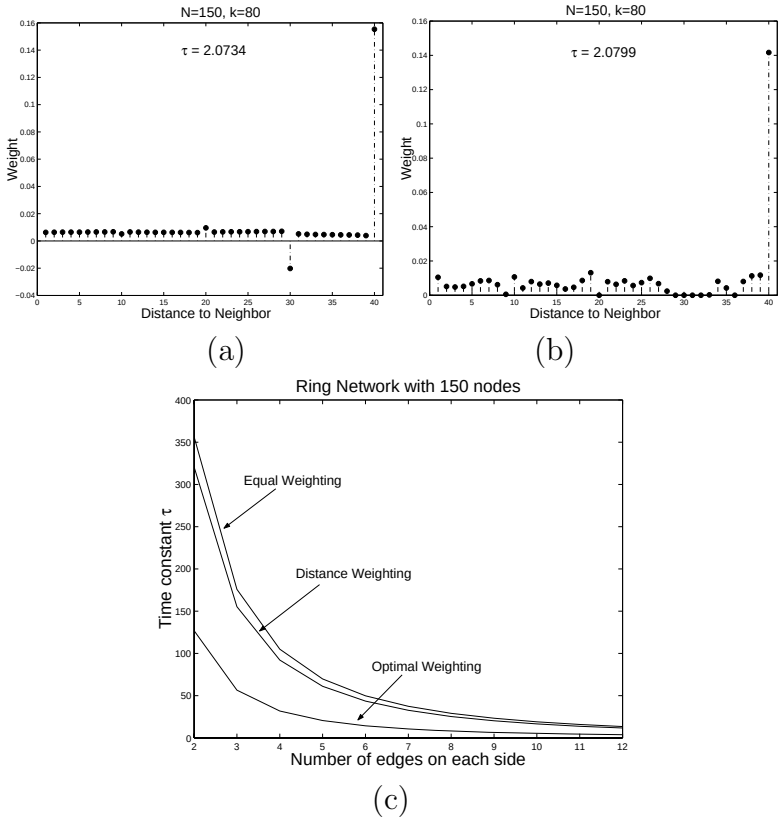


Figure 3.2. (a) The optimal weights for a network with $n = 150, k = 80$. (b) The optimal nonnegative weights for the same network (a). (c) The consensus speedup for optimal and distance weighting for $n = 150$ and for a range of k values ($k/2$ shown on the x-axis).

To do this, find an $n \times (n-1)$ matrix A satisfying $A^T A = I$ whose column vectors form an orthogonal complement of the eigenvector $\mathbf{1}$, which corresponds to the eigenvalue of L at 0. Then the eigenvalues of the $(n-1) \times (n-1)$ matrix $A^T L A$ are the same as the remaining eigenvalues of L. The smallest eigenvalue of $A^T L A$ is equivalent to $\lambda_2(L)$.

As $A^T L A$ is symmetric, the quantity $A^T L A - \lambda I$ is positive definite if λ is smaller than the smallest eigenvalue of $A^T L A$, and similarly the normalization constraint can be

formulated as $A^T LA \leq I$. So the optimization becomes

$$\begin{aligned} \text{find} \quad & u_1, \ldots u_r, \lambda \\ \text{maximizing} \quad & \lambda \\ \text{subject to} \quad & A^T L(u_1, \ldots, u_r)A - \lambda I \geq 0 \\ & I - A^T L(u_1, \ldots, u_r)A \geq 0, \end{aligned}$$

an LMI problem, as L is linear in u. This problem can be solved readily using standard methods. We are using this formulation to compare optimal weighting strategies on arbitrary graphs to simple distributed heuristic weighting strategies, as discussed in the next section (Figure 5).

3.2. Decentralized Heuristic Edge Weighting

The optimal edge-weighting algorithms above are centralized algorithms. If the network topology is unknown or changing, each agent must decide the weights to apply to its edges in a decentralized way, without sacrificing the correctness of the algorithm. As shown in [**55**], the network must be *balanced* to converge to the average of the inputs, i.e., for each agent, the sum of its outgoing weights must be equal to the sum of its incoming weights.

A simple way to satisfy this condition is to ensure L is symmetric. We also must make sure that the nonzero eigenvalues of L are positive. This is satisfied trivially by requiring that each edge weight be positive. Therefore we restrict our search for heuristic weighting

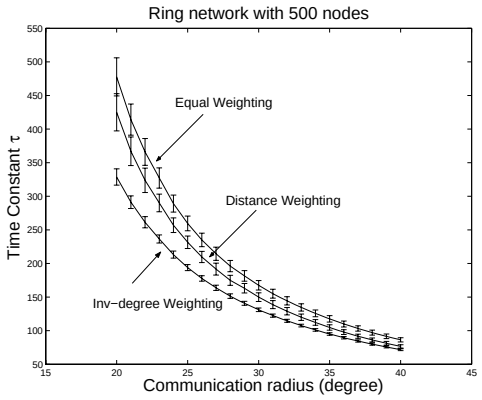


Figure 3.3. The consensus speedup for different heuristic weighting schemes over equal weighting on a random radius-limited graph on a ring with 500 nodes. The communication radius is in degrees around the ring. Error bars are standard deviations over 100 graphs; the same holds for all subsequent graphs.

strategies to symmetric functions of the form $a(i,j) = a(j,i) > 0$ for any two nodes i, j with a communication link.

One heuristic weighting strategy satisfying these conditions that each agent can implement in a decentralized fashion is to weight each edge proportionally to the metric distance between the agents. This captures the idea that more distant information should be weighted more heavily, in the interest of propagating information more quickly. This heuristic is motivated by the fact that the metric of the underlying space is encoded in the topology of the radius-limited communication graph. As seen in Figure 3.2(b), this simple heuristic provides marginal speedup over the equal weighting strategy.

Figure 3.3 compares time constants of distance weighting over the equal weighting for networks consisting of 500 nodes randomly chosen on the ring from a uniform distribution. A similar speedup as in the regular network case is observed.

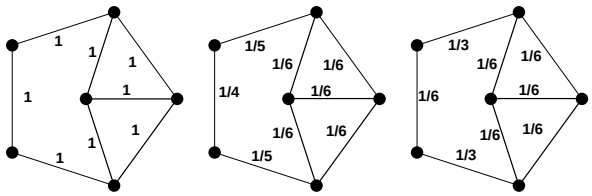


Figure 3.4. The weights assigned under different schemes: (a) Equal weighting with $\tau = 3.73$. (b) Inverse-degree weighting with $\tau = 3.43$. (c) Optimal weighting with $\tau = 3$ solved by LMI.

Another weighting heuristic we have explored is choosing a link's weight to be the reciprocal of the total number of agents the two agents are in communication with. This inverse-degree weighting algorithm requires each agent to communicate the number of its neighbors. The idea is that information from an agent with little contact with other agents should be weighted more heavily to help its information propagate faster, and to help information from other agents propagate more quickly to it, as the convergence time is determined by the slowest mode. Simulations prove the effectiveness of this heuristic for different network sizes and densities, as shown in Figure 3.3.

To further explore the hypothesis that the power of this heuristic derives from more heavily weighting nodes with fewer connections, we tested the heuristic on a number of random graphs. Each graph had 500 nodes and was generated to satisfy a desired distribution of the connection degrees of the nodes, using the algorithm in [**80**]. (Note: these graphs do not derive from radius-limited communication among agents on a ring, as with other graphs discussed in this thesis.) We tested a one-parameter family of graphs with degree histograms with fixed total degree, as it is known that adding edges decreases the convergence time [**27**]. The degree histograms are symmetric triangles with a peak at

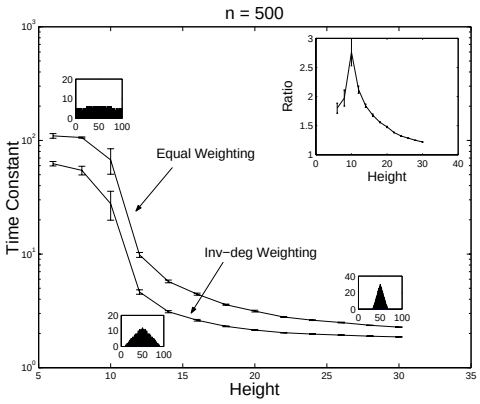


Figure 3.5. The influence of degree distribution on convergence time. The network has 500 nodes and an average degree connection of 50. Each sample point is generated from 100 trials. Insets show the degree histograms at sample points as well as the time constant ratio between two weighting schemes.

degree 50. The one parameter controlling the histogram shape is the height of the peak, which we varied from a minimum of 5 nodes to a maximum of 40. As the histogram becomes narrower and more peaked, the random graph approaches a regular graph, and the inverse-degree weighting becomes equivalent to equal weighting. Figure 3.5 confirms that the inverse-degree heuristic is more useful when the degree distribution contains nodes of low degree. It also shows a phase transition in the time constant as a function of the degree histogram for both the equal and inverse-degree weighting schemes.

As the nonnegative combination of any two symmetric positive functions is also symmetric and positive, we would like to know if some linear combination of the distance weighting scheme and inverse-degree weighting scheme can result in better network performance. We do not yet know the answer in general. For a particular network of $n = 200$ and communication radius of $r = 15^\circ$ (Figure 3.6), the inverse-degree heuristic is better than any linear combination with the distance metric. In contrast to the good heuristics

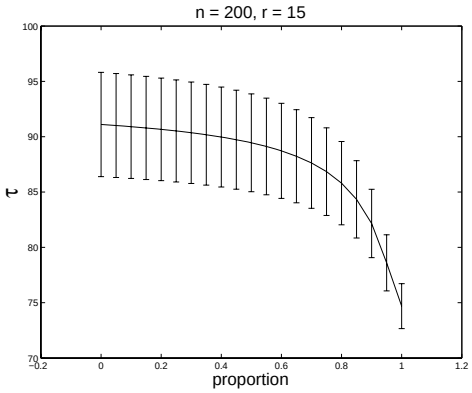


Figure 3.6. Time constants for the combined heuristic weighing. Proportion is 0 for the purely distance weighting case and 1 for the purely inverse-degree weighting case.

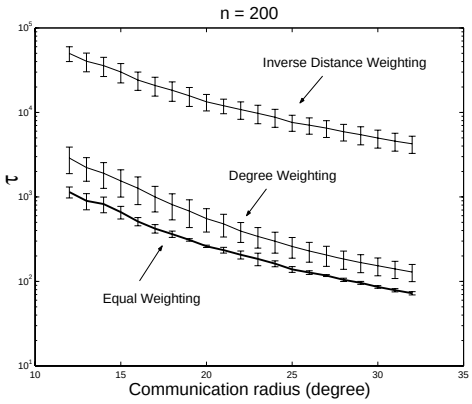


Figure 3.7. Time constants for the degree weighting and inverse-distance weighting schemes. Note that the time constant is plotted in log scale, so these two bad schemes have much larger time constants.

described above, Figure 3.7 shows the performance of some bad heuristics, specifically weighting by inverse distance or by the sum of the degrees of the two nodes. Simulation shows that time constants under these schemes are consistently worse than that of the equal weighting case.

3.3. The Benefit of Long-range Connections

Adding a few long-range connections (LRCs) to a network is well known to greatly reduce the average path length between nodes in a grid, making it into a small world [**48**, **82**] when enough are added, usually a small percentage of the total number of edges in the grid. Given a specific set of LRCs on the grid, we can use the LMI formulation to evaluate the maximum benefit that can be obtained from LRCs.

Instead of adding new connections, we can also create a small world utilizing the "random rewiring" method introduced in [**82**]. This algorithm starts with a regular network $\{\mathcal{V}, \mathcal{E}\}$ on S^1, say Figure 3.1(a). For each edge $e_{ij} \in \mathcal{E}$, we then replace it by a new edge $e_{ij'}$ with probability p. Node j' is chosen uniformly at random from the nodes that were not previously connected to node i. An example network after rewiring is shown in Figure 3.1(b).

Watts and Strogatz [**82**] show that as we increase the parameter p, the average path of the network reduces significantly, turning the regular grid into a small world. Subsequent work by Olfati-Saber [**54**] revealed other key features of this small-world network: as p increases, λ_2 increases quickly while $\lambda_{\max}$ increases only a little. This property means the rewiring procedure will decrease the convergence time $\frac{\lambda_{\max}}{\lambda_2}$ as we increase the parameter p. Figure 3.8 shows the time constants for a ring network with 100 nodes and 20 neighbors per node, for a varying parameter p under equal weighting and inverse-degree weighting schemes. Note that the additional speedup by the heuristic weighting slowly takes effect as p increases. This is because the inverse-degree weighting heuristic works better for irregular networks, and particularly it reduces to equal weighting in the case of a regular lattice.

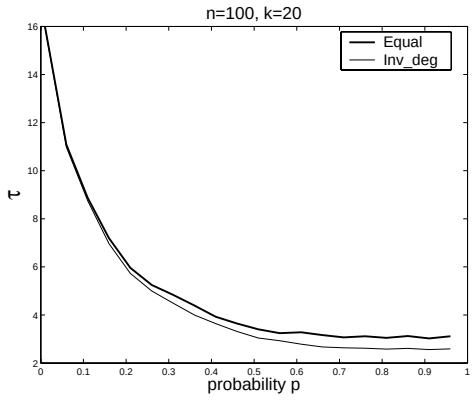


Figure 3.8. The speedup of network consensus through rewiring.

Boosting the communication radius of each agent, making the swarm heterogeneous by introducing some long-range connections, and "good" edge weighting each reduce the convergence time to consensus. The mobile network designer can trade these off against each other according to their cost to implement.

CHAPTER 4

Estimating and Maintaining the Communication Network Connectivity

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

4.1. Connectivity Measure

Given n mobile agents, we assume they can exchange information on an undirected communication network. For agent i, we denote its set of communication neighbors as $\mathcal{N}^i$. We denote the overall communication graph as G and the edge set as $E = \{(i,j) | \, j \in \mathcal{N}^i\}$.

The *adjacency* matrix $A \in \mathbb{R}^{n\times n}$ is defined as

$$A_{ij} = \begin{cases} A_{ji} > 0 & \text{if } j \in \mathcal{N}^i, \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

The degree of each node is $d_i = \sum_{j=1}^{n} A_{ij}$ or $d = A\mathbf{1}$ where $\mathbf{1}$ is a column vector of all ones. The *degree* matrix is defined as $D = \text{diag}(d)$, and the *weighted Laplacian* matrix of the graph is defined as $L = D - A$. The unweighted Laplacian matrix $\overline{L}$ can be treated as a special case where $A_{ij} = 1$. The spectral properties of $\overline{L}$ have been shown to be critical in many multiagent applications, such as formation control [**30**, **25**], consensus seeking [**56**] and direction alignment [**38**].

For the weighted Laplacian L, because we restrict the weights A_{ij} to be positive, the spectral properties of L are similar to those of $\overline{L}$ [**45**]. Specifically, we know

(1) $L\mathbf{1} = 0$.

(2) Given $\{\{\lambda_i\} | \, i = 1, \ldots, n\}$ as the spectrum of L, all the eigenvalues are real and they satisfy $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$, and $\lambda_2 > 0$ if and only if the graph is connected. As in the unweighted case, we call λ_2 the *algebraic connectivity* of the graph.

4.2. Centralized Power Iteration

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

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

4.2.1. Discrete-time Power Iteration

Given a square matrix Q and its eigenvalue spectrum satisfying $|\mu_1| < |\mu_2| \cdots < |\mu_n|$, *power iteration* is an established iterative method to compute the eigenvalue μ_n and its associated eigenvector v_n [**78**]. Now assume instead of μ_n, we are interested in its second-largest eigenvalue μ_{n-1}. If we already know μ_n and v_n, we can estimate μ_{n-1} by running the power iteration on the deflated matrix

$$\widetilde{Q} = Q - v_n v_n^T. \tag{4.2}$$

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

(1) Deflation on Q: $\widetilde{Q} = Q - v_n v_n^T$.

(2) Direction update: $x = \widetilde{Q} w$.

(3) Renormalization: $w = \frac{x}{\|x\|}$. Then go to step 2.

This power iteration method converges exponentially with time constant μ_{n-1}/μ_{n-2}. Once it converges, w is the eigenvector corresponding to the second largest eigenvalue μ_{n-1} of Q. In the case of repeated eigenvalues where $\mu_{n-1} = \ldots = \mu_{n-k+1} > \mu_{n-k}$, the

iteration converges in the ratio μ_{n-k}/μ_{n-1}. If $\mu_{n-1} = \ldots = \mu_1$, then any unit vector w is a solution.

4.2.2. Continuous-time Power Iteration

Inspired by the power iteration algorithm, we define a variant to find the second-smallest eigenvalue λ_2. To do this, we make two primary modifications to the algorithm. First, we modify the algorithm to run in continuous-time. Second, instead of performing the direction update step using a deflated matrix $\tilde{Q}$, we use a deflated matrix of $-L$. The deflation causes $-\lambda_2$ to be the leading (least negative) eigenvalue, and the negative sign results in convergence to the eigenvector v_2 corresponding to the eigenvalue with the minimum magnitude (as opposed to that with the maximum magnitude in the previous section).

Let $x = (x^1 \ \ldots \ x^n)^T \in \mathbb{R}^n$ be the estimate of the eigenvector v_2. The continuous-time algorithm has three parts:

(1) Deflation: $\dot{x} = -\text{Ave}(\{x^i\})\mathbf{1}$.

(2) Direction update: $\dot{x} = -Lx$.

(3) Renormalization: $\dot{x} = -(\text{Ave}(\{(x^i)^2\}) - 1)x$.

where the function $\text{Ave}(\{q^i\}) \triangleq (\sum_i q^i)/n$. Step 1 drives x to the null space of $\mathbf{1}$, i.e., the space spanned by the eigenvectors $\{v_2, \ldots, v_n\}$. For most initial conditions the direction update in step 2 drives x towards the eigenvector direction associated with the largest eigenvalue of $-L$, which is 0. But if the state x belongs to the null space of $\mathbf{1}$, the direction update step will keep x in the null space, and drive x towards the eigenvector direction

associated with the largest eigenvalue of the null space, which is $-\lambda_2$. Step 3 drives x towards the unit sphere.

In order to achieve the three steps simultaneously, we combine the three pieces in a linearly weighted fashion:

$$\dot{x} = -k_1 \text{Ave}(\{x^i\})\mathbf{1} - k_2 L x - k_3(\text{Ave}(\{(x^i)^2\}) - 1)x \tag{4.3}$$

where k_1, k_2, k_3 are scalar control gains. This equation can be rewritten as

$$\dot{x} = -\frac{k_1}{n}\mathbf{1}\mathbf{1}^T x - k_2 L x - k_3\Big(\frac{x^T x}{n} - 1\Big)x. \tag{4.4}$$

The weighted Laplacian matrix L is real symmetric, so it has an eigenvalue decomposition $L = T^T L^* T$ with $L^* = \text{diag}(0, \lambda_2, \ldots, \lambda_n)$ and T being an orthonormal matrix. It is easier to analyze system (4.4) under a new set of coordinates $y = (y^1 \ldots y^n)^T = Tx$ where both matrices L and $\mathbf{1}\mathbf{1}^T$ can be simultaneously diagonalized:

$$\dot{y} = -k_1 \text{diag}(1, 0, \ldots, 0)y - k_2 L^* y - k_3\Big(\frac{y^T y}{n} - 1\Big)y. \tag{4.5}$$

Denoting $\widetilde{L}^* = \text{diag}\{k_1/k_2, \lambda_2, \ldots, \lambda_n\}$, the system (4.5) can be rewritten as

$$\dot{y} = -k_2 \widetilde{L}^* y - k_3\Big(\frac{y^T y}{n} - 1\Big)y. \tag{4.6}$$

The following theorem shows that for suitable gain conditions on k_1, k_2 and k_3, system (4.3) is convergent from almost all initial conditions to an eigenvector $\tilde{v}_2$ corresponding to the eigenvalue $-\lambda_2$.

Theorem 4. *Given any initial condition $x(t_0)$ and positive gains $k_1, k_2, k_3 > 0$, as long as $y^2(t_0) \neq 0$, the gain conditions*

$$k_1 > k_2\lambda_2 \tag{4.7}$$

$$k_3 > k_2\lambda_2 \tag{4.8}$$

are necessary and sufficient for system (4.4) to converge to an eigenvector $\tilde{v}_2$ corresponding to the eigenvalue $-\lambda_2$ of the weighted Laplacian matrix $-L$ satisfying $\|\tilde{v}_2\| = \sqrt{n\left(\frac{k_3 - k_2\lambda_2}{k_3}\right)}$.

Proof. See the Appendix. □

Next we modify the continuous-time power iteration (4.3) so that it can be decentralized over the graph. In the decentralized algorithm, no single agent maintains an estimate of the entire eigenvector $\tilde{v}_2$; instead, agent i maintains the single component x^i of the network's estimate x of $\tilde{v}_2$. This is sufficient to maintain an estimate λ_2^i of λ_2.

4.3. Decentralized Power Iteration and Connectivity Estimation

To obtain a decentralized version of the power iteration algorithm, we first note that it is possible for each agent to satisfy the gain conditions (4.7) and (4.8) without knowing the graph topology. We know

$$\sum_i \lambda_i = \operatorname{trace}(L) = \sum A_{ij} \leq n(n-1) \max A_{ij}.$$

Additionally, in our edge weighting scheme introduced in Section 4.4, we have $A_{ij} \leq 1$. Therefore each agent can satisfy (4.7) and (4.8) by choosing $k_3, k_1 > n(n-1)k_2$ (assuming n is known to every agent).

Next we point out that the matrix iteration $\dot{x} = -Lx$ is a naturally decentralized operation, and its implementation only requires local communication.

The last obstacle to decentralize the continuous-time power iteration (4.3) is the averaging operation $\text{Ave}(\cdot)$. We can use the *PI average consensus estimator* [**31**] to decentralize this averaging operation. As there are two averaging functions in (4.3), we need two consensus estimators. Average consensus estimators allow n agents, each of which measures some time-varying scalar $\alpha^i(t)$, to compute an approximation of $\overline{\alpha}(t) = \frac{1}{n}\sum_i \alpha^i(t)$ using only local communication. We use the PI estimator introduced in Chapter 2.2.2:

$$\begin{aligned} \dot{z}^i =& \gamma(\alpha^i - z^i) - K_P \sum_{j\in\mathcal{N}^i} \left[z^i - z^j\right] \\ &+ K_I \sum_{j\in\mathcal{N}^i} \left[w^i - w^j\right] \end{aligned} \tag{4.9}$$

$$\dot{w}^i = - K_I \sum_{j\in\mathcal{N}^i} \left[z^i - z^j\right]. \tag{4.10}$$

Here z^i is the average estimate, $\gamma > 0$ is the rate new information replaces old information, $\mathcal{N}^i$ contains all one-hop neighbors of agent i in the communication network, and K_P, K_I are estimator gains. When the network is connected, the estimator error is $e^i(t) = y^i(t) - \frac{1}{n}\sum_{i=1}^n \alpha^i(t)$ for each agent i.

In the decentralized implementation of (4.3), agent i maintains a scalar x^i (which converges to the i-th component of the eigenvector $\tilde{v}_2$) and four consensus estimator states $\{z^{i,1}, w^{i,1}, z^{i,2}, w^{i,2}\}$ ($z^{i,1}$ and $z^{i,2}$ are agent i's estimates for $\text{Ave}(\{x^i\})$ and $\text{Ave}(\{(x^i)^2\})$

respectively) and receives from communication its neighbors' $\{x^j, z^{j,1}, w^{j,1}, z^{j,2}, w^{j,2}\}$ for all $j \in \mathcal{N}^i$. There are two ways to estimate the connectivity measure λ_2. First, noticing $-L\tilde{v}_2 = \lambda_2 v_2$, agent i can estimate λ_2 as

$$\lambda_2^i = -\frac{\sum_{j\in\mathcal{N}^i} L_{ij}x^j}{x^i} \tag{4.11}$$

when $x^i \neq 0$. This estimate is nonsmooth when x^i passes through zero, however. Therefore we use a second method, based on Theorem 4, which says that $z^{i,2} \to \frac{\|\tilde{v}_2\|^2}{n} = \frac{k_3 - k_2\lambda_2}{k_3}$. Agent i can therefore compute its estimate of λ_2 as

$$\lambda_2^i = \frac{k_3}{k_2}(1 - z^{i,2}). \tag{4.12}$$

Example 1: We simulated the eigenvalue estimation algorithm over the 5-node constant graph (Fig. 4.1), where the weights are set as $A_{ij} = 1$. The eigenvalue spectrum of its Laplacian matrix is $\{0, 0.83, 2.69, 4.00, 4.48\}$. The gains for the two PI average consensus estimators are $\gamma = 25, K_P = 50, K_I = 10$ and the gains for the eigenvector estimator are $k_1 = 6, k_2 = 1, k_3 = 20$, satisfying (4.7) and (4.8). Fig. 4.1 (b) shows the estimated λ_2^i for each node i.

4.4. Control to Maintain Connectivity

In this section we show how the connectivity estimator can be applied in a connectivity-control algorithm. We start by showing one additional property of λ_2.

Lemma 5. *Given any positively weighted graph G, λ_2 is a nondecreasing function of each weight A_{ij}.*

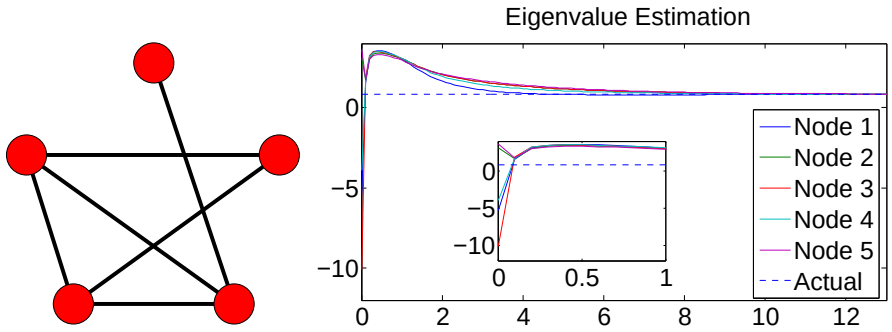


Figure 4.1. (a) A five-node network with all weights equal to 1. Nodes are counter-clockwise numbered from 1 to 5 starting from the top node. (b) Eigenvalue estimation through equation (4.12). The initial eigenvalue estimate for each agent is randomized. The inset plot shows the transient dynamics of the eigenvalue estimator.

Remark 1. *This lemma is easily demonstrated from the following equivalent definition of* λ_2*:*

$$\lambda_2 = \min_{x \perp \mathbf{1}, x \neq 0} \frac{x^T L x}{x^T x} = \min_{x \perp \mathbf{1}, x \neq 0} \frac{\sum_{(i,j) \in E} A_{ij} (x^i - x^j)^2}{x^T x}. \tag{4.13}$$

Based on this property, we can choose a weight function A_{ij} that is position-dependent. Then we can design connectivity-maintaining motion controllers, moving the agents to increase the connectivity of the network.

Given a scalar r as the maximal reliable inter-agent communication distance, one simple weighting choice is

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

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

We know $\lambda_2 > 0$ for connected graphs, and based on Lemma 5, λ_2 increases as the graph adds more links or as individual link weights increase as two agents come closer. We can design a gradient controller where each node moves to maximize λ_2, and this will in effect maintain the connectivity of a graph. The gradient controller in [**88**] was designed based on a similar idea. In that paper, each node moves to maximize the determinant of the deflated Laplacian matrix of a graph, in effect guaranteeing the algebraic connectivity λ_2 is bounded away from 0.

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

$$\begin{aligned} d\lambda_2 &= d(\hat{v}_2^T L \hat{v}_2) \\ &= d\hat{v}_2^T L v_2 + \hat{v}_2^T dL \hat{v}_2 + \hat{v}_2^T L d\hat{v}_2. \end{aligned} \tag{4.14}$$

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

$$\hat{v}_2^T L d\hat{v}_2 = d\hat{v}_2^T L v_2 = \lambda_2 d\hat{v}_2^T \hat{v}_2 = \frac{1}{2} d(\hat{v}_2^T \hat{v}_2) = 0. \tag{4.15}$$

Based on (4.14) and (4.15), the gradient controller for agent k is

$$u^k = \dot{p}^k = \frac{\partial \lambda_2}{\partial p^k} = \hat{v}_2^T \frac{\partial L}{\partial p^k} \hat{v}_2. \tag{4.16}$$

Next we replace the $\hat{v}_2$ in (4.16) with the $\tilde{v}_2$ in Theorem 4, which scales the control effort but does not change its direction:

$$u^k = \tilde{v}_2^T \frac{\partial L}{\partial p^k} \tilde{v}_2 = \sum_{(i,j)\in E} \frac{\partial A_{ij}}{\partial p^k} (\tilde{v}_2^i - \tilde{v}_2^j)^2. \tag{4.17}$$

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

$$\frac{\partial A_{ij}}{\partial p^i} = -A_{ij}(p^i - p^j)/\sigma^2 \qquad i \neq j \tag{4.18}$$

$$\frac{\partial A_{ij}}{\partial p^j} = A_{ij}(p^i - p^j)/\sigma^2 \qquad i \neq j \tag{4.19}$$

$$\frac{\partial A_{ii}}{\partial p^i} = 0 \tag{4.20}$$

$$\frac{\partial A_{ij}}{\partial p^k} = 0 \qquad k \neq i, j. \tag{4.21}$$

Plugging (5.1)-(5.4) into (4.17), we get

$$\begin{aligned} u^k &= \sum_{(k,j)\in E} \frac{\partial A_{kj}}{\partial p^k} (\tilde{v}_2^k - \tilde{v}_2^j)^2 \\ &= \sum_{(k,j)\in E} -A_{kj} (\tilde{v}_2^k - \tilde{v}_2^j)^2 \frac{p^k - p^j}{\sigma^2}. \end{aligned} \tag{4.22}$$

Compared to the eigenvector estimators (4.11) and (4.12), the implementation of (4.22) requires agent k to additionally obtain its neighbors' positions $\{p^j, j \in \mathcal{N}^i\}$. Agent k

approximates the exact $\tilde{v}_2^k, \tilde{v}_2^j$ with the estimates x^k, x^j, yielding the final control law:

$$u^k = \sum_{(k,j)\in E} -A_{kj}(x^k - x^j)^2 \frac{p^k - p^j}{\sigma^2}. \tag{4.23}$$

Example 2: We simulated the connectivity-maintaining algorithm over a randomly-generated six-node network. The communication radius is $r = 20$ and we set the threshold $\epsilon = 0.01$. In this network, the three big nodes are leaders. They all follow the same sinusoidal motion model $\dot{p}_x^i(t) = -0.2$, $\dot{p}_y^i(t) = 0.5\cos(p_x^i)$ with different initial configurations. The three small nodes run (4.23) to move along with the leaders and maintain graph connectivity.

The gains for the two average consensus estimators are $\gamma = 100, K_P = 50, K_I = 200$ and the gains for the eigenvector estimator are $k_1 = 18, k_2 = 3, k_3 = 60$. We choose the consensus and eigenvector estimator gains to approximately achieve a time-scale separation: the time constant of consensus estimation is significantly less than the time constant of eigenvector estimation, which is significantly less than the time constant of the motion controller. Fig. 4.2 shows four snapshots of these nodes during the motion and Fig. 4.3 shows the estimated λ_2^i of each node i during the motion. A video of the simulation is available at `http://lims.mech.northwestern.edu/projects/swarm/connect.wmv`.

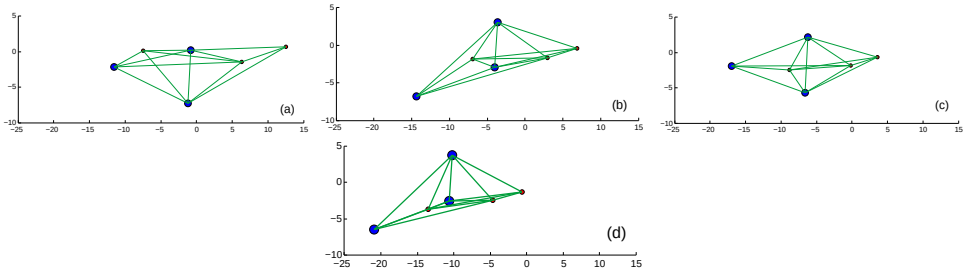


Figure 4.2. Snapshots of the agents during motion: $(a)\, t = 0$; $(b)\, t = 14$; $(c)\, t = 27$; $(d)\, t = 47$.

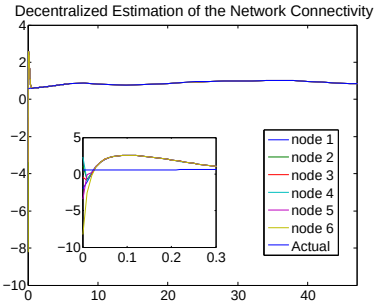


Figure 4.3. Each agent's estimate of the the graph connectivity λ_2 over time. All agents' estimates converge to the true algebraic connectivity of the graph within a few seconds.

CHAPTER 5

Formation Control

In this chapter we apply the decentralized estimation and control framework to the problem of controlling a group of mobile agents to achieve the desired formation statistics. We described two controllers, constructed based on the P consensus and the PI consensus estimator respectively. Through the simulation section we show the advantage of the PI estimator that this estimation procedure is robust to individual agent failures and has zero steady-steady error for constant or slowly-changing inputs. We also validate our approaches through physical experiments.

5.1. Formulation

Suppose a swarm consists of a collection of n mobile agents having positions $p_1, \ldots, p_n \in \mathbb{R}^m$, which we write also as the combined vector $p = [p_1^T \ldots p_n^T]^T \in \mathbb{R}^{mn}$. Then we can represent the collection of all possible swarm configurations as the topological coproduct $\mathfrak{P} \triangleq \coprod_{n=1}^{\infty} \mathbb{R}^{mn}$. We describe the desired swarm configuration using a vector *goal function* $f : \mathfrak{P} \to \mathbb{R}^\ell$. The primary objective of each agent is to move itself to an equilibrium position so that the final swarm configuration p satisfies $f(p) = f^\star$, where $f^\star \in \mathrm{Im}(f)$ is a goal vector known to each agent.

The total number n of agents in the swarm is unknown to each agent, although the agents may have knowledge of an upper bound on n. Each agent measures its own position and velocity and controls its own acceleration, $\ddot{p}_i = u_i$. Furthermore, each agent

can communicate with its neighbors; specifically, agents i and j can communicate with each other whenever $p_i \leftrightarrow p_j$, where $\leftrightarrow$ is a fixed symmetric relation on $\mathbb{R}^m$. For example, we may have $p_i \leftrightarrow p_j$ if and only if $|p_i - p_j| \leqslant r$, where r represents a fixed communication radius. Thus each configuration $p \in \mathfrak{P}$ defines the graph of an underlying communication network, and we let $\mathfrak{C} \subset \mathfrak{P}$ denote the set of all such configurations for which this graph is connected. As the agents move with time, the topology of this network can change, but we will perform our stability analysis below under the assumption that $p(t) \in \mathfrak{C}$, namely, that the network remains connected in forward time. For this reason we will assume $f^\star \in f(\mathfrak{C})$.

Our approach is based on following estimates of the gradient ∇J of the potential function $J : \mathfrak{P} \to \mathbb{R}$ given by

$$J(p) = \big[f(p) - f^\star\big]^T \Gamma \big[f(p) - f^\star\big], \tag{5.1}$$

where $\Gamma \in \mathbb{R}^{\ell \times \ell}$ is a suitably chosen symmetric positive-definite global gain matrix. We let the set

$$\mathrm{Crit}(J) \triangleq \big\{p \in \mathfrak{P} : \nabla J(p) = 0\big\} \tag{5.2}$$

denote the set of critical points of J, and we classify such points as "good" critical points where $f(p) = f^\star$ (these are the global minima of J) and "bad" critical points where $f(p) \neq f^\star$. We want the swarm to avoid getting stuck at bad critical points. Unfortunately, even if a bad critical point of a C^∞ potential J is a strict local maximum of J, it can still be a stable equilibrium of the associated gradient flow $\dot{p} = -\nabla J(p)$. For example, suppose

$J : \mathbb{R} \to \mathbb{R}$ is the C^∞ function

$$J(p) = \left[1 - \int_0^p x \, \exp\left(-\frac{1}{x^2}\right) \cos^2\left(\frac{1}{x^2}\right) dx\right]^2 . \tag{5.3}$$

This function $J(p)$ has a strict local maximum at $p = 0$, but this point is not isolated in $\mathrm{Crit}(J)$ and is in fact a stable equilibrium of the gradient flow. To rule out such pathological behavior, we will assume that J is locally constant on $\mathrm{Crit}(J)$.[1] This will indeed be the case for potentials of the form (5.1) when the goal function f is *subanalytic* (see the Appendix) or in particular when f is a polynomial function.

Our algorithms will guarantee that the swarm trajectories always converge to equilibrium sets.[2] For this reason we want all positive limit sets containing bad critical points of J to be "unstable" in the following sense:

Definition 6. *Let $\pi(t, x)$ be a continuous stationary flow on a topological space $\mathcal{X}$. A closed, π-invariant set $L \subset \mathcal{X}$ is* strongly unsteady *(respectively,* weakly unsteady*) when there exists an open set $\mathcal{O} \subset \mathcal{X}$ with $L \subset \mathcal{O}$ such that for any open set $\mathcal{U} \subset \mathcal{X}$ with $L \cap \mathcal{U} \neq \varnothing$ (respectively, with $L \subset \mathcal{U}$), there exists an initial state $x_0 \in \mathcal{U}$ and a time $T \geqslant 0$ such that $\pi(t, x_0) \in \mathcal{X} \setminus \mathcal{O}$ for all $t \geqslant T$.*

Clearly all strongly unsteady invariant sets are weakly unsteady, and the two notions coincide for equilibria. A weakly unsteady invariant set is both unstable (in the sense of Lyapunov) and unattractive, but the converse is not always true (for example, one can have an unstable, unattractive equilibrium which is not unsteady). If all positive limits

[1] We say that a function f on a topological space $\mathcal{X}$ is *locally constant on a set $S \subset \mathcal{X}$* when every $x \in S$ has an open neighborhood $N \subset \mathcal{X}$ such that f is constant on $N \cap S$.

[2] Note that convergence to an equilibrium set does not guarantee convergence to a single equilibrium, even in gradient systems.

sets containing bad critical points are strongly unsteady, then whenever a swarm trajectory approaches such a limit set, a small perturbation will cause it to leave a neighborhood of this set forever.

5.2. Moment Statistics

We focus on goal functions f of the form

$$f(p) = \frac{1}{n(p)} \sum_{i=1}^{n(p)} \phi(p_i)\,, \tag{5.4}$$

where $\phi : \mathbb{R}^m \to \mathbb{R}^\ell$ is a given *moment-generating function* and $n(p)$ is the unique integer n such that $p \in \mathbb{R}^{mn}$. For example, for $m = 3$ and $p_i = [p_{ix}\ p_{iy}\ p_{iz}]^T$, this function ϕ might be a list of ℓ monomials of the form

$$\phi(p_i) = p_{ix}^a\, p_{iy}^b\, p_{iz}^c\,, \tag{5.5}$$

where a, b, and c are nonnegative integers. In this case the goal function (5.4) is a list of ℓ *moments* of the form

$$M_{abc} = \frac{1}{n} \sum_{i=1}^{n} p_{ix}^a\, p_{iy}^b\, p_{iz}^c\,, \tag{5.6}$$

where the sum $a + b + c$ is called the *order* of the moment. Given a particular swarm formation, a sufficient number of moments is guaranteed to distinguish it from any other formation. In other words, moments can provide an exact formation description. We are interested, however, in the case where a small number of low-order moments is used to specify a family of formations. If ℓ moment constraints are specified on n robots in an m-dimensional space, in general there is an $(mn - \ell)$-dimensional algebraic set of

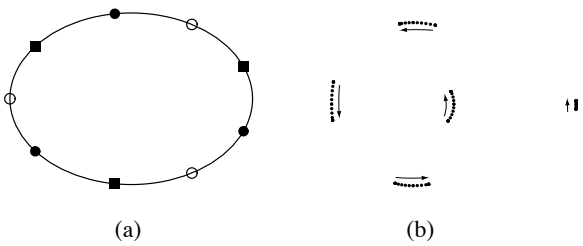


Figure 5.1. (a) For three robots in the plane ($mn = 6$), first- and second-order moment constraints ($\ell = 5$) restrict the swarm formation to a one-dimensional space. Shown are three example formations, where the robots in the same formation share the same symbol. The robots are confined to an ellipse determined by the constraints. (b) For five robots in the plane ($mn = 10$), first-, second-, and third-order moment constraints ($\ell = 9$) restrict the swarm formation to a one-dimensional space. The robots are shown moving along the constraint-preserving set.

swarm configurations that satisfy the constraints. Examples of one-dimensional families of swarm configurations are given in Figure 5.1. The structure and topology of such formation families can be studied using tools from real algebraic geometry [**9**]. Our primary example in this thesis involves formations defined by first- and second-order moments. The m first-order moments specify the center of mass of the swarm. From the $m(m+1)/2$ second-order moments we can derive $m(m-1)/2$ variables describing the orientation of orthogonal principal axes of inertia of the swarm and m shape variables summarizing the elongation of the swarm along the principal axes. Our abstraction of the swarm formation, then, is given by the $m(m+1)/2$ group variables describing the position and orientation of the principal axis frame in $SE(m)$ and the m shape variables describing the elongation of the swarm along these axes [**5**].

To write the moment-generating function ϕ for first- and second-order moments, let vech : $\mathbb{R}^{m\times m} \to \mathbb{R}^{m(m+1)/2}$ denote the linear map which stacks the main and upper

diagonals of a matrix into a vector, so that for $m = 3$,

$$\text{vech}\begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} \end{bmatrix} = \begin{bmatrix} \alpha_{11} \\ \alpha_{22} \\ \alpha_{33} \\ \alpha_{12} \\ \alpha_{23} \\ \alpha_{13} \end{bmatrix}. \tag{5.7}$$

Then ϕ for first- and second-order moments can be written as

$$\phi(p_i) = \begin{bmatrix} p_i \\ \text{vech}(p_i p_i^T) \end{bmatrix} \in \mathbb{R}^\ell, \tag{5.8}$$

where $\ell = m(m+3)/2$.

Given a closed set of swarm configurations $\mathcal{D} \subset \mathfrak{P}$ and a goal vector $f^\star \in f(\mathcal{D})$, we let $\mathcal{G}(f^\star, \mathcal{D})$ denote the cone of all symmetric positive definite matrices Γ such that no bad critical points of J in $\mathcal{D}$ are local minima of J. To reduce the risk of the swarm getting stuck at a bad critical point of J, we would ideally choose a weighting matrix Γ belonging to $\mathcal{G}(f^\star, \mathcal{D})$ for a large set $\mathcal{D}$ (i.e., one containing all likely swarm configurations). However, it may be difficult to find such matrices Γ for general goal functions f. Nevertheless, for the case of formations defined by first- and second-order moments with ϕ as in (5.8), we can always compute members of $\mathcal{G}(f^\star, \mathcal{D})$ when $\mathcal{D}$ contains all possible configurations of at least $m + 1$ agents:

Theorem 7. *Let ϕ be as in (5.8), let $\mathcal{D} = \bigcup_{n=m+1}^{\infty} \mathbb{R}^{mn}$, and let $f^\star \in f(\mathcal{D})$. Then there exists a symmetric matrix $\Gamma > 0$ such that for every bad critical point $p \in \mathcal{D}$ of J, the Hessian matrix $\mathcal{H}J(p)$ has at least one strictly negative eigenvalue. In particular, $\Gamma \in \mathcal{G}(f^\star, \mathcal{D})$.*

The proof of this theorem, which is constructive, is provided in the Appendix.

5.3. Nonlinear Gradient Control with High-Pass Estimators

In the notation of Section 2.1, the physical state of agent i is $x_i = [p_i^T \;\; \dot{p}_i^T]^T$, with dynamics

$$\dot{x}_i = F(x_i, u_i) = \begin{bmatrix} \dot{p}_i \\ u_i \end{bmatrix} \tag{5.9}$$

and noise-free measurements

$$z_i = C(x_i, \mathcal{P}_i^{\text{sens}}) = \begin{bmatrix} x_i \\ f^\star \end{bmatrix}. \tag{5.10}$$

We have already completed the first step in the design methodology of Section 2.1.2 by choosing the cost J in (5.1) with $f(p)$ in (5.4). According to the second step, we choose an initial (unimplementable) local controller $u_i = K^{\text{initial}}$ based on the gradient of this cost, with an additional damping term:

$$\begin{aligned} u_i &= K^{\text{initial}}(p, \dot{p}_i, f^\star) \\ &= -B\dot{p}_i - \big[\mathcal{J}\phi(p_i)\big]^T \Gamma \big[f(p) - f^\star\big], \end{aligned} \tag{5.11}$$

where $B \in \mathbb{R}^{m\times m}$ is a damping matrix and $\mathcal{J}\phi(\cdot)$ denotes the $\ell \times m$ Jacobian matrix of ϕ. Here $f(p)$ represents global information unavailable to each agent, so according to the third step in our methodology, we design signal generator G and a global state estimator Q and R to provide local estimates y_i of the global variable $f(p)$. In this section we consider the P estimator in 2.2.1:

$$s_i = G(x_i, z_i, \eta_i, y_i, \mathcal{S}_i) = \begin{bmatrix} p_i \\ y_i \end{bmatrix} \tag{5.12}$$

$$\begin{aligned} \dot{\eta}_i &= Q(x_i, z_i, \eta_i, y_i, \mathcal{S}_i) \\ &= -\gamma\eta_i - \sum_{j\neq i} a(p_i, p_j)\big[y_i - y_j\big] \end{aligned} \tag{5.13}$$

$$y_i = R(x_i, z_i, \eta_i, \mathcal{S}_i) = \eta_i + \phi(p_i)\,. \tag{5.14}$$

Here $y_i(t) \in \mathbb{R}^\ell$ is the agent's current estimate of $f(p)$ and $\eta_i(t) \in \mathbb{R}^\ell$ is the internal estimator state. To implement this estimation algorithm, each agent i must continually transmit its current values of p_i and y_i to its neighbors, as indicated by the signal generator (5.12). In the estimator dynamics (5.13), $\gamma \geqslant 0$ is an estimator "forgetting factor" and $a : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ is a C^1 symmetric function[3] such that $\mathrm{supp}(a) \subset \mathrm{Graph}(\leftrightarrow)$ (so that $a(p_i, p_j) \neq 0$ only if agents i and j can communicate with each other).

The fourth and final step in our design is to construct the actual local control law K by replacing $f(p)$ in (5.11) with y_i and adding a stabilizing nonlinear damping term:

$$u_i = K(x_i, z_i, \eta_i, y_i, \mathcal{S}_i)$$

[3]If $\mathcal{X}$ and $\mathcal{Y}$ are nonempty sets, we say that a function $\psi : \mathcal{X}\times\mathcal{X} \to \mathcal{Y}$ is *symmetric* when $\psi(a, b) = \psi(b, a)$ for all $a, b \in \mathcal{X}$.

$$= -B\dot{p}_i - \big[\mathcal{J}\phi(p_i)\big]^T \Gamma \big[y_i - f^\star\big] - \big[\mathcal{J}\phi(p_i)\big]^T \Lambda \big[\mathcal{J}\phi(p_i)\big]\dot{p}_i \,, \tag{5.15}$$

where $\Lambda \in \mathbb{R}^{\ell\times\ell}$ is a damping gain matrix. The utility of the extra nonlinear damping term is apparent in the proof (in the Appendix) of Theorem 8, stated below. We now proceed to investigate the behavior of this scheme (5.12)–(5.15).

Suppose the number n of agents in the swarm is fixed. We let $\mathbf{1}_n \in \mathbb{R}^n$ denote the vector of n ones (or simply $\mathbf{1}$ when n is clear from context), and we let $\mathrm{Orth}(\mathbf{1})$ denote the collection of $n \times (n-1)$ matrices S such that $S^TS = I$ and $S^T\mathbf{1} = 0$ (namely, the columns of S form an orthonormal basis for $\mathrm{span}\{\mathbf{1}\}^\perp$). Then by orthogonal decomposition,

$$I = SS^T + \frac{\mathbf{1}\mathbf{1}^T}{n} \tag{5.16}$$

and thus $ASS^TA^T \leqslant AA^T$ for any n-column real matrix A (in particular we have $|AS|_F \leqslant |A|_F$ where $|\cdot|_F$ denotes the Frobenius norm). We define the *Laplacian* $L(p) \in \mathbb{R}^{n\times n}$ to be the symmetric matrix whose off-diagonal elements in row i, column j are equal to $-a(p_i, p_j)$ and whose diagonal elements are the negatives of the sums of the off-diagonal elements in the same row (so that $L(p)\mathbf{1} \equiv 0$). Moreover, fixing some $S \in \mathrm{Orth}(\mathbf{1})$, we define the *reduced Laplacian* $L^\star(p)$ to be the $(n-1)\times(n-1)$ symmetric matrix

$$L^\star(p) = S^TL(p)S \,, \tag{5.17}$$

and we note from (5.16) that $SL^\star(p) = L(p)S$. Furthermore, for a connected configuration $p \in \mathfrak{C}$ and for positive estimator weights $a(\cdot,\cdot)$ on the connected arcs, the smallest eigenvalue of the reduced Laplacian $L^\star(p)$ (called the *algebraic connectivity* of the underlying

graph) will be strictly positive [**23**]. The first of our two primary assumptions we use to prove our convergence results is that this eigenvalue is bounded away from zero, i.e., that

$$L^\star(p) \geqslant \varepsilon I \tag{5.18}$$

along trajectories in forward time for some constant $\varepsilon > 0$. In particular, (5.18) implies that $p(t) \in \mathfrak{C}$ for all $t \geqslant t_0$. The second of our two primary assumptions takes the form of the small-gain condition

$$\lambda_{\max}(\Gamma) < \frac{\varepsilon}{2}\,\lambda_{\min}\big(\Lambda + \Lambda^T\big)\,, \tag{5.19}$$

where $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ denote the maximum and minimum eigenvalues (respectively). To better understand these conditions (5.18) and (5.19), suppose the estimator gain function $a(\cdot,\cdot)$ in (5.13) is simply

$$a(p_i, p_j) = \begin{cases} a_0 & \text{when } \; p_i \leftrightarrow p_j \\ 0 & \text{otherwise,} \end{cases} \tag{5.20}$$

where $a_0 > 0$ is a scalar constant estimator gain (technically, $a(\cdot,\cdot)$ would be a C^1 approximation to this simple choice). Then for a connected network, the value of ε is bounded from below by $2a_0 - 2a_0\cos(\pi/n)$, its value for the worst-case configuration of a linear chain of agents [**23**]. Thus if we know an upper bound $n_{\max}$ on the total number n of agents in the swarm, then we can choose our gains to satisfy

$$\lambda_{\max}(\Gamma) < a_0\Big[1 - \cos\big(\frac{\pi}{n_{\max}}\big)\Big]\,\lambda_{\min}\big(\Lambda + \Lambda^T\big)\,. \tag{5.21}$$

In this case we will always satisfy (5.18) and (5.19) provided the network remains connected. The inequality (5.21) involves the gradient-descent control gain Γ, the estimator gain a_0, and the nonlinear damping gain Λ. Because of the possibility of noise and delay in the communication channels, we would not choose the estimator gain a_0 to be too large, which means we would satisfy (5.21) by moving slowly enough, either by choosing a small control gain Γ or a large damping gain Λ.

The following theorem states our results for the case that $\gamma = 0$ and that each state η_i has the initial value $\eta_i(t_0) = 0$; the more general cases will be discussed below.

Theorem 8. *Suppose ϕ is C^2 and proper, fix $f^\star \in f(\mathfrak{C})$, suppose $B + B^T > 0$, suppose $\eta_i(t_0) = 0$ for each i, and suppose the weighting $a(\cdot,\cdot)$ is C^1 and symmetric. Suppose n is fixed, suppose* (5.18) *and* (5.19) *hold for some $\varepsilon > 0$, and fix $\gamma = 0$. Then each trajectory of the swarm system* (5.12)–(5.15) *is bounded in forward time, and its positive limit set L^+ consists of equilibria. If in addition ϕ is subanalytic and there exists a closed set $\mathcal{D} \subset \mathfrak{P}$ such that $\Gamma \in \mathcal{G}(f^\star, \mathcal{D})$ and $p(t) \in \mathcal{D}$ for all $t \geqslant t_0$, then every positive limit set L^+ containing a bad equilibrium (i.e., an equilibrium corresponding to a bad critical point of J) is strongly unsteady.*

In particular, if we choose ϕ as in (5.8) to include all first- and second-order moments, if we assume $n \geqslant m+1$, and if we choose Γ and $\mathcal{D}$ according to Theorem 7, then clearly ϕ is C^2, proper, and subanalytic, $\Gamma \in \mathcal{G}(f^\star, \mathcal{D})$, and $p(t) \in \mathcal{D}$ for all $t \geqslant t_0$. In this case the swarm will generically converge to the set of configurations satisfying the desired moment statistics, leaving any bad configuration after a slight perturbation.

As is evident in the proof of this theorem in the Appendix, the dynamics of the estimator (5.13)–(5.14) include a subsystem of the form $\dot{\chi} = -\gamma\chi$ which is uncontrollable from the inputs $\phi(p_i)$ but observable through the estimation errors $e_i = f(p) - y_i$. If $\gamma = 0$ and if the states $\eta_i(t_0)$ are not initialized to zero, then the constants χ will generate persistent nonzero constant offsets in the error variables e_i. These steady-state estimation errors will cause the swarm to converge to a formation with the wrong statistics. To avoid such errors, one would have to somehow globally simultaneously reinitialize these states to zero whenever agents leave the swarm (e.g., due to failure) or new agents join the swarm. Furthermore, if $\gamma = 0$ then any additive noise in the communication channels will pass through pure integrators $\dot{\chi} =$ noise, resulting in random drift in the estimation errors. To alleviate these problems one could choose $\gamma > 0$; in this case any incorrect initialization of the states $\eta_i(t_0)$ will be asymptotically forgotten, and communication noise will not cause random drift. However, the estimator (5.13)–(5.14) exhibits steady-state error under constant inputs, an error whose size is proportional to $\gamma/(\gamma + \varepsilon)$ (and hence nonzero for $\gamma > 0$) [**31**]. Nevertheless, as we will illustrate in Section 5.5, a small error due to a small positive γ may be preferable to errors caused by incorrect initializations. In the next subsection we introduce a more complex estimator which achieves robustness to initialization errors and adding or subtracting agents from the network but does so without introducing any steady-state error.

The conclusion of Theorem 8 (and likewise of Theorem 9 below) remains valid if the damping matrix B is a C^1 function of the states $x_1, \ldots, x_n$ and $\eta_1, \ldots, \eta_n$, provided $B(\cdot) + B^T(\cdot) > 0$ holds globally (however, keep in mind that B can only depend on local variables, i.e., variables available to each agent via sensing or communication). Hence we

can view this damping matrix B as an additional source of control, and we might design it to help maintain network connectivity or to help avoid collisions between agents. This extension is a topic of future research.

5.4. Nonlinear Gradient Control with PI Estimators

In this section we assume that there exists a proper metric d on $\mathbb{R}^m$ such that the quantity

$$\mathfrak{d}(n) \triangleq \sup_{p \in \mathfrak{C} \cap \mathbb{R}^{mn}} \max_{1 \leqslant i,j \leqslant n} d(p_i, p_j)\,, \tag{5.22}$$

which is the maximum diameter of a connected swarm of n agents, is finite for every n. For the case in which $p_i \leftrightarrow p_j$ if and only if $|p_i - p_j| \leqslant r$, where $r > 0$ is a fixed communication radius, we can take d to be the usual Euclidean metric on $\mathbb{R}^m$. It follows from (5.22) that there exists a class-$\mathcal{K}$ function $\mathfrak{a}$ and a C^1 function $\zeta : \mathbb{R}^m \to \mathbb{R}$ such that

$$|\phi(p_i) - \phi(p_j)|^2 \leqslant \mathfrak{a}\big(\mathfrak{d}(n(p))\big) \cdot \zeta(p_i) \tag{5.23}$$

for every $p \in \mathfrak{C}$ and every $i, j \in \{1, \ldots, n(p)\}$ [**29**, Corollary A.15].

The agent dynamics, measurements, and initial local controller are as before in (5.9), (5.10), and (5.11), but now we use the PI estimator introduced in Section 2.2.2:

$$s_i = \begin{bmatrix} p_i \\ \eta_i \end{bmatrix}, \qquad \text{where } \eta_i = \begin{bmatrix} v_i \\ w_i \end{bmatrix} \tag{5.24}$$

$$\begin{aligned}\dot{v}_i = &-\gamma v_i - \sum_{j\neq i} a(p_i,p_j)\big[v_i - v_j\big] \\ &+ \sum_{j\neq i} b(p_i,p_j)\big[w_i - w_j\big] + \gamma\phi(p_i)\end{aligned} \tag{5.25}$$

$$\dot{w}_i = -\sum_{j\neq i} b(p_i,p_j)\big[v_i - v_j\big] \tag{5.26}$$

$$y_i = v_i\,. \tag{5.27}$$

Here $\gamma > 0$ is a global forgetting factor which controls the rate of replacing old information with new (with $\gamma = 0$ no longer allowed as it now scales the input to the estimator), and $a, b : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ are bounded C^1 symmetric functions such that $\mathrm{supp}(a) \cup \mathrm{supp}(b) \subset \mathrm{Graph}(\leftrightarrow)$. We also assume that b has bounded first-order partial derivatives.

The actual control law for use with this new PI estimator is similar to the one in (5.15) but includes an additional nonlinear damping term:

$$u_i = -B\dot{p}_i - \big[\mathcal{J}\phi(p_i)\big]^T\Gamma\big[y_i - f^\star\big] - \big[\mathcal{J}\phi(p_i)\big]^T\Lambda\big[\mathcal{J}\phi(p_i)\big]\dot{p}_i - c\zeta(p_i)\dot{p}_i\,, \tag{5.28}$$

where B and Λ are damping gain matrices as before and $c > 0$ is a new scalar nonlinear damping gain. We now proceed to investigate the behavior of this scheme (5.24)–(5.28).

Again, suppose n is fixed. We define the *proportional Laplacian* $L_P(p) \in \mathbb{R}^{n\times n}$ to be the symmetric matrix whose off-diagonal elements in row i, column j are equal to $-a(p_i,p_j)$ and whose diagonal elements are such that $L_P(p)\mathbf{1} \equiv 0$. We define the *integral Laplacian* $L_I(p) \in \mathbb{R}^{n\times n}$ in the same way but using $b(\cdot,\cdot)$ instead of $a(\cdot,\cdot)$. Again fixing $S \in \mathrm{Orth}(\mathbf{1})$, we define the corresponding reduced Laplacians $L_P^\star(p) = S^T L_P(p) S$ and $L_I^\star(p) = S^T L_I(p) S$. Our first primary assumption we use to prove our convergence results

is that there exist constants $\rho > -\gamma$ and $\varepsilon > 0$ such that

$$\rho I \leqslant L_P^\star(p) \leqslant \bar{\rho} I \tag{5.29}$$

$$\varepsilon I \leqslant L_I^\star(p) \leqslant \bar{\varepsilon} I \tag{5.30}$$

along trajectories in forward time (again implying a connected network $p(t) \in \mathfrak{C}$). Here the constants $\bar{\rho}, \bar{\varepsilon} > 0$ represent upper bounds on the reduced Laplacians which exist because the functions a and b are bounded. Notice that ρ need not be a positive number; in particular, the choice $a(\cdot, \cdot) \equiv 0$ results in $L_P^\star(\cdot) \equiv 0$ which satisfies (5.29) with $\rho = 0$. Such a choice simplifies the estimator without changing our convergence results, but it might adversely impact performance.

Our second primary assumption takes the form of the small-gain condition

$$\lambda_{\max}(\Gamma) < \delta_1 \lambda_{\min}\big(\Lambda + \Lambda^T\big) \leqslant \delta_2 c\,, \tag{5.31}$$

where $\delta_1, \delta_2 > 0$ are scalar constants depending on n, ρ, $\bar{\rho}$, ε, $\bar{\varepsilon}$, γ, and the bounds on the partial derivatives of b (the exact dependencies are provided in the proof of Theorem 9 in the Appendix). As before, if an upper bound on n is known, then we can compute gains Γ, Λ, and c which satisfy (5.31).

Theorem 9. *Suppose ϕ is C^2 and proper, fix $f^\star \in f(\mathfrak{C})$, suppose $B + B^T > 0$, and suppose $a(\cdot, \cdot)$ and $b(\cdot, \cdot)$ are C^1, bounded, symmetric, and such that b has bounded first-order partial derivatives. Suppose n is fixed, suppose* (5.29) *and* (5.30) *hold for some $\rho > -\gamma$ and $\varepsilon > 0$ (with $\gamma > 0$), and suppose the inequalities* (5.31) *are satisfied. Then each trajectory of the swarm system* (5.24)–(5.28) *is bounded in forward time, and its*

positive limit set L^+ consists of equilibria. If in addition ϕ is subanalytic and there exists a closed set $\mathcal{D} \subset \mathfrak{P}$ such that $\Gamma \in \mathcal{G}(f^\star, \mathcal{D})$ and $p(t) \in \mathcal{D}$ for all $t \geqslant t_0$, then every positive limit set L^+ containing a bad equilibrium is strongly unsteady.

Like the high-pass estimator (5.13)–(5.14) with $\gamma = 0$, the PI estimator (6.21)–(5.27) (with $\gamma > 0$) includes a subsystem of the form $\dot{\chi} = 0$ which is uncontrollable from the inputs $\phi(p_i)$ (see the proof in the Appendix). Thus, as before, χ might be nonzero due to inconsistent initializations and might drift due to communication noise. However, unlike the high-pass case, these states χ are not observable through the estimation errors $e_i = f(p) - y_i$, which means their behavior will not affect the swarm dynamics.

5.5. Simulation Results

We simulated the algorithms in Sections 5.3 and 5.4 for a swarm of $n = 7$ planar robots ($m = 2$), ϕ as in (5.8), and $f^\star = [0 \;\; 0 \;\; 50 \;\; 0 \;\; 50]^T$. The controller gain matrix was $\Gamma = \mathrm{diag}(80, 80, 8, 8, 8)$. The estimator gain functions where chosen according to an equal weighting scheme with a communication radius of 15: $a(p_i, p_j) = a_0$ and $b(p_i, p_j) = b_0$ when $|p_i - p_j| \leq 15$ and $a(p_i, p_j) = b(p_i, p_j) = 0$ otherwise (the fact that these gain functions are discontinuous had little effect on the simulations). Also, we set the nonlinear damping gains Λ and c in (5.15) and (5.28) to zero as the constant B provided adequate damping over a bounded region.

We first simulated the high-pass scheme of Section 5.3 with damping $B = 40I$, estimator gain $a_0 = 20$, and no forgetting factor ($\gamma = 0$). Figure 5.2 shows the results of the inertial moments $M_{10} = \mathrm{CMx}$ (the first component of f) and $M_{02} = \mathrm{Ixx}$ (the fifth component of f). The first 25 seconds show the convergence of the formation statistics to

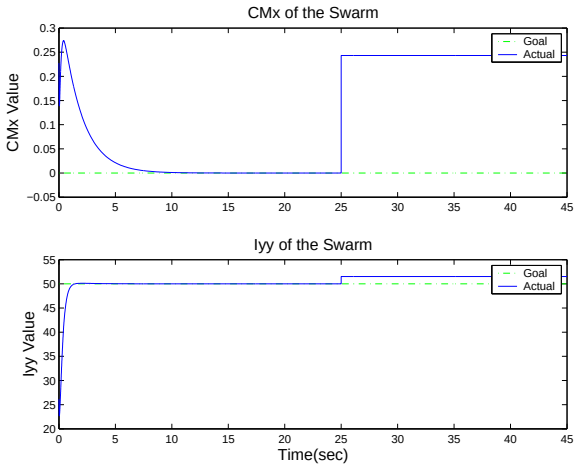


Figure 5.2. The high-pass algorithm with no forgetting factor ($\gamma = 0$).

their desired values with no steady-state error. At time $t = 25$, one of the agents fails and leaves the swarm, resulting in a permanent nonzero steady-state error after that point. Actually, the remaining agents do not move at all from their equilibria after time $t = 25$, demonstrating that the high-pass estimator with no forgetting factor does not recover from initialization errors. If we include a nonzero forgetting factor of $\gamma = 0.3$, then we do recover from the loss of the agent (Figure 5.3), but we now incur a small nonzero steady-state error both before and after the loss.

We next simulated the PI scheme of Section 5.4 with increased damping $B = 100I$, estimator gains $a_0 = 20$ and $b_0 = 0.2$, and $\gamma = 6$. Figures 5.4 and 5.5 show that the PI algorithm can also recover from the loss of an agent (again at time $t = 25$) but now with zero steady-state error.

5.6. Performance Increase through Optimized Communication Weights

To illustrate the effect of improved estimation speed, we consider a formation control example with 6 robots. We assume the robots maintain a fixed communication topology

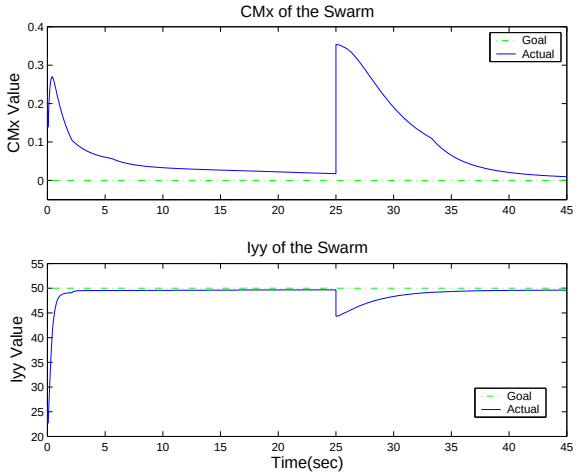


Figure 5.3. The high-pass algorithm with forgetting factor $\gamma = 0.3$.

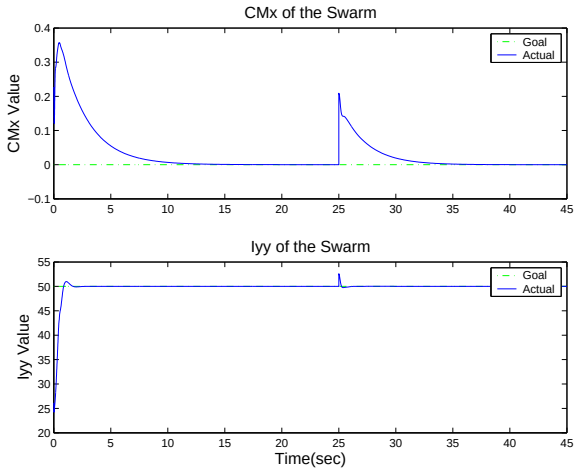


Figure 5.4. The PI algorithm.

as they move, shown in Figure 3.4. Figure 5.6 shows the convergence of the x-coordinate of the swarm center of mass from the same initial condition for estimators using the equal weighting strategy and the inverse-degree weighting heuristic. The performance is significantly improved using the heuristic weighting strategy.

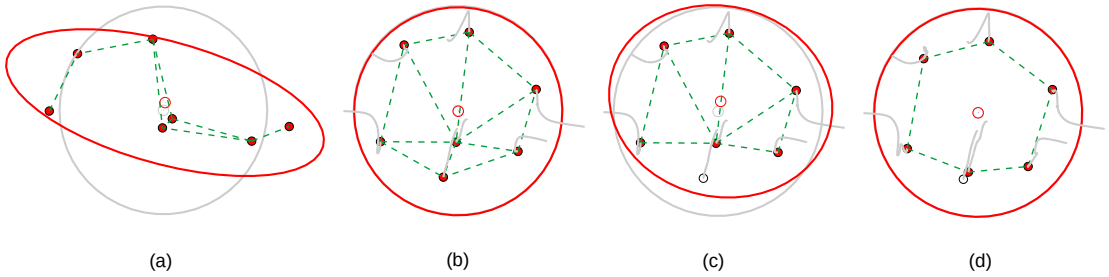


Figure 5.5. Snapshots of the robots' movement under the PI scheme: (a) $t = 0$ initial condition, (b) $t = 25$ robots satisfy the goal, (c) $t = 25$ one robot dies, (d) $t = 45$ robots move to re-satisfy the goal.

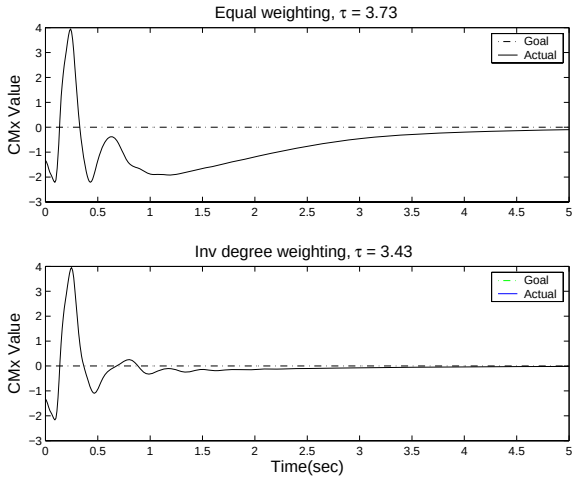
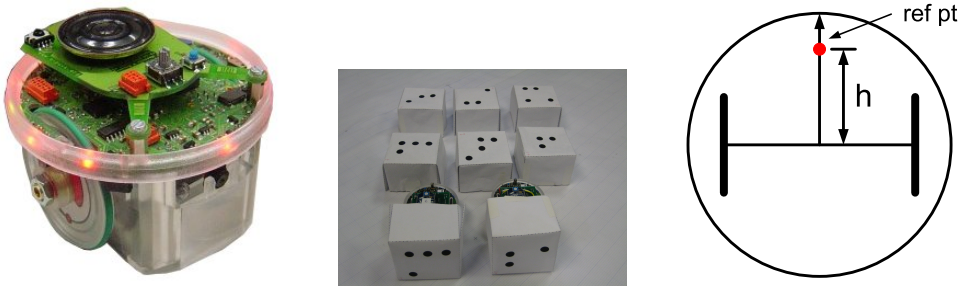


Figure 5.6. Convergence of CMx with different weighting schemes. For this particular network topology, the encoded Laplacian has time constant $\tau = 3.73$ under equal weighting and $\tau = 3.43$ under inverse degree weighting.

5.7. Physical Experiments

5.7.1. Hardware and Software Description

We use a group of e-puck robots designed by EPFL [**13**] as our test bed. Each e-puck is a two-wheel differential drive vehicle powered by a 3.6V, 1.4Ah Lion battery. It has two stepper motors each with 20 steps per revolution and a gear reduction ratio of 50 : 1



(a) (b) (c)

Figure 5.7. (a) The e-puck mobile robot developed by EPFL. (b) For vision localization system, each e-puck is associated with a uniquely-identifiable pattern. (c) The gradient controller effectively controls the velocity of the reference point (the offset $h = 35.0mm$), and this reference velocity is translated to the individual wheel velocity through a motion controller (described in Section 5.7.3.

(see Fig. 5.7 (a)). The e-puck uses a 16 bit PIC30F6014A microprocessor (144k program space and 8k data RAM), and it can communicate with other e-pucks through bluetooth or Xbee® communication. During the experiment, we use the 802.15.4 based Xbee® communication, at a baud rate of 115200.

On the software side, each e-puck is driven by two interrupts. $Timer1$ generates a high level ISR at 2.5Hz: this routine handles obstacle avoidance and the high level estimation and control calculation (Fig. 5.8). Dead reckoning is calculated each time the stepper motor steps. $UART2$ handles the low level ISR, and within this routine the robot receives the packet from other e-pucks and preprocesses the data for the consensus calculation. Each data packet contains thirteen 32 bit floating-point numbers representing the ten estimator states, and two 32 bit floating-point numbers representing the reference point position of the e-puck. This low level ISR is designed to run as fast as possible.

To offset the position drift caused by the odometry error, each e-puck is covered by a white box with a uniquely-identifiable grid pattern on top (See Fig. 5.7 (b)). A camera

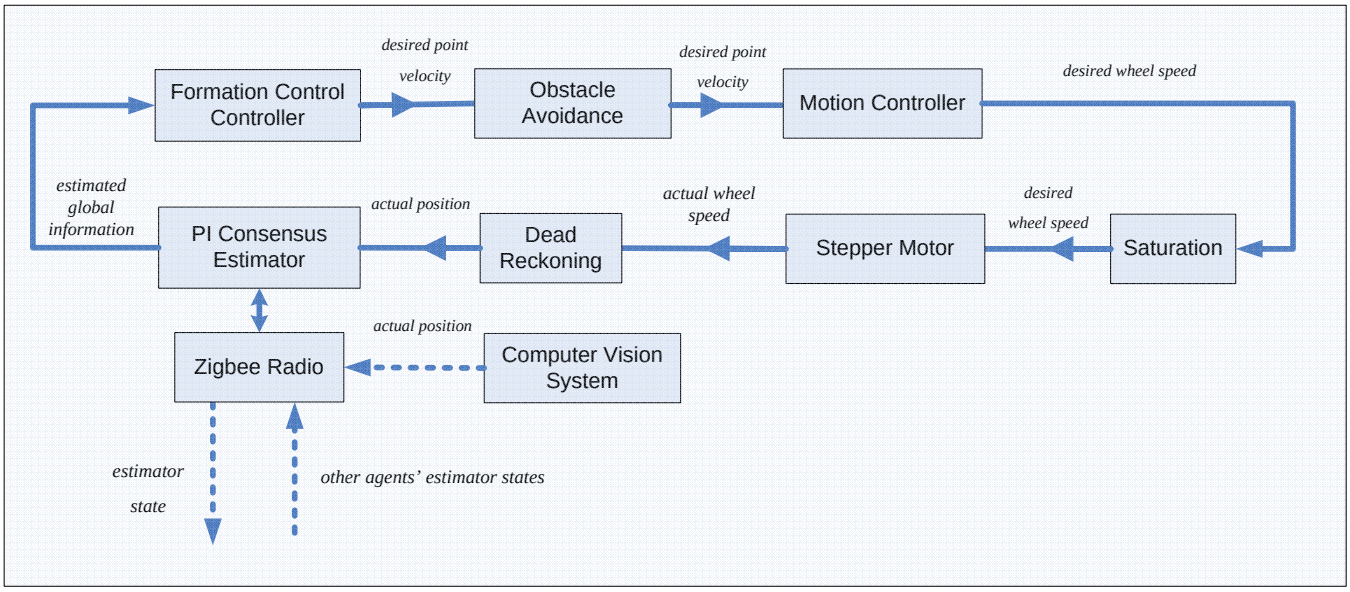


Figure 5.8. A block diagram of the control systems on each e-puck.

system (See Fig. 5.9), overseeing all grid patterns, sends out continuous images to a central PC which runs computer vision algorithms to track the positions of all the e-pucks. The PC sends out the updated position information to each e-puck at approximately 1Hz.

5.7.2. First-order Formation Control Controller

The original formation control controller we developed in Section 5.4 is for second-order systems. In this experiment, we implement a first-order version of this controller:

$$u_i = \Big[I + \big[\mathcal{J}\phi(p_i)\big]^T \Lambda \big[\mathcal{J}\phi(p_i)\big]\Big]^{-1} \big[\mathcal{J}\phi(p_i)\big]^T \Gamma \big[f^\star - y_i\big] \tag{5.32}$$

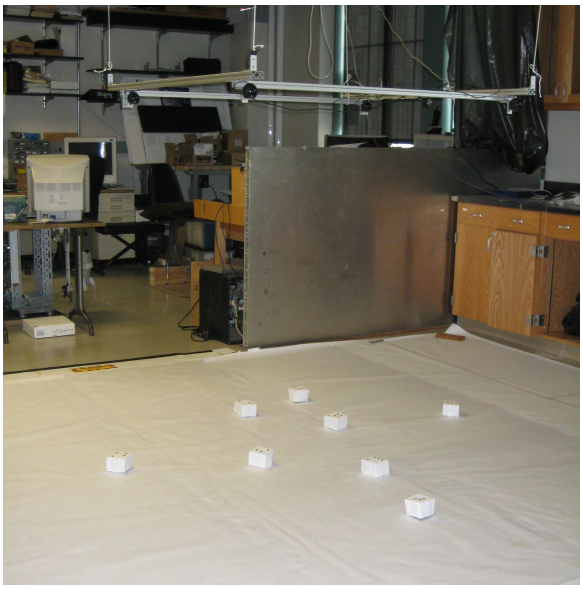


Figure 5.9. The camera system, consisting of four overhead Logitech webcams, can cover a 3m by 3m workspace. After calibration, the camera system can recognize a static pattern anywhere in the workspace within an error bound of 2.

where y_i is the output of the PI consensus estimator introduced in Section 2.2.2. We rewrite the estimator equations here for easy reference:

$$\begin{aligned}
\dot{v}_i &= -\gamma v_i - \sum_{j \neq i} a\big[v_i - v_j\big] + \sum_{j \neq i} b\big[w_i - w_j\big] + \gamma \phi(p_i) \\
\dot{w}_i &= -\sum_{j \neq i} b\big[v_i - v_j\big] \\
y_i &= v_i \,.
\end{aligned}$$

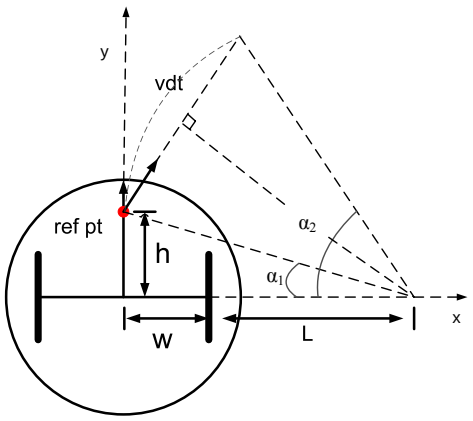


Figure 5.10. Schematic of the motion control algorithm. In between the mainloop interrupts, the gradient controller computes the desired speed for each wheels based on the desired ending position of the reference point.

5.7.3. Motion Controller Implementation

The gradient controller described in the last section computes a desired velocity for the reference point (See Fig. 5.10). The motion controller translates this velocity into individual wheel speed based on the desired end position of the reference point before the next mainloop interrupt.

Assuming the desired velocity for the reference point is $v = (v_x, v_y)$ (in the local frame) and the time between two interrupts is dt. Then in between two interrupts the desire angle of rotation for the reference point is

$$\theta = \alpha_2 - \alpha_1 \tag{5.33}$$

where

$$\alpha_1 = \arctan(\frac{h}{w + L}) \tag{5.34}$$

$$\alpha_2 = \arctan(\frac{h + v_y dt}{w + L - v_x dt}) \tag{5.35}$$

In order to find the center of rotation, we need to solve for L. Based on the geometry, we know that

$$h^2 + (w + L)^2 = (h + v_y dt)^2 + (w + L - v_x dt)^2 \tag{5.36}$$

which gives

$$2w + 2L - v_x dt = \frac{v_y}{v_x}(2h + v_y dt) \tag{5.37}$$

Based on (5.37), we can solve for the angles α_1 and α_2:

$$\alpha_1 = \arctan(\frac{2hv_x}{2hv_y + v^2 dt}) \tag{5.38}$$

$$\alpha_2 = \arctan(\frac{2v_x(h + v_y dt)}{2v_y(h + v_y dt) - v^2 dt}). \tag{5.39}$$

Then the left and right wheel speed can be solved as

$$v_l = \frac{\alpha_2 - \alpha_1}{dt}(2w + L) \tag{5.40}$$

$$v_r = \frac{\alpha_2 - \alpha_1}{dt} L \tag{5.41}$$

where the value of L is given in (5.37).

5.7.4. Obstacle Avoidance Implementation

Each packet the e-puck broadcast out contains its own reference point position, so by receiving the packets from other agents each e-puck can keep track of the variable *dist*,

the distance from itself to its nearest neighbor. $dist$ is computed as the distance between the reference point positions of the two e-pucks. When $dist$ is smaller than a predefined safe distance $SAFEDIST$, the e-puck changes its heading direction to the right of it nearest neighbor.

In the implementation, we design the turning rate to be related to the variable $dist$: the smaller $dist$ is, the faster the e-puck turns. Each e-puck has a radius of 35mm, so two e-pucks may collide into each other when $dist \leq 120$mm. Based on this, we defined a distance-based weighting function

$$wt = \frac{SAFEDIST - dist}{SAFEDIST - 120} \tag{5.42}$$

when $dist > 120$ and $wt = 1$ when $dist < 120$. If θ_{act} is the current heading direction and θ_0 is the collision-safe direction (See Fig. 5.11), θ_{act} is updated as

$$\theta_{act} \leftarrow \theta_{act} - (\theta_{act} - \theta_0) wt_{wt}^{\alpha} \tag{5.43}$$

where $0 < \alpha_{wt} \leq 1$ is a parameter used to tune the turning rate. An increase of the maximal speed allowed by each e-puck or a slow down of the main interrupt loop $Timer1$ may cause more collisions, and one can decrease α_{wt} to increase the turning rate. An increased tuning rate will help avoid more potential collisions, but too high turning rate will make the trajectory of the e-puck nonsmooth.

5.7.5. Experiment Setup and Results

Our formation control experiment has three phases. After initialized in random locations, eight e-pucks first move to achieve the desired statistics of $(100, 300, 130000, 60000, 120000)$.

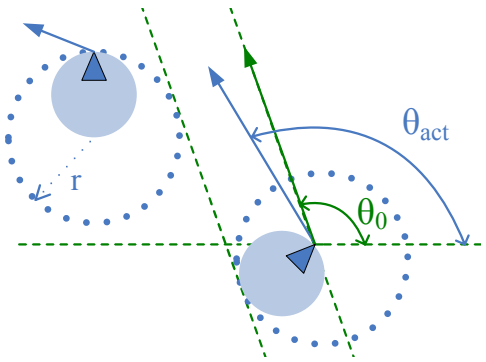


Figure 5.11. Schematic of the obstacle avoidance algorithm. Each e-puck is drawn as a solid circle with a triangle indicating the reference point. Because each e-puck only knows the reference point position of its nearest neighbor, to the e-puck its neighbor can be anywhere within a circle with radius $r = 70$mm, which is the diameter of an e-puck.

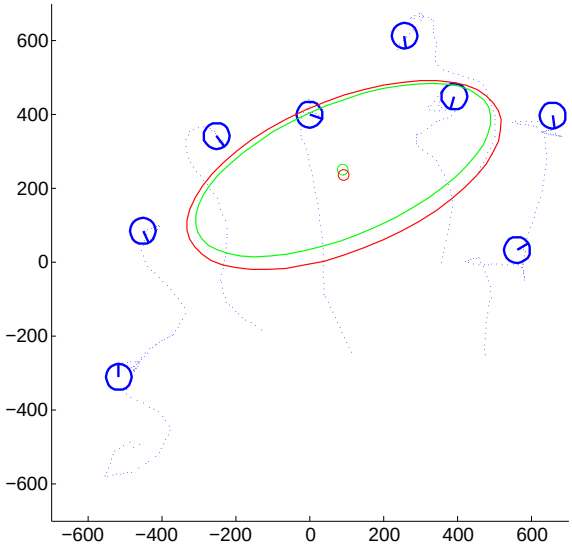


Figure 5.12. Trajectories of the e-pucks for the first 100 seconds during phase 1.

After they have satisfied these statistics, at approximately $t = 500$sec we let them move again to satisfy a different set of statistics $(100, -200, 130000, 20000, 80000)$ (See Fig. 5.12). After the group of e-pucks satisfy the new statistics again, at approximately

$t = 570$sec, we randomly turned off two e-pucks, the remaining six e-pucks move again to re-satisfy the statistics $(100, -200, 130000, 20000, 80000)$.

As mentioned before, since the gradient-based swarm controller is developed for point-mass system, in the experiment we are effectively controlling the velocity of the reference point of the e-puck, and this desired velocity is translated to the two desired wheel velocities. We set our controller gains as $\Gamma = \text{diag}(8, 8, 0.01, 0.01, 0.01)$. We set the communication mode for each e-puck to be broadcasting, and within this workspace range all e-pucks form a all-to-all communication network. The estimator gains (Section 5.7.2) are $a_0 = 0.23, b_0 = 0.03, \gamma = 0.02$.

As shown in Fig. 5.13, the PI dynamic consensus estimator is able to track the average of all agents' inputs. Moreover, the feedback system is stable and the agents reach the group equilibrium which satisfies the desired moment statistics.

There are a number of factors that limit the performance of the system. It is desirable to speed the system up, but the maximum rotating speed of the wheel is limited to 1 revolution per second and the main loop interrupt time can not be less than 0.4 second due to the current hardware constraint (See Fig. 5.8 for the main loop tasks). One potential idea for future improvement is to use state-dependant control gains, so that the wheel velocities are always saturated when the actual group statistics is still far from the desired group statistics. In terms of accuracy of the system, encoder noise causes errors in dead reckoning, and the central vision system itself has position errors around 2cm as well. In addition, packet error in the wireless communication system causes additional transient in the consensus estimation and delays the correct convergence of the overall system. Due to these numerous error sources, each e-puck jitters when the group statistics

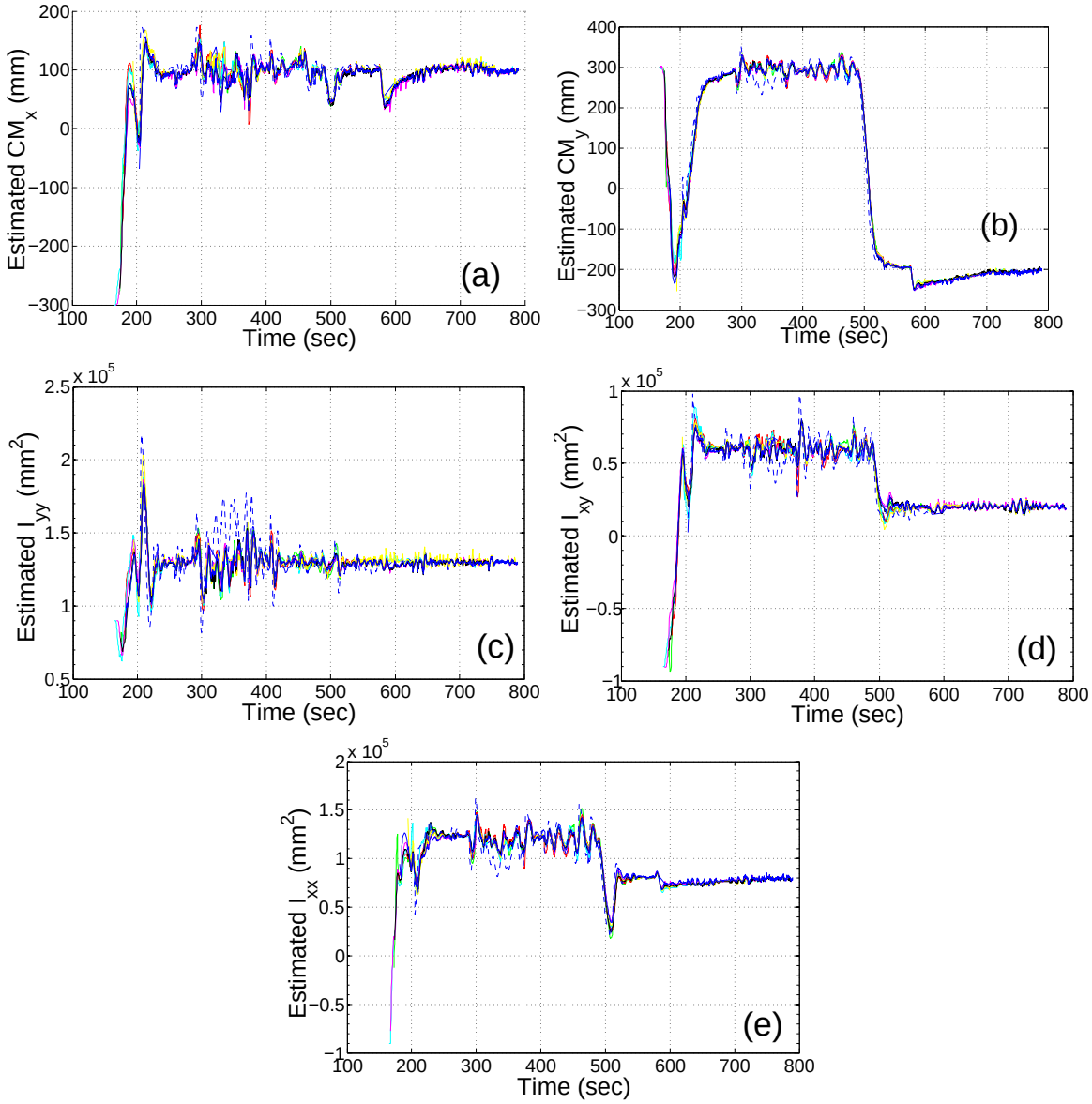


Figure 5.13. Estimated statistics from all e-pucks.

is close to the target statistics. We can further add in a dead-band or hysteresis in the control gains to reduce the jittering behavior.

CHAPTER 6

Target Localization

In this chapter we will apply the decentralized estimation and control framework to the problem of localizing a moving target with a mobile sensor network. Each agent maintains a target estimate, fuses its own estimate with its neighbors estimates, and moves so as to maximize the expected information from its sensor, relative to the current uncertainty in the estimate.

6.1. Formulation

6.1.1. Measurement Model

We consider n sensors and one target moving in the plane, having positions $p_1, \cdots, p_n \in \mathbb{R}^2$ and $x_t \in \mathbb{R}^2$, respectively. The observation made by the i^{th} sensor is given by

$$z_i = H_i x_t + v_i \,, \qquad i = 1, \ldots, n \,, \tag{6.1}$$

where the measurement noise v_i is a continuous-time Gaussian noise with zero mean. This measurement model can include several different types of sensors, and in this thesis we focus on range-bearing sensors and range-only sensors as illustrative examples.

In a standard linear range-finding sensor model [**63**], [**12**], $H_i = I_2$ (the 2×2 identity matrix) and its covariance matrix R_i assumes a diagonal structure in the sensor's local

range/bearing frame:

$$R_i = \begin{bmatrix} (\sigma^i_{\text{range}})^2 & 0 \\ 0 & (\sigma^i_{\text{bearing}})^2 \end{bmatrix}. \tag{6.2}$$

The range measurement noise variance $(\sigma^i_{\text{range}})^2$ is commonly represented by a function $f_r(r_i)$ of the distance r_i from the target to sensor i. The bearing noise variance $(\sigma^i_{\text{bearing}})^2$ also depends on the range and can be modeled as $f_b(r_i)$. We use the following simple yet representative forms of these functions:

$$(\sigma^i_{\text{range}})^2 = f_r(r_i) = a_2(r_i - a_1)^2 + a_0 \tag{6.3}$$

$$(\sigma^i_{\text{bearing}})^2 = f_b(r_i) = \alpha f_r(r_i)\,, \tag{6.4}$$

where a_0, a_1, a_2, α are model parameters. This measurement uncertainty model assumes the existence of a "sweet spot" location $r_i = a_1$ at which the noise is at its minimum value. In practice, when the target is out of the sensing range, we can initialize the diagonal entries of R_i to be ∞.

If the sensor being used takes a nonlinear measurement of the state, we will use its linearized approximate model. For example, given a range-only sensor i:

$$z_i = \|x_t - p_i\|_2 + v_i \tag{6.5}$$

with the Gaussian noise level $R_i = f_r(r_i)$ (as in (6.3)), we can linearize it around the point $x_{t0} = (x_0, y_0)$:

$$\widetilde{z_i} = -H_i x_t + v_i \tag{6.6}$$

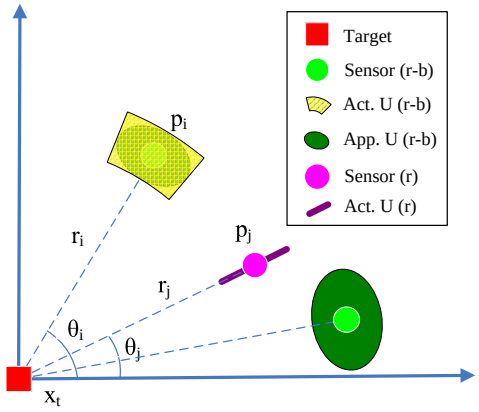


Figure 6.1. Schematic of the measurement models for range-bearing sensors (r-b) and range-only sensors (r). For the range-bearing sensor, the segment of the annulus shows the one-sigma uncertainty of each sensor's estimate and the ellipse is the approximation that we use.

with

$$
\begin{aligned}
H_i &= \begin{bmatrix} \frac{p_i^x - x_0}{\sqrt[2]{(x_0-p_i^x)^2+(y_0-p_i^y)^2}} & \frac{p_i^y - y_0}{\sqrt[2]{(x_0-p_i^x)^2+(y_0-p_i^y)^2}} \end{bmatrix} \\
&= \begin{bmatrix} \cos(\theta_{i0}) & \sin(\theta_{i0}) \end{bmatrix} \qquad (6.7)
\end{aligned}
$$

and $\widetilde{z}_i = z_i - ||x_{t0} - p_i||_2 - H_i x_{t0}$ is our modified measurement to take into account the linearization effect. Here both H_i and $\widetilde{z}_i$ can be obtained by sensor i locally. In this thesis, we do not deal with the sensor i's self-localization error and assume p_i can be measured perfectly.

6.1.2. Gradient Controller Design

We consider two different ways of fusing the local target position measurements z_i and error covariances R_i to obtain a global target position estimate $\hat{x}_{\text{global}}$ and global error covariance P_{global}. The first method, described in Section 6.2 and based on the work

in [**12**], uses only current measurements to obtain $\hat{x}_{\text{global}}$ and P_{global}. The second method, described in Section 6.3, defines $\hat{x}_{\text{global}}$ and P_{global} by means of a Kalman filter. In either case, the matrix P_{global} depends on the sensor and target locations, which means the sensors can move to reduce the uncertainty P_{global}. To formulate a proper cost function, we can use either

$$J = \det(P_{\text{global}}) \tag{6.8}$$

or

$$J = \operatorname{tr}(P_{\text{global}}) \tag{6.9}$$

In optimal experiments theory [**26**], they are referred to as *D-optimal design* and *A-optimal design*, respectively. For simplicity, we assume all agents are kinematic and fully actuated so that $\dot{p}_i = u_i$, and we use the gradient controller

$$u_i = K^{\text{initial}}(\cdot) = -\Gamma T_i^T \begin{bmatrix} \dfrac{\partial J}{\partial r_i} \\ \dfrac{1}{r_i}\dfrac{\partial J}{\partial \theta_i} \end{bmatrix}, \tag{6.10}$$

where $\Gamma > 0$ is a gain matrix, $\theta_i = \angle(p_i - x_t)$ is the angle from the target to sensor i, and T_i is the rotation matrix

$$T_i = \begin{bmatrix} \cos(\theta_i) & \sin(\theta_i) \\ -\sin(\theta_i) & \cos(\theta_i) \end{bmatrix}. \tag{6.11}$$

We also use T_i to transform R_i, the covariance matrix in the local frame, to $T_i R_i T_i^T$, the covariance matrix in the global Cartesian frame. In the face of the nonlinear transformation from polar frames to Cartesian frames, this is a convenient approximation (Fig. 6.1). Furthermore, we define

$$P_i^r \triangleq \frac{\partial P_{\text{global}}}{\partial r_i}\,, \qquad\qquad P_i^\theta \triangleq \frac{\partial P_{\text{global}}}{\partial \theta_i}\,, \tag{6.12}$$

and use the following facts from matrix calculus [**10**]:

$$\frac{\partial}{\partial x} f(A(x)) = \text{tr}[\frac{\partial f}{dA}\frac{\partial A}{\partial x}] \tag{6.13}$$

$$\frac{\partial}{\partial A}\det(A) = |A|A^{-T} = |A|A^{-1} \tag{6.14}$$

$$\frac{\partial}{\partial A}\text{tr}(A) = I \tag{6.15}$$

$$\frac{\partial}{\partial x}A^{-1} = -A^{-1}(\frac{\partial A}{\partial x})A^{-1} \tag{6.16}$$

From above we can calculate the gradients in (6.10) as

$$\frac{\partial J}{\partial r_i} = J \cdot \text{tr}\big[P_{\text{global}}^{-1} P_i^r\big] \tag{6.17}$$

$$\frac{\partial J}{\partial \theta_i} = J \cdot \text{tr}\big[P_{\text{global}}^{-1} P_i^\theta\big] \tag{6.18}$$

when we use the D-optimal design (6.8) or the alternative form

$$\frac{\partial J}{\partial r_i} = \text{tr}\big[P_i^r\big] \tag{6.19}$$

$$\frac{\partial J}{\partial \theta_i} = \text{tr}\big[P_i^\theta\big] \tag{6.20}$$

when we use the A-optimal design (6.9).

In general the controller in (6.17)− (6.20) is centralized because $P_{\text{global}}, P_i^\theta, P_i^r$ each contains information from all sensors. We will obtain the implementable, decentralized local controller $u_i = K(\cdot)$ from (6.10) by replacing any unavailable global quantities with local estimates.

6.1.3. Distributed Estimator Design

In both the sensor fusion schemes in Sections 6.2 and 6.3, the sum of the information from each individual sensors is used to calculate the global information P_{global} (also P_i^θ, P_i^r). To have better noise suppression, we use *PI dynamic average consensus estimator* this time. For n agents, assume each agent i has an input $u_i(t) \in \mathbb{R}^{k\times r}$, internal states $v_i, w_i \in \mathbb{R}^{k\times r}$ and output $y_i = v_i$. We use the following simplified PI estimator form:

$$\begin{aligned} \dot{v}_i = &-\gamma v_i - K_p \sum_{j\in N_i} \big[v_i - v_j\big] \\ &+ K_i \sum_{j\in N_i} \big[w_i - w_j\big] + \gamma u_i \end{aligned} \tag{6.21}$$

$$\dot{w}_i = - K_i \sum_{j\in N_i} \big[v_i - v_j\big]. \tag{6.22}$$

The following two sections propose two generic sensor fusion schemes to obtain P_{global}. In each scheme, the explicit form of the derived motion controller depends on the choice of cost functions (D-optimal or A-optimal) and sensor models (range-bearing, range-only). In Sections 6.2 and 6.3 we derive these equations for range-bearing sensors. The more complicated case with the range-only sensors are dealt with in Section 6.5.

6.2. One-Time Measurement Approach

An instantaneous fusion of current sensor readings leads to the following relations [**12**, **63**]:

$$P_{\text{global}}^{-1}\hat{x}_{\text{global}} = \sum_{i=1}^{n} H_i^T (T_i R_i T_i^T)^{-1} z_i = \sum_{i=1}^{n} H_i^T T_i R_i^{-1} T_i^T z_i \tag{6.23}$$

$$P_{\text{global}}^{-1} = \sum_{i=1}^{n} H_i^T (T_i R_i T_i^T)^{-1} H_i = \sum_{i=1}^{n} H_i^T T_i R_i^{-1} T_i^T H_i \,, \tag{6.24}$$

We further use the rules in (6.13) – (6.16) to find P_i^r, P_i^θ:

$$\begin{aligned} P_i^r &= \frac{\partial}{\partial r_i}(\sum_{i=1}^{n} H_i^T T_i R_i^{-1} T_i^T H_i)^{-1} \\ &= -P_{\text{global}} \frac{\partial}{\partial r_i}(\sum_{i=1}^{n} H_i^T T_i R_i^{-1} T_i^T H_i) P_{\text{global}} \\ &= -P_{\text{global}} \frac{\partial}{\partial r_i}(H_i^T T_i R_i^{-1} T_i^T H_i) P_{\text{global}} \end{aligned} \tag{6.25}$$

and similarly

$$P_i^\theta = -P_{\text{global}} \frac{\partial}{\partial \theta_i}(H_i^T T_i R_i^{-1} T_i^T H_i) P_{\text{global}}. \tag{6.26}$$

For range-bearing sensors, we plug in $H_i = I_2$ and (6.3), (6.11) into (6.25), (6.26):

$$P_i^r = 2a_2(r_i - a_1) P_{\text{global}} T_i R_i^{-2} \begin{bmatrix} 1 & 0 \\ 0 & \alpha \end{bmatrix} T_i^T P_{\text{global}} \tag{6.27}$$

$$P_i^\theta = P_{\text{global}} \big(A_i + A_i^T\big) P_{\text{global}} \tag{6.28}$$

with

$$A_i = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} T_i R_i^{-1} T_i^T \,. \tag{6.29}$$

We implement a decentralized version of the resulting gradient control law (6.10) as follows. Each agent runs a PI average consensus estimator with local matrix input $u_i = nT_iR_i^{-1}T_i^T$ (for a total of 3 scalar estimators due to the symmetry of this 2×2 matrix), but with the unknown quantities r_i and θ_i replaced by the measurements

$$r_i \approx |p_i - z_i| \,, \qquad \theta_i \approx \angle(p_i - z_i) \,. \tag{6.30}$$

The inverse of the output of this estimator is P_i, the local estimate of P_{global}. Each agent runs a second average consensus estimator with local vector input $nT_iR_i^{-1}T_i^Tz_i$ (for a total of 2 scalar estimators), again with the replacements (6.30). The output of this second estimator, when multiplied by P_i, yields $\hat{x}_i$, the local estimate of $\hat{x}_{\text{global}}$. We now evaluate the expressions (6.8), (6.27), and (6.28) by replacing P_{global} with P_i and using the following filtered versions of the replacements (6.30):

$$r_i \approx |p_i - \hat{x}_i| \,, \qquad \theta_i \approx \angle(p_i - \hat{x}_i) \,. \tag{6.31}$$

These replacements lead to the decentralized version of the control law (E.10) with gradients (6.17), (6.18) The same approach applies when we use the control law (6.19), (6.20). This implementation assumes the sensor model parameters a_0, a_1, a_2, and α are known to each agent.

6.3. Kalman Filter Approach

The approach in Section 6.2 fuses sensor readings from current measurements only. To make use of past measurements as well, we can adopt a Kalman filter approach to defining $\hat{x}_{\text{global}}$ and P_{global}. We begin with a linear target model

$$\dot{x}_t = Fx_t + Gu_t + w\,, \tag{6.32}$$

where u_t is an exogenous input and w is a continuous-time Gaussian noise with zero mean and covariance matrix Q. We consider the centralized Kalman-Bucy filter [**74**]:

$$\dot{P}_{\text{global}} = FP_{\text{global}} + P_{\text{global}}F^T + Q - nP_{\text{global}}CP_{\text{global}} \tag{6.33}$$

$$\dot{\hat{x}}_{\text{global}} = F\hat{x}_{\text{global}} + Gu_t + nP_{\text{global}}(y - C\hat{x}_{\text{global}})\,, \tag{6.34}$$

where C and y are the fused measurements

$$C = \frac{1}{n}\sum_{i=1}^{n} H_i^T T_i R_i^{-1} T_i^T H_I\,, \qquad\qquad y = \frac{1}{n}\sum_{i=1}^{n} H_i^T T_i R_i^{-1} T_i^T z_i \tag{6.35}$$

and initial conditions are given by the one-time measurements (6.23) and (6.24). The partial derivatives in (6.12) can be obtained by taking partial derivatives on both sides of (6.33):

$$\begin{aligned}
\dot{P}_i^r &= FP_i^r + P_i^r F^T - nP_i^r CP_{\text{global}} - nP_{\text{global}}CP_i^r \\
&\quad + P_{\text{global}}\frac{\partial}{\partial r_i}(H_i^T T_i R_i^{-1} T_i^T H_i)P_{\text{global}} \\
\dot{P}_i^\theta &= FP_i^\theta + P_i^\theta F^T - nP_i^\theta CP_{\text{global}} - nP_{\text{global}}CP_i^\theta
\end{aligned} \tag{6.36}$$

$$+ P_{\text{global}} \frac{\partial}{\partial \theta_i} (H_i^T T_i R_i^{-1} T_i^T H_i) P_{\text{global}} \tag{6.37}$$

For range-bearing sensors, we plug in $H_i = I_2$ and (6.3), (6.11) into (6.36), (6.37):

$$\begin{aligned} \dot{P}_i^r &= F P_i^r + P_i^r F^T - n P_i^r C P_{\text{global}} - n P_{\text{global}} C P_i^r \\ &\quad + 2a_2(r_i - a_1) P_{\text{global}} T_i R_i^{-2} \begin{bmatrix} 1 & 0 \\ 0 & \alpha \end{bmatrix} T_i^T P_{\text{global}} \end{aligned} \tag{6.38}$$

$$\begin{aligned} \dot{P}_i^\theta &= F P_i^\theta + P_i^\theta F^T - n P_i^\theta C P_{\text{global}} - n P_{\text{global}} C P_i^\theta \\ &\quad + P_{\text{global}} \big(A_i + A_i^T \big) P_{\text{global}} \end{aligned} \tag{6.39}$$

with initial conditions calculated according to the one-time measurements (6.27) and (6.28).

We implement a decentralized version of the resulting gradient control law (6.10) as follows. Each agent runs two average consensus estimators, one with local matrix input $T_i R_i^{-1} T_i^T$ and local output C_i, and the other with local vector input $T_i R_i^{-1} T_i^T z_i$ and local output y_i. Each agent also maintains estimates P_i and $\hat{x}_i$ of P_{global} and $\hat{x}_{\text{global}}$ (respectively) by means of the differential equations

$$\dot{P}_i = F P_i + P_i F^T + Q - n P_i C_i P_i \tag{6.40}$$

$$\dot{\hat{x}}_i = F \hat{x}_i + G u_t + n P_i (y_i - C_i \hat{x}_i) \tag{6.41}$$

with initial conditions

$$P_i(0) = (T_i R_i T_i^T)(0)\,, \qquad \hat{x}_i(0) = z_i(0)\,. \tag{6.42}$$

Finally, each agent maintains local copies of the gradients P_i^r and P_i^θ (which we again name P_i^r and P_i^θ with a slight abuse of notation) by means of the differential equations

$$\begin{aligned}
\dot{P}_i^r &= \frac{\partial}{\partial r_i}(FP_i + P_iF^T + Q - nP_iC_iP_i) \\
&= FP_i^r + P_i^rF^T - nP_i^rC_iP_i - nP_iC_iP_i^r \\
&\quad + 2a_2(r_i - a_1)P_iT_iR_i^{-2}\begin{bmatrix} 1 & 0 \\ 0 & \alpha \end{bmatrix}T_i^TP_i \qquad (6.43) \\
\dot{P}_i^\theta &= \frac{\partial}{\partial \theta_i}(FP_i + P_iF^T + Q - nP_iC_iP_i) \\
&= FP_i^\theta + P_i^\theta F^T - nP_i^\theta C_iP_i - nP_iC_iP_i^\theta \\
&\quad + P_i\big(A_i + A_i^T\big)P_i \qquad (6.44)
\end{aligned}$$

with initial conditions given by (6.27) and (6.28) but with $P_i(0)$ replacing $P_{\text{global}}(0)$. In all of these equations we use the replacements (6.31), and we arrive at an implementable version of the local controller (E.10). This implementation assumes that F, G, Q, u_t, n, and the sensor model parameters a_0, a_1, a_2, and α are known to each agent.

6.4. Simulation Results

We use three range-bearing sensors starting from $(88.73, 106.76)$, $(89.05, 75.98)$, $(99.94, 77.93)$ and a moving target starting from $(100, 100)$. The dynamic model of the target is $\dot{x}_t = u_t + w$ with $u_t = \begin{bmatrix} 0.1 & 0.1 \end{bmatrix}^T$ and $Q = \text{diag}(0.05\ 0.05)$. The sensor model parameters are $a_0 = 0.3528$, $a_1 = 15.625$, $a_2 = 0.0008$, and $\alpha = 5$. Here we use a radius based communication model. The communication radius is set at $r = 50$ to guarantee the connectedness of the network. We choose a controller gain of $\Gamma = 20I$.

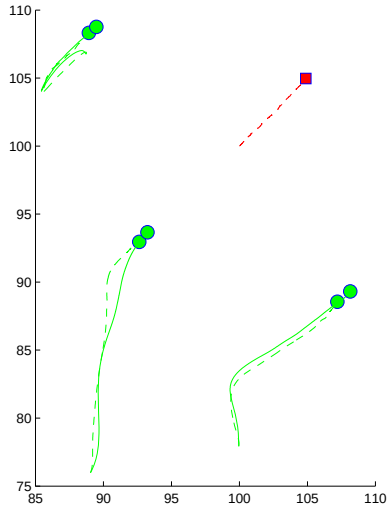


Figure 6.2. Trajectories of sensors with a moving target starting from $(100, 100)$. The solid lines denote the Kalman filter scheme and the dashed lines denote the one-time measurement scheme.

Figure 6.2 shows the actual trajectories of the sensors. In both sensor fusion schemes, the sensors space themselves from others by 60 degrees (relative to the target). Previous analysis shows this is the optimal collaborative configuration to do a one-measurement sensor fusion [**12**]. In Figure 6.3 we compare the performance of these decentralized algorithms with each other and with the centralized versions, where each sensor has access to the correct centralized computation of P_{global}. In both cases, the decentralized schemes recover the results of their centralized counterparts after an initial transient.

Figure 6.4 compares the performance of static and mobile sensor fusion schemes. Sensors start from the same positions, and in this simulation we changed the parameter a_2 of the sensor model to 0.5 to increase the spatial influence on the measurement noise level. We see that the moving sensors more quickly obtain accurate estimates of the target position. For control gains $\Gamma > 5I$, we start to see oscillatory behaviors in sensor motions and the dynamics of sensor 1's estimate looks almost the same as the case when $\Gamma = 5I$.

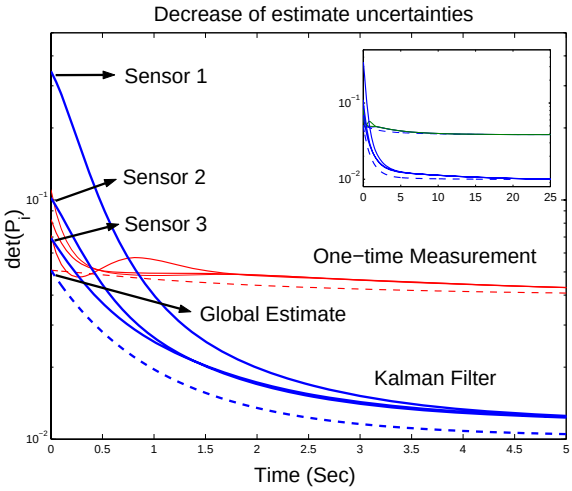


Figure 6.3. Comparison of the individual belief uncertainty matrices P_i. The centralized versions P_{global} for each scheme are shown as dotted lines.

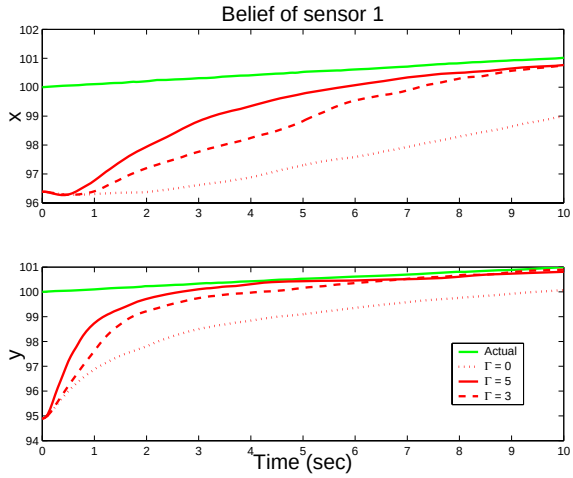


Figure 6.4. Comparison between static sensors (Γ=0) and mobile sensors ($\Gamma = 3I, 5I$).

6.5. Heterogeneous Sensors and Multiple Targets

6.5.1. Extension to Heterogeneous Sensors

Here we address the cases where range-bearing sensors need to collaborate with range-only sensors for the estimation task. The control laws for the range-bearing sensors remain as they are in (6.27), (6.28), (6.38), (C.20). All we need to do is to derive the controllers for those range-only sensors. We derive the explicit form of P_i^r, P_i^θ for the one-time measurement case, and the Kalman filter case can be done in a similar manner.

From the schematic of the linearized measurement model (Fig. 6.5), we have:

$$\cos\theta_{i0} = \frac{a + r_i\cos\theta_i}{\sqrt{(a + r_i\cos\theta_i)^2 + (r_i\sin\theta_i - b)^2}} \tag{6.45}$$

$$\sin\theta_{i0} = \frac{r_i\sin\theta_i - b}{\sqrt{(a + r_i\cos\theta_i)^2 + (r_i\sin\theta_i - b)^2}} \tag{6.46}$$

Then based on (6.25) we have

$$\begin{aligned} P_i^r &= -P_{\text{global}}\frac{\partial}{\partial r_i}(H_i^T T_i R_i^{-1} T_i^T H_i) P_{\text{global}} \\ &= -P_{\text{global}}\frac{\partial}{\partial r_i}(\frac{1}{R_i} H_i^T H_i) P_{\text{global}} \\ &= P_{\text{global}}(\frac{2a_2(r_i - a_1)}{a_2(r_i - a_1)^2 + a_0} H_i^T H_i - R_i^{-1}\begin{bmatrix} \zeta & \eta \\ \eta & -\zeta \end{bmatrix}) P_{\text{global}} \end{aligned} \tag{6.47}$$

with

$$\begin{aligned} \zeta &= \frac{-2\cos\theta_{i0}\sin\theta_{i0}}{||p_i - x_{t0}||_2}(\sin\theta_i\cos\theta_{i0} - \sin\theta_{i0}\cos\theta_i) \\ \eta &= \frac{\cos^2\theta_{i0} - \sin^2\theta_{i0}}{||p_i - x_{t0}||_2}(\sin\theta_i\cos\theta_{i0} - \sin\theta_{i0}\cos\theta_i) \end{aligned}$$

and similarly

$$
\begin{aligned}
P_i^{\theta} &= -P_{\text{global}} \frac{\partial}{\partial \theta_i} (H_i^T T_i R_i^{-1} T_i^T H_i) P_{\text{global}} \\
&= -P_{\text{global}} \frac{\partial}{\partial \theta_i} (\frac{1}{R_i} H_i^T H_i) P_{\text{global}} \\
&= -\frac{1}{R_i} P_{\text{global}} \begin{bmatrix} \zeta_2 & \eta_2 \\ \eta_2 & -\zeta_2 \end{bmatrix} P_{\text{global}} \qquad (6.48)
\end{aligned}
$$

with

$$
\begin{aligned}
\zeta_2 &= \frac{-2 r_i \cos\theta_{i0} \sin\theta_{i0}}{||p_i - x_{t0}||_2} (\sin\theta_i \sin\theta_{i0} + \cos\theta_i \cos\theta_{i0}) \\
\eta_2 &= \frac{r_i (\cos^2\theta_{i0} - \sin^2\theta_{i0})}{||p_i - x_{t0}||_2} (\sin\theta_i \sin\theta_{i0} + \cos\theta_i \cos\theta_{i0}).
\end{aligned}
$$

Now we can finish the distributed design by replacing P_{global} with P_i and using the approximation (6.30) or (6.31).

For range-only sensors, singularity issues can arise when implementing these control algorithms. This is because we need to use the invert the estimator output to calculate P_i and the estimator input $u_i = H_i^T T_i R_i^{-1} T_i^T H_i$ has 0 determinant. One solution is to let the estimator run a short time before inverting its output to calculate the control effort; the problematic matrix will become nonsingular when a range-only sensor fuses information with other sensors.

6.5.2. Extension to Multiple Targets

In the multiple targets scenario we only consider the case when each sensor is capable of taking multiple measurements at a time and able to distinguish different targets.

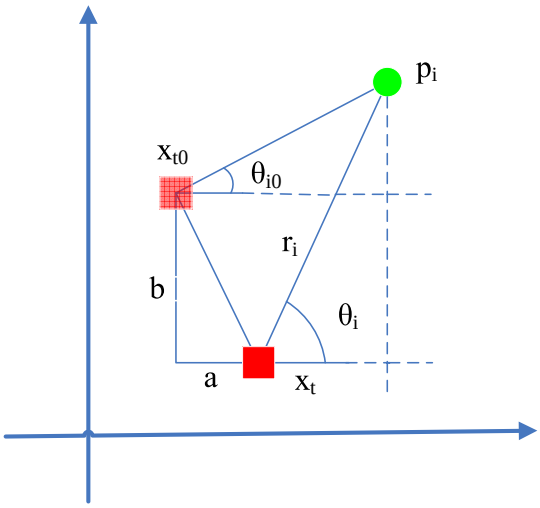


Figure 6.5. The linearized measurement model for range-only sensors.

Otherwise some dynamic *sensor scheduling* and *target association* algorithms need to be developed, which is outside the scope of this thesis.

Assume we have M targets in total. For each target j, the previous calculation gives a control vector u_{ij} for sensor i, and we can simply add them up in a weighted manner:

$$u_i = \sum_{j=1}^{M} w_{ij} u_{ij}, w_{ij} > 0, \sum_{j=1}^{M} w_{ij} = 1 \tag{6.49}$$

This approach will retain the distributed nature of the algorithm. In essence, this approach is the same as in [**12**], which took an algebraic approach by redefining the target state: $X_t = \begin{bmatrix} x_t^1 & \dots & x_t^M \end{bmatrix}$ and assume the measurement noise for each target is uncorrelated from others. In this approach the number of consensus estimators being used is proportional to the number of targets, and future research effort is needed to deal with this communication constraint.

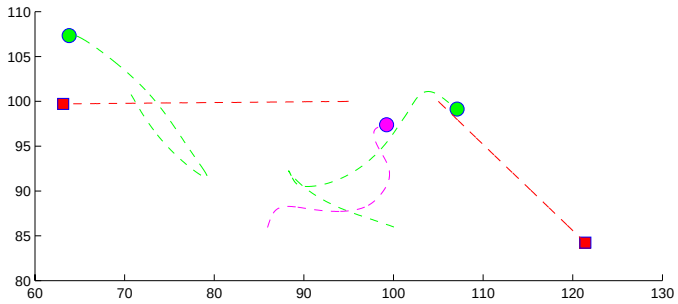


Figure 6.6. Two range-bearing sensors (green circle) and one range-only sensor (magenta circle) collaborate to track two moving targets (red square).

Here is an illustrative example: two different range-bearing sensors collaborate with a range-only sensor to track the trajectories of two moving targets. Mobile sensors start from $(70.7, 100.8)$, $(100.0, 86.0)$, $(86.0, 85.9)$ and their measurement models are given below:

	a_0^j	a_1^j	a_2^j	α_j
$j = 1$	0.1528	15.625	0.0008	5
$j = 2$	0.0166	10.800	0.0010	3
$j = 3$	0.1100	15.396	0.001	n/a

We set the control gain $\Gamma = 50I$, estimator gain $K_p = 50I, K_i = 0.5I$. The range-only sensor measurement model is linearized around the point $(100, 100)$. Figure 6.6 shows how sensors divide into groups to track individual targets.

CHAPTER 7

Conclusions and Future Work

7.1. Summary of the Thesis

Among other natural wonders, fish schools and bird flocks are amazing from an engineering design perspective. Within a fish school of enormous size, each fish plans its motion by only sensing its nearest neighbors' positions. The global behavior of the group emerges from each individual's local interactions with its neighbors, and this behavior is often termed self-organization.

In this thesis we describe a systematic way of designing emergent behaviors in sensing and communication networks of mobile agents. The problem is to design a control law to run on each agent, based on sensor and communication input, so that the desired collective behavior emerges. We find that a wide range of tasks, including network connectivity maintenance, swarm statistic control and cooperative target tracking, can be solved through our proposed *decentralized estimation and control* approach.

In Chapter 2 and 3, we describe the detailed design steps for each agent using our decentralized estimation and control approach. Each agent simultaneously estimates properties of the global behavior of the system by running dynamic consensus estimators and use those estimates in a modified gradient control law to guarantee feedback stability and accomplish the task. Furthermore, the extra design freedom in the estimator communication weights can be used to increase the overall convergence speed of the system.

For the rest of the thesis, we apply this methodology to solve three coordination tasks: (1) Maintaining the connectivity in a mobile sensor network; (2) Drive a group of robots to satisfy some desired first and second-order moment statistics and (3) Use a mobile sensor network to track a moving target. These designs are validated by both computer simulations and physical experiments. In the process of solving the first problem, we also develop a decentralized power iteration algorithm that allows each agent to estimate the algebraic connectivity of the graph and its associated eigenvector.

7.2. Directions of Future Research

7.2.1. Decentralized Optimal Weights Tuning for the Dynamic Consensus Estimator

In this thesis we have shown the convergence speed of the consensus estimators depends significantly on the edge weights: the time constant of an unweighted consensus estimator can be several times that of the optimally-weighted one. It is more desirable to have a decentralized optimal weights tuning scheme as it is more robust to network topology changes, which is inevitable in mobile sensor networks.

This decentralized optimal weighting problem looks hard initially because each node can not accurately evaluate the connectivity function, or the change of the connectivity function, with only local weights information. However, with the new technique of estimating the eigenvector v_2 also developed in this thesis, the solution may exist. This is considering the fact that $\lambda_2 = \frac{\sum_{ij} w_{ij}(v_2^i - v_2^j)^2}{\sum_{ij}((v_2^i - v_2^j)^2)}$. Under this eigenvalue structure and with good estimates of the eigenvector v_2, each node may be able to evaluate the change of the connectivity function due to weight changes.

This direction of research falls under the category of decentralized optimization over sensor networks. This is a burgeoning field, as there is a shifting focus in having more in-network data processing in the use of sensor networks. Compared to the traditional use of each sensor node for merely data relaying, this new approach offers higher communication efficiency [**62**], measured by transmitted bits over the transmitted distance. The distributed optimization algorithm in [**62**] can only deal with quadratic functions, and the development of an optimization algorithm for the eigenvalue type of functions will be a nice addition in this field.

7.2.2. Finite-time Consensus Protocols

One promising extension of the research on existing consensus protocols is to investigate finite-time consensus protocols. This is partially motivated by the constraints on the current hardware: Each sensor node typically has relatively large storage space, lower power consumption in computing but higher power consumption in communicating. This first paper on finite-time static consensus [**75**] deals with fixed topologies. Its method, which relies on using all the history of estimator states, is hard to be extended to the time-varying graph case. Some alternative approaches are being investigated on the time-varying graph case [**81**]. Another possible avenue is to investigate finite-time consensus for the time-varying inputs. In the case of fixed topologies, the presence of time-varying inputs will dilute the value of past estimator states. But since these inputs are available information, the history of the inputs plus the history of the estimator states may still help the consensus estimator converge faster. As far as the author knows, there is no existing research in this area yet.

7.2.3. Extending the Capabilities of Consensus Estimators

Existing consensus estimators can estimate the maximum, minimum and the average of the node values (or any nonlinear function of the node value). We can not estimate $\prod_i x_i$ now (assuming we do not know the total number of agents n), nor do we know it is impossible. Future research is needed to answer the question of what other global information can be estimated. In practice, the interested global information of interest in mobile sensor networks is data-aggregated rather than sensor-aggregated [**3**]. This means the global information has to be a symmetric function with respect to the sensor node numbering. Functional basis of the symmetrical function space is an existing topic in mathematics [**60**], and it would be useful to see if those established results can be applied to the sensor network consensus estimation case.

7.2.4. Analysis of Dynamic Consensus Estimators under Realistic Networking Conditions

Some analysis for the two dynamic consensus estimators under more realistic networking conditions are needed, for example the effect of asynchronicity, packet delay and especially packet loss due to packet collision and packet corruption. In the eight-robot physical experiment that we conducted, possibly due to packet collision the packet loss rate on each e-puck varies significantly from each other, and can be as severe as 20%. The packet loss rate will further increase when the density of the sensor network increases. The impact of these realistic communication conditions on the convergence and steady-state error of the dynamic consensus estimators needs to be characterized.

References

[1] The btnodes. In *http://www.btnode.ethz.ch/Main/Overview*.

[2] The intel mote. In *http://www.intel.com/research/exploratory/motes.htm*.

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

[4] S. Aranda, S. Martínez, and F. Bullo. On optimal sensor placement and motion coordination for target tracking. In *IEEE International Conference on Robotics and Automation*, 2005.

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

[6] D. Bertsekas and J. Tsitsiklis. *Parallel and Distributed Computation*. Old Tappan, NJ (USA); Prentice Hall Inc., 1989.

[7] D. P. Bertsekas and J. N. Tsitsiklis. Comment on 'coordination of groups of mobile autonomous agents using nearest neighbor rules'. *IEEE Transactions on Automatic Control*, 52(5):968–969, May 2007.

[8] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis. Convergence in multiagent coordination, consensus and flocking. In *IEEE International Conference on Decision and Control*, 2005.

[9] J. Bochnak, M. Coste, and M.-F. Roy. *Real Algebraic Geometry*, volume 36 of *Ergebnisse der Mathematik und ihrer Grenzgebiete, 3. Folge*. Springer-Verlag, Berlin, 1998.

[10] M. Brookes. Matrix manual. In *http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/calculus.html*.

[11] J.-Y. Chen, G. Pandurangan, and D. Xu. Robust computation of aggregates in wireless sensor networks: distributed randomized algorithms and analysis. In *IPSN '05: Proceedings of the 4th*

international symposium on Information processing in sensor networks, page 46, Piscataway, NJ, USA, 2005. IEEE Press.

[12] T. H. Chung, J. W. Burdick, and R. M. Murray. Decentralized motion control of mobile sensing agents in a network. In *IEEE International Conference on Decision and Control*, 2005.

[13] C. Cianci, X. Raemy, J. Pugh, and A. Martinoli. Communication in a swarm of miniature robots: The e-puck as an educational tool for swarm robotics. In *Swarm Robotics: Second Sab 2006 International Workshop*. Springer Verlag, 2007.

[14] J. Cortés. Characterizing robust coordination algorithms via proximity graphs and set-valued maps. In *American Control Conference*, pages 8–13, 2006.

[15] J. Cortés and F. Bullo. Coordination and geometric optimization via distributed dynamical systems. *SIAM Journal on Control and Optimization*, 44(5):1543–1574, 2005.

[16] J. Cortés, S. Martínez, and F. Bullo. Analysis and design tools for distributed motion coordination. In *American Control Conference*, pages 1680–1685, 2005.

[17] J. Cortés, S. Martínez, and F. Bullo. Robust rendezvous for mobile autonomous agents via proximity graphs in d dimensions. *IEEE Transactions on Automatic Control*, 2005.

[18] J. Cortés, S. Martínez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.

[19] I. D. Couzin, J. Krause, R. James, G. D. Ruxton, and N. R. Franks. Collective memory and spatial sorting in animal groups. *Theoretical Biology*, 218:1–11, 2002.

[20] K. Dantu, M. H. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme. Robomote: Enabling mobility in sensor networks. In *IEEE/ACM Fourth International Conference on Information Processing in Sensor Networks (IPSN-SPOTS)*, pages 404–409, Apr 2005.

[21] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (slam)problem. *IEEE Journal of Robotics and Automation*, 17(3):229–241, 2001.

[22] M. Dorigo, E. Tuci, F. Mondada, S. Nolfi, J.-L. Deneubourg, D. Floreano, and L. M. Gambardella. The SWARM-BOTS project. *Kunstliche Intelligenz*, 2005, in press.

[23] S. Fallat and S. J. Kirkland. Extremizing algebraic connectivity subject to graph theoretic constraints. *Electronic Journal of Linear Algebra*, 3:48–74, 1998.

[24] L. Fang and P. Antsaklis. On communication requirements for multi-agent consensus seeking. In *Workshop on Networked Embedded Sensing and Control*, University of Notre Dame, 2005.

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

[26] V. Fedorov. *Theory of optimal experiments*. Academic Press, New York, 1972.

[27] M. Fiedler. Algebraic connectivity of graphs. *Czech. Math. J.*, 23, 1973.

[28] D. Fox, W. Burgard, and S. Thrun. Active Markov localization for mobile robots. *Robotics and Autonomous Systems*, 25:195–207, 1998.

[29] R. A. Freeman. Time-varying feedback for the global stabilization of nonlinear systems with measurement disturbances. In *Proceedings of the 1997 European Control Conference*, Brussels, Belgium, July 1997.

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

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

[32] C. D. Godsil and G. F. Royle. *Algebraic Graph Theory*. Springer-Verlag, 2001.

[33] R. M. Gray. *Toeplitz and Circulant Matrices*. http://ee.stanford.edu/g̃ray/toeplitz.pdf, 1971, revised 2002.

[34] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. *SIGOPS Oper. Syst. Rev.*, 34(5):93–104, 2000.

[35] A. Howard. Multi-robot simultaneous localization and mapping using particle filters. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[36] A. Howard, M. Mataric, and G. Sukhatme. Mobile sensor network deployment using potential fields: A distributed. Fukuoka, Japan, June 2002.

[37] A. Howard, M. J. Matarić, and G. S. Sukhatme. An incremental self-deployment algorithm for mobile sensor networks. *Autonomous Robots Special Issue on Intelligent Embedded Systems*, 13(2):113–126, 2002.

[38] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, Jun 2003.

[39] D. Lee and M. W. Spong. Stable flocking of multiple inertial agents on balanced graphs. In *American Control Conference*, pages 2136–2141, 2006.

[40] J. Lin, A. S. Morse, and B. D. O. Anderson. The multi-agent rendezvous problem. In *IEEE International Conference on Decision and Control*, 2003.

[41] D. Lucarelli and I.-J. Wang. Decentralized synchronization protocols with nearest neighbor communication. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 62–68, New York, NY, USA, 2004. ACM Press.

[42] J. A. Marshall, M. E. Broucke, and B. A. Francis. Formations of vehicles in cyclic pursuit. *IEEE Transactions on Automatic Control*, 49(11):1963–1974, Nov. 2004.

[43] J. M.Esposito and T. W.Dunbar. Maintaining wireless connectivity constraints for swarms in the presence of obstacles. In *IEEE International Conference on Robotics and Automation*, Orlando, FL, USA, 2006.

[44] C. C. Moallemi and B. V. Roy. Consensus propagation. *IEEE Transactions on Information Theory*, 52(11):4753–4766, 2006.

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

[46] L. Moreau. Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, 50(2):169–182, Feb. 2005.

[47] A. E. Motter, C. Zhou, and J. Kurths. Weighted networks are more synchronizable: how and why. In *AIP Conference Proceedings*, volume 76, 2005.

[48] M. E. J. Newman and D. J. Watts. Renormalization group analysis of the small-world network model. *Physics Letters A*, 263:341, 1999.

[49] T. Nishikawa and A. Motter. Maximum performance at minimum cost in network synchronization. *Physica D: Nonlinear Phenomena*, 224(1-2):77–89, 2006.

[50] G. Notarstefano and F. Bullo. Distributed consensus on enclosing shapes and minimum time rendezvous. In *IEEE International Conference on Decision and Control*, pages 4295–4300, San Diego, CA, Dec. 2006.

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

[52] P. Ogren, E. Fiorelli, and N. E. Leonard. Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment. *IEEE Transactions on Automatic Control*, 49(8):1292–1302, 2004.

[53] R. Olfati-Saber. Distributed Kalman filter with embedded consensus filters. In *IEEE International Conference on Decision and Control*, 2005.

[54] R. Olfati-Saber. Ultrafast consensus in small-world networks. In *American Control Conference*, 2005.

[55] R. Olfati-Saber and R. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, Sep 2004.

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

[57] A. Olshevsky and J. N. Tsitsiklis. Convergence speed in distributed consensus and averaging. In *IEEE International Conference on Decision and Control*, San Diego, CA, 2006.

[58] D. Paley, F. Zhang, and N. E. Leonard. Cooperative control for ocean sampling: The glider coordinated control system. *IEEE Transactions on Control Systems Technology*, (4), July 2008.

[59] J. K. Parrish, S. V. Viscido, and D. Grunbaum. Self-organized fish schools: An examination of emergent properties. *Biol. Bull.*, 202:296–305, June 2002.

[60] A. Plichco and E. Tokarev. Bases of symmetric function spaces. *Matematicheskie Zametki*, 42(2):227–234, 1987.

[61] S. Poduri and G. Sukhatme. Constrained coverage for mobile sensor networks. In *IEEE International Conference on Robotics and Automation*, 2004.

[62] M. Rabbat and R. Nowak. Distributed optimization in sensor networks. In *Proceedings of the third international symposium on Information processing in sensor networks*, pages 20–27. ACM Press New York, NY, USA, 2004.

[63] K. Ramachandra. *Kalman Filtering Techniques for Radar Tracking*. Marcel Dekker, New York, NY, 2000.

[64] W. Ren. Consensus based formation control strategies for multi-vehicle systems. In *American Control Conference*, pages 4237–4242, 2006.

[65] W. Ren and R. W. Beard. Consensus seeking in multi-agent systems using dynamically changing interaction topologies. *IEEE Transactions on Automatic Control*, 50(5):655–661, May 2005.

[66] W. Ren, R. W. Beard, and E. M. Atkins. Information consensus in multivehicle cooperative control: Collective group behavior through local interaction. *IEEE Control Systems Magazine*, 27(2):71–82, April 2007.

[67] C. W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4):25–34, 1987.

[68] J. A. Rothermich, M. I. Ecemis, and P. Gaudiano. Distributed localization and mapping with a robotic swarm. In *SAB 2004 International Workshop*, pages 58–69, 2004.

[69] S. Roumeliotis and G. Bekey. Distributed multirobot localization. *IEEE Transactions on Robotics and Automation*, 18(5):781–795, 2002.

[70] B. Schechter. Birds of a feather. *New Scientist*, pages 30–33, Janurary 23 1999.

[71] M. Schwager, J. McLurkin, and D. Rus. Distributed coverage control with sensory feedback for networked robots. In *Proceedings of Robotics: Science and Systems*, Philadelphia, USA, August 2006.

[72] D. P. Spanos, R. Olfati-Saber, and R. M. Murray. Dynamic consensus for mobile networks. In *Proceedings of the 2005 IFAC World Congress*, Prague, 2005.

[73] D. P. Spanos, R. Olfati-Saber, and R. M. Murray. Dynamic consensus on mobile networks. In *IFAC*, 2005.

[74] R. F. Stengel. *Optimal control and estimation*. Dover Publications, New York, 1994.

[75] S. Sundaram and C. Hadjicostis. Finite-time distributed consensus in graphs with time-invariant topologies. In *American Control Conference*, pages 711–716, 2007.

[76] S. Sundaram and C. N. Hadjicostis. Finite-time distributed consensus in graphs with time-invariant topologies. In *American Control Conference*, 2007. submitted.

[77] S. Thrun and Y. Liu. Multi-robot SLAM with sparse extended information filers. In *Proceedings of the 11th International Symposium of Robotics Research (ISRR'03)*, Sienna, Italy, 2003. Springer.

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

[79] J. N. Tsitsiklis and D. P. Bertsekas. Distributed asynchronous optimal routing in data networks. *IEEE Transactions on Automatic Control*, 31(4):325–332, 1986.

[80] F. Viger and M. Latapy. Fast generation of random connected graphs with prescribed degrees. *Available at http://www.citebase.org/cgi-bin/citations?id=oai:arXiv.org:cs/0502085*, 2005.

[81] L. Wang and F. Xiao. Finite-time consensus problems for networks of dynamic agents. *Arxiv preprint math/0701724*, 2007.

[82] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–42, 1998.

[83] S. Williams, G. Dissanayake, and H. Durrant-Whyte. Towards multi-vehicle simultaneous localisation and mapping. In *IEEE International Conference on Robotics and Automation*, 2002.

[84] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Systems and Control Letters*, 53:65–78, Sept. 2004.

[85] F. Xue and P. R. Kumar. The number of neighbors needed for connectivity of wireless networks. *Wireless Networks*, 10(2):169–181, 2004.

[86] P. Yang, R. A. Freeman, and K. M. Lynch. Optimal information propagation in sensor networks. In *IEEE International Conference on Robotics and Automation*, 2006.

[87] M. M. Zavlanos and G. J. Pappas. Controlling connectivity of dynamic graphs. In *IEEE Conference on Decision and Control*, 2005. submitted.

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

[89] F. Zhang and N. E. Leonard. Generating contour plots using multiple sensor platforms. In *Proc. IEEE Swarm Intelligence Symposium*, 2005.

APPENDIX A

Stability of the Centralized Power Iteration

In the appendix we analyze the stability properties of system (4.3). We show the boundedness of all trajectories, characterize the equilibrium sets and their local stability, and finally give a proof of Theorem 4.

Proposition 1. *Given any initial condition $x(t_0)$ and any positive gains $k_1, k_2, k_3 > 0$, the system trajectory remains bounded over time:*

$$\|x(t)\| \leq \max\{\|x(t_0)\|, \sqrt{n}\}. \tag{A.1}$$

Proof. Defining $V_1 = x^T x = y^T y$, we have

$$\begin{aligned} \dot{V}_1 &= 2y^T \dot{y} \\ &= 2y^T[-k_1 \mathrm{diag}(1, 0, \ldots, 0) - k_2 L^* - k_3\Big(\frac{y^T y}{n} - 1\Big) I]y. \end{aligned} \tag{A.2}$$

If $\|x(t_0)\| > \sqrt{n}$, then $k_3(\frac{y^T y}{n} - 1) > 0$ and $\dot{V}_1 < 0$ until $\|x(t)\| \leq \sqrt{n}$. If $\|x(t_0)\| \leq \sqrt{n}$, then $\|x(t)\| \leq \sqrt{n}$ for all $t > 0$. □

The following two propositions completely characterize the equilibrium sets of system (4.3) and their local stability properties.

Proposition 2. *System (4.3) has an equilibrium point $x = 0$, and it is locally unstable when $k_3 > k_2\lambda_2$.*

Proof. It is easy to verify that $x = 0$ (or $y = 0$) is an equilibrium state of system (4.3). Linearizing the equivalent system equation (4.6) around the point $y = \tilde{y}$ we get

$$\dot{y} = [-k_2\widetilde{L}^* - k_3\Big(\frac{\tilde{y}^T\tilde{y}}{n} - 1 + 2\frac{\tilde{y}\tilde{y}^T}{n}\Big)I]y. \tag{A.3}$$

Plugging in $\tilde{y} = 0$, equation (A.3) is simplied to $\dot{y} = [k_3 - k_2\tilde{L}^*]y$. The gain condition $k_3 > k_2\lambda_2$ makes the equilibrium point $y = 0$ locally unstable, at least in one direction. □

Now we proceed to investigate the non-zero equilibrium points of system (4.3).

Proposition 3. *When the gain conditions (4.7), (4.8) are satisfied, system (4.3) has n (when $k_3 > k_1$) or $n - 1$ (when $k_3 \leq k_1$) pairs of distinct non-zero equilibrium points $\{y_i \mid 1 \leq i \leq n\}$ where*

$$y_1 = \Big(\pm\sqrt{n(\frac{k_3 - k_1}{k_3})}, 0, \ldots, 0\Big)^T, \text{ if } k_3 > k_1; \tag{A.4}$$

and $\{y_i \mid 2 \leq i \leq n\}$ is

$$y_i^j = \begin{cases} 0 & \text{if } 2 \leq j \leq n, j \neq i, \\ \pm\sqrt{n(\frac{k_3 - k_2\lambda_i}{k_3})} & \text{if } j = i. \end{cases} \tag{A.5}$$

Additionally, among all the n or $n - 1$ pairs of equilibria, only y_2 is locally stable.

Proof. The insight here is that any nonzero equilibrium point y of system (4.6) has to be an eigenvector of the matrix $\widetilde{L}^*$ with an associated eigenvalue $\frac{k_3}{k_2}(\frac{y^Ty}{n} - 1)$. Furthermore, we know the n different unit eigenvectors for the diagonal matrix $\widetilde{L}^* \in \mathbb{R}^{n\times n}$. Therefore, we can solve for all the eigenvectors of the system (4.6) that are also equilibria of the system. There are n such eigenvectors in total, described in (A.4) and (A.5). Additionally,

we use the linearized models (A.3) to check the local stability of every y_i. For y_1, its eigenvalue spectrum $\{\mu_1^j \,|\, j = 1, \ldots, n\}$ is

$$\text{(A.6)} \qquad \begin{cases} \mu_1^1 = -2(k_3 - k_1) & \text{if } j = 1, \\ \mu_1^j = k_1 - k_2\lambda_j & \text{if } j = 2, \ldots, n. \end{cases}$$

Since at least $\mu_1^2 > 0$, y_1 is locally unstable. Similarly for the equilibrium point y_i, $i = 2, \ldots, n$, its eigenvalue spectrum $\{\mu_i^j \,|\, j = 2, \ldots, n\}$ is

$$\text{(A.7)} \qquad \begin{cases} \mu_i^1 = k_2\lambda_i - k_1 & \text{if } j = 1, \\ \mu_i^j = k_2(\lambda_i - \lambda_j) & \text{if } j = 2, \ldots, n, j \neq i, \\ \mu_i^i = -2(-k_2\lambda_i + k_3) & \text{if } j = i. \end{cases}$$

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

Finally we give a proof for the near-global convergence result stated in Theorem 4.

It is useful to write out equation (4.6) in its scalar form:

$$\text{(A.8)} \qquad \dot{y}^1 = (-k_1 - k_3\Big(\frac{y^T y}{n} - 1\Big))y^1$$

$$\text{(A.9)} \qquad \dot{y}^2 = (-k_2\lambda_2 - k_3\Big(\frac{y^T y}{n} - 1\Big))y^2$$

$$\vdots$$

$$\text{(A.10)} \qquad \dot{y}^n = (-k_2\lambda_n - k_3\Big(\frac{y^T y}{n} - 1\Big))y^n.$$

We first notice that the value of each component y^i will not change its sign over time and if $y^i(t_0) = 0$, $y^i(t)$ remains zero. Next we present the complete proof of the main theorem.

Proof. (Sufficiency) Let us first consider y^1. If $y^1(t_0) = 0$, then $y^1(t) = 0$ for all t. If $y^1(t_0) \neq 0$, combining (A.8) and (A.9) we get

$$\frac{d}{dt}(\ln\frac{y^2}{y^1}) = \frac{d}{dt}(\ln y^2) - \frac{d}{dt}(\ln y^1) = k_1 - k_2\lambda_2 > 0 \tag{A.11}$$

which implies $y^2/y^1 \to \infty$. We know y^2 is bounded from Theorem 1, therefore $y^1 \to 0$. The cases are similar for $y^i, i > 2$. If $y^i(t_0) = 0$, then $y^i(t) = 0$ for all t. If $y^i(t_0) \neq 0$, then $y^2/y^i \to \infty$ and $y^i \to 0$.

Therefore over time equation (A.9) is reduced to $\dot{y}^2 = (-k_2\lambda_2 - k_3(\frac{(y^2)^2}{n} - 1))y^2$. When (4.8) holds, this scalar dynamical system can be rewritten as

$$\dot{y}^2 = \frac{k_3}{n}(\sqrt{n\Big(\frac{k_3 - k_2\lambda_2}{k_3}\Big)} + y^2)(\sqrt{n\Big(\frac{k_3 - k_2\lambda_2}{k_3}\Big)} - y^2)y^2. \tag{A.12}$$

We see that $y^2 \to \pm\sqrt{n\Big(\frac{k_3 - k_2\lambda_2}{k_3}\Big)}$ depending on the initial condition $y^2(t_0)$ and the equilibrium point $y^2 = 0$ is unstable.

(Necessity) When $y^2 \to \pm\sqrt{n\Big(\frac{k_3 - k_2\lambda_2}{k_3}\Big)}$ obviously condition (4.8) holds. Now we suppose the condition $k_1 \leq k_2\lambda_2$ holds. If $k_1 < k_2\lambda_2$, using the same argument method in (A.11), $y^1/y^2 \to \infty$ and therefore $y^2 \to 0$, which is a contradiction. If $k_1 = k_2\lambda_2$, then $\frac{d}{dt}(\ln\frac{y^1}{y^2}) = 0$ and y^1/y^2 is a constant c. For initial conditions $y^1(t_0) \neq 0$, $c = y^1(t_0)/y^2(t_0) \neq 0$, therefore y cannot converge to y_2 where $y_2^1/y_2^2 = 0$, which is also a contradiction. Therefore, the gain condition (4.7) must hold. □

Remark 2. *In case of repeated eigenvalues $\lambda_2 = \cdots = \lambda_k < \lambda_{k+1}$, Theorem 4 still holds. In this case all trajectories with $y^2(t_0) \neq 0$ converge to an equilibrium point on the k-dimensional manifold $\{y | \, \|y\| = \sqrt{n\Big(\frac{k_3 - k_2\lambda_2}{k_3}\Big)}, y^1 = 0, y^i = 0, \forall i > k\}$.*

APPENDIX B

Proof of Theorem 7

Let M denote the subspace of $R^{m\times(m+1)}$ comprised of all matrices of the form $[x \;\; X]$, where $x \in \mathbb{R}^m$ and X is a symmetric $m \times m$ matrix. Let $\mathcal{V} : M \to \mathbb{R}^\ell$ denote the invertible linear map given by

$$\mathcal{V}\big([x \quad X]\big) \triangleq \begin{bmatrix} x \\ \\ \mathrm{vech}(X) \end{bmatrix}. \tag{B.1}$$

First suppose that $f^\star$ has the special form

$$f^\star = \mathcal{V}\big([0 \quad \Delta]\big) \tag{B.2}$$

for some diagonal matrix $\Delta \geqslant 0$. We now show that any diagonal $\Gamma > 0$ works in this case. Given a swarm configuration $p \in \mathfrak{P}$, we let $n = n(p)$ be such that $p \in \mathbb{R}^{mn}$, and we define $P \triangleq [p_1 \;\; \ldots \;\; p_n] \in \mathbb{R}^{m\times n}$ so that $p = \mathrm{vec}(P)$ (where $\mathrm{vec}(\cdot)$ is the invertible linear map which stacks the columns of a matrix to produce a vector). For convenience we introduce $q = \mathrm{vec}(Q)$, where $Q = P^T = [q_1 \;\; \ldots \;\; q_m] \in \mathbb{R}^{n(p)\times m}$. In these coordinates, the function $f(p)$ becomes

$$f(p) = \frac{1}{n(p)} \begin{bmatrix} Q^T\mathbf{1} \\ \\ \mathrm{vech}(Q^TQ) \end{bmatrix}. \tag{B.3}$$

Computing $\mathcal{J}_q f$, the q-Jacobian of f, we obtain

$$\mathcal{J}_q f = \frac{1}{n(p)} \begin{bmatrix} \mathbf{1}^T & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & \mathbf{1}^T \\ 2q_1^T & 0 & 0 & \dots & 0 & 0 \\ 0 & 2q_2^T & 0 & \dots & 0 & 0 \\ 0 & 0 & 2q_3^T & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 2q_{m-1}^T & 0 \\ 0 & 0 & 0 & \dots & 0 & 2q_m^T \\ q_2^T & q_1^T & 0 & \dots & 0 & 0 \\ 0 & q_3^T & q_2^T & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & q_{m-1}^T & q_{m-2}^T & 0 \\ 0 & 0 & \dots & 0 & q_m^T & q_{m-1}^T \\ q_3^T & 0 & q_1^T & 0 & \dots & 0 \\ 0 & q_4^T & 0 & q_2^T & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & q_m^T & 0 & q_{m-2}^T \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ q_m^T & 0 & 0 & \dots & 0 & q_1^T \end{bmatrix}.$$

If we define

$$z = \Gamma\big[f(p) - f^\star\big]\,, \tag{B.4}$$

then we can write the q-gradient of J in (5.1) as

$$\nabla_q J(p) = 2\big[\mathcal{J}_q f\big]^T z\,. \tag{B.5}$$

We write $z = [z_1 \ \dots \ z_\ell]^T$, $\Gamma = \mathrm{diag}(\gamma_1, \dots, \gamma_\ell)$ with each $\gamma_i > 0$, and $\Delta = \mathrm{diag}(\delta_1, \dots, \delta_m)$ with each $\delta_i \geqslant 0$. Now suppose p is such that $\nabla J(p) = 0$ but $z \neq 0$; then also $\nabla_q J(p) = 0$, and in particular

$$\begin{aligned} 0 &= \big[q_1^T\ 0\ \dots\ 0\big]\big[\mathcal{J}_q f\big]^T z \qquad\qquad (B.6)\\ &= \frac{1}{n^2(p)}\big[\gamma_1(\mathbf{1}^T q_1)^2 + 2\gamma_{m+1}(q_1^T q_1)\big(q_1^T q_1 - n(p)\delta_1\big) \\ &\qquad + \gamma_{2m+1}(q_2^T q_1)^2 + \ldots + \gamma_\ell(q_m^T q_1)^2\big]\,. \qquad\qquad (B.7) \end{aligned}$$

This implies $q_1^T q_1 \leqslant n(p)\delta_1$, and in a similar manner we can show that $q_i^T q_i \leqslant n(p)\delta_i$ for $1 \leqslant i \leqslant m$. Furthermore, because $z \neq 0$, there exists some j such that $q_j^T q_j < n(p)\delta_j$, and for simplicity we assume $j = 1$ (the other cases are similar). Let $v \in \mathbb{R}^{mn(p)}$ be a constant vector, and define

$$F = v^T \nabla_q J(p) = 2w^T z\,, \quad \text{where} \qquad w = \big[\mathcal{J}_q f\big] v\,. \tag{B.8}$$

If we let $\mathcal{H}_q J(p)$ denote the q-Hessian of J at p, then

$$\big[\nabla_q F\big]^T v = v^T\big[\mathcal{H}_q J(p)\big] v \;=\; 2w^T \Gamma w + 2z^T\big[\mathcal{J}_q w\big] v\,. \tag{B.9}$$

Now $\nabla_q J(p) = 0$ and $z \neq 0$ imply $\operatorname{rank}[\mathbf{1} \;\; Q] < m+1$; thus because $n(p) \geqslant m+1$ there exists a nonzero $u \in \mathbb{R}^{n(p)}$ such that $u^T[\mathbf{1} \;\; Q] = 0$. Choosing $v = [u^T \;\; 0 \;\; \dots \;\; 0]^T$, we see that $w = 0$ and thus

$$\begin{aligned} v^T\big[\mathcal{H}_q J(p)\big]v = 2z^T\big[\mathcal{J}_q w\big]v \;&=\; 2z^T\frac{\partial w}{\partial q_1}\,u \\ &= \frac{4}{n(p)}\,z_{m+1}\cdot(u^Tu) \\ &= \frac{4}{n^2(p)}\,\gamma_{m+1}(u^Tu)\big(q_1^Tq_1 - n(p)\delta_1\big) \\ &< 0\,. \end{aligned} \tag{B.10}$$

Therefore the Hessian of J has at least one strictly negative eigenvalue at p, and we conclude that p cannot be a local minimum of J.

To complete the proof, we show the existence of a computable coordinate change *which is independent of* $n(p)$ and is such that any $f^\star \in f(\mathcal{D})$ takes the special form (B.2) in the new coordinates. For each n, define the map $K_n : \mathbb{R}^{m\times n} \to M$ as

$$K_n(X) \triangleq \frac{1}{n}\begin{bmatrix} X\mathbf{1}_n & XX^T \end{bmatrix} \tag{B.11}$$

for $X \in \mathbb{R}^{m\times n}$. Then because

$$\sum_{i=1}^{n}\begin{bmatrix} p_i & p_ip_i^T \end{bmatrix} = \begin{bmatrix} P\mathbf{1}_n & PP^T \end{bmatrix}, \tag{B.12}$$

we can write the moment vector $f(p)$ in (5.4) as

$$f(p) = \big(\mathcal{V}\circ K_{n(p)}\circ \operatorname{vec}^{-1}\big)(p)\,. \tag{B.13}$$

Given $f^\star \in f(\mathcal{D})$, let $\nu \geqslant m+1$ and $p^\star \in \mathbb{R}^{m\nu}$ be such that $f(p^\star) = f^\star$, and define $P^\star = \mathrm{vec}^{-1}(p^\star) \in \mathbb{R}^{m\times\nu}$. We fix $S \in \mathrm{Orth}(\mathbf{1}_\nu)$ and write the SVD of the matrix $P^\star S$ as $P^\star S = U\Sigma V^T$, where $U \in \mathbb{R}^{m\times m}$ is an orthogonal matrix. Using U, for each n we construct the invertible affine transformation $J_n : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ as follows:

$$J_n(X) \triangleq U^T\big[X - \frac{1}{\nu}\,P^\star \mathbf{1}_\nu \mathbf{1}_n^T\big]\,, \tag{B.14}$$

where $X \in \mathbb{R}^{m\times n}$. This mapping has the property that

$$(K_\nu \circ J_\nu)(P^\star) = \frac{1}{\nu}\begin{bmatrix}0 & \Sigma\Sigma^T\end{bmatrix}. \tag{B.15}$$

Next we define $T : M \to M$ as

$$T\big(\begin{bmatrix}x & X\end{bmatrix}\big) \triangleq \begin{bmatrix}T_1(x) & T_2(x,X)\end{bmatrix} \tag{B.16}$$

where $x \in \mathbb{R}^m$, $X \in \mathbb{R}^{m\times m}$ is symmetric, and

$$T_1(x) \triangleq U^T\big[x - \frac{1}{\nu}P^\star \mathbf{1}_\nu\big]\,, \tag{B.17}$$

$$\begin{aligned} T_2(x,X) \triangleq U^T\big[X &- \frac{1}{\nu}\,P^\star \mathbf{1}_\nu x^T - \frac{1}{\nu}\,x\mathbf{1}_\nu^T(P^\star)^T \\ &+ \frac{1}{\nu^2}P^\star \mathbf{1}_\nu \mathbf{1}_\nu^T(P^\star)^T\big]U\,. \end{aligned} \tag{B.18}$$

It is clear that T is an invertible affine map, and it is straightforward to verify that $K_n \circ J_n \equiv T \circ K_n$ for any n. We define $G : \mathfrak{P} \to \mathfrak{P}$ and $W : \mathbb{R}^\ell \to \mathbb{R}^\ell$ as

$$G(p) \triangleq \big(\mathrm{vec} \circ J_{n(p)} \circ \mathrm{vec}^{-1}\big)(p) \tag{B.19}$$

$$W(x) \triangleq \big(\mathcal{V} \circ T \circ \mathcal{V}^{-1}\big)(x) \tag{B.20}$$

for $p \in \mathfrak{P}$ and $x \in \mathbb{R}^\ell$. These maps G and W are such that $f \circ G \equiv W \circ f$, and in particular we have

$$\big(f \circ G\big)(p^\star) = W(f^\star) = \frac{1}{\nu}\,\mathcal{V}\big(\begin{bmatrix}0 & \Sigma\Sigma^T\end{bmatrix}\big)\,. \tag{B.21}$$

To summarize, we have the following commutative diagram:

$$\begin{array}{ccccccc} \mathfrak{P} & \xrightarrow{\ f\ } & \mathbb{R}^\ell & \xrightarrow{\ \mathcal{V}^{-1}\ } & M & \xleftarrow{\ K_n\ } & \mathbb{R}^{m\times n} \\ {\scriptstyle G}\downarrow & & \downarrow{\scriptstyle W} & & \downarrow{\scriptstyle T} & & \downarrow{\scriptstyle J_n} \\ \mathfrak{P} & \xrightarrow{\ f\ } & \mathbb{R}^\ell & \xleftarrow{\ \mathcal{V}\ } & M & \xleftarrow{\ K_n\ } & \mathbb{R}^{m\times n} \end{array} \tag{B.22}$$

Because W is an invertible affine map, there exist an invertible matrix $A \in \mathbb{R}^{\ell\times\ell}$ and $b \in \mathbb{R}^\ell$ be such that $W(x) = Ax + b$ for all $x \in \mathbb{R}^\ell$. Thus for $\Gamma > 0$,

$$\begin{aligned}\big[f(G(p)) - W(f^\star)\big]^T (A^T)^{-1}\Gamma A^{-1}\big[f(G(p)) - W(f^\star)\big] \\ = \big[f(p) - f^\star\big]^T \Gamma \big[f(p) - f^\star\big]\end{aligned} \tag{B.23}$$

for all $p \in \mathfrak{P}$. Now because the restriction of G to each set $\mathbb{R}^{mn}$ is invertible, we have $\Gamma \in \mathcal{G}(f^\star, \mathcal{D})$ if and only if $\Gamma_0 \in \mathcal{G}(W(f^\star), \mathcal{D})$, where

$$\Gamma = A^T \Gamma_0 A\,. \tag{B.24}$$

Thus without loss of generality, we may assume that $f^\star$ has the structure of $W(f^\star)$, namely, that $f^\star$ has the special form (B.2).

APPENDIX C

Proof of Theorem 8

We will need the following technical result:

Lemma 10. *Let $\mathcal{X}$ be a topological space, let $\mathcal{Y}$ be a set, let $A \subset \mathcal{X}$, let $B \subset A$ be connected, and let $h : \mathcal{X} \to \mathcal{Y}$. Suppose every $x \in B$ has an open neighborhood $N_x \subset \mathcal{X}$ such that h is constant on $\mathrm{cl}(N_x) \cap A$. Then h is constant on $\mathrm{cl}(N) \cap A$, where $N \triangleq \bigcup\{N_x : x \in B\}$.*

Proof. Suppose there exist $x_1, x_2 \in \mathrm{cl}(N) \cap A$ such that $h(x_1) \neq h(x_2)$, and define the following subsets of $\mathcal{X}$:

$$N_1 \triangleq \bigcup\big\{N_x \,:\, h(\cdot) = h(x_1) \text{ on } \mathrm{cl}(N_x) \cap A\big\}\,, \tag{C.1}$$

$$N_2 \triangleq \bigcup\big\{N_x \,:\, h(\cdot) \neq h(x_1) \text{ on } \mathrm{cl}(N_x) \cap A\big\}\,, \tag{C.2}$$

$$\mathcal{O}_1 \triangleq N_1 \cap B\,, \qquad \mathcal{O}_2 \triangleq N_2 \cap B\,. \tag{C.3}$$

Then $\mathcal{O}_1$ and $\mathcal{O}_2$ are nonempty, disjoint, open relative to B, and such that $B = \mathcal{O}_1 \cup \mathcal{O}_2$. However, this contradicts the fact that B is connected. □

Consider the consensus estimators (5.13)–(5.14). Defining variables $z_i, e_i \in \mathbb{R}^\ell$ as

$$z_i = \frac{d}{dt}\,\phi(p_i) = \big[\mathcal{J}\phi(p_i)\big]\dot{p}_i \tag{C.4}$$

$$e_i = f(p) - y_i\,, \tag{C.5}$$

we introduce the following $\ell \times n$ matrices:

$$Y = \left[\begin{array}{ccc} y_1 & \dots & y_n \end{array} \right] \tag{C.6}$$

$$H = \left[\begin{array}{ccc} \eta_1 & \dots & \eta_n \end{array} \right] \tag{C.7}$$

$$\Phi(p) = \left[\begin{array}{ccc} \phi(p_1) & \dots & \phi(p_n) \end{array} \right] \tag{C.8}$$

$$E = \left[\begin{array}{ccc} e_1 & \dots & e_n \end{array} \right] = \Phi(p)\,\frac{\mathbf{1}\mathbf{1}^T}{n} - Y \tag{C.9}$$

$$Z = \left[\begin{array}{ccc} z_1 & \dots & z_n \end{array} \right] = \frac{d}{dt}\,\Phi(p)\,. \tag{C.10}$$

Hence we may write the collection of consensus estimators (5.13)–(5.14) in matrix form as

$$\dot{H} = -\gamma H - Y\,L(p) \tag{C.11}$$

$$Y = H + \Phi(p)\,. \tag{C.12}$$

We write the complete state of the closed-loop system as either the triple $(p, \dot{p}, H)$, or with the global coordinate change given by (C.9) and (C.12), the triple $(p, \dot{p}, E)$. We see from (5.15) that the derivative of the storage function

$$V(p, \dot{p}) = \dot{p}^T\dot{p} \,+\, n\,J(p) \tag{C.13}$$

can be written as

$$\dot{V} = \sum_{i=1}^{n} \Big[-\dot{p}_i^T \big[B + B^T\big]\dot{p}_i - z_i^T \big[\Lambda + \Lambda^T\big] z_i + 2 z_i^T \Gamma e_i \Big]\,. \tag{C.14}$$

We observe from (C.9) and (C.12) that $E\mathbf{1} \equiv -H\mathbf{1}$ and thus

$$\dot{E}\mathbf{1} = \gamma H\mathbf{1} = 0 \qquad (\text{using } \gamma = 0), \tag{C.15}$$

which means

$$E(t)\mathbf{1} = -H(t_0)\mathbf{1} = 0 \qquad (\text{using } \eta_i(t_0) = 0) \tag{C.16}$$

for any $t \geqslant t_0$. Because $ES \equiv -YS$, we can write

$$\dot{E}S = YL(p)S - ZS = -ESL^\star(p) - ZS\,. \tag{C.17}$$

Also because $E\mathbf{1} \equiv 0$ we have $EE^T \equiv ESS^TE^T$, so

$$\begin{aligned}
\frac{d}{dt}\,EE^T &= \dot{E}SS^TE^T + ESS^T\dot{E}^T \\
&= -2ESL^\star(p)S^TE^T - ZSS^TE^T - ESS^TZ^T \\
&\leqslant -\varepsilon EE^T + \frac{1}{\varepsilon}ZSS^TZ^T \\
&\leqslant -\varepsilon EE^T + \frac{1}{\varepsilon}ZZ^T\,. \qquad (\text{C.18})
\end{aligned}$$

Defining the storage function $U = \operatorname{Tr}(EE^T)$ we see that

$$\dot{U} \leqslant -\varepsilon U + \frac{1}{\varepsilon}\operatorname{Tr}(ZZ^T) = \sum_{i=1}^{n}\Big[-\varepsilon|e_i|^2 + \frac{1}{\varepsilon}|z_i|^2\Big]\,. \tag{C.19}$$

Furthermore, using the fact that

$$2z_i^T\Gamma e_i \leqslant \frac{1}{2}\,z_i^T\big[\Lambda + \Lambda^T\big]z_i + 2e_i^T\Gamma\big[\Lambda + \Lambda^T\big]^{-1}\Gamma e_i\,, \tag{C.20}$$

we can bound (C.14) from above as

$$\dot{V} \leqslant \sum_{i=1}^{n} \Big[-\dot{p}_i^T \big[B + B^T \big] \dot{p}_i - \frac{1}{2}\, z_i^T \big[\Lambda + \Lambda^T \big] z_i + 2 e_i^T \Gamma \big[\Lambda + \Lambda^T \big]^{-1} \Gamma e_i \Big] . \tag{C.21}$$

To combine the storage functions V and U, we first use (5.19) to choose $\mu > 0$ such that

$$\Gamma < \mu I < \frac{\varepsilon}{2} \big[\Lambda + \Lambda^T \big] . \tag{C.22}$$

Upon taking inverses and then multiplying by Γ from the left and the right, we obtain

$$\frac{2}{\varepsilon}\, \Gamma \big[\Lambda + \Lambda^T \big]^{-1} \Gamma < \Gamma . \tag{C.23}$$

In particular, (C.22) and (C.23) imply the existence of a scalar constant $\nu > 0$ such that

$$\frac{1}{2} \big[\Lambda + \Lambda^T \big] - \frac{\mu}{\varepsilon}\, I \geqslant \nu I \tag{C.24}$$

$$\mu \varepsilon I - 2 \Gamma \big[\Lambda + \Lambda^T \big]^{-1} \Gamma \geqslant \nu I . \tag{C.25}$$

We then define $\Upsilon(p, \dot{p}, E) = V + \mu U$ and use (C.19), (C.21), (C.24), and (C.25) to obtain

$$\dot{\Upsilon} \leqslant \sum_{i=1}^{n} \Big[-\dot{p}_i^T \big[B + B^T \big] \dot{p}_i - \nu |z_i|^2 - \nu |e_i|^2 \Big] . \tag{C.26}$$

In particular, $\Upsilon(t)$ is nonincreasing along trajectories in forward time. Because $J(p)$ is proper, Υ is a proper function of the states p, $\dot{p}$, and E, and we can conclude that all signals are bounded in forward time. By LaSalle's theorem we further conclude that every trajectory converges to its nonempty, compact, connected positive limit set L^+, and that every point in L^+ is an equilibrium point of the form $(p, \dot{p}, E) = (\bar{p}, 0, 0)$ for some $\bar{p} \in \mathbb{R}^{mn}$

such that

$$\left[\mathcal{J}\phi(\bar{p}_i)\right]^T \Gamma \left[f^\star - f(\bar{p})\right] = 0 \tag{C.27}$$

for every i, or equivalently such that $\nabla J(\bar{p}) = 0$. It follows that Υ and thus also J are constant on every positive limit set. In particular, any positive limit set containing one bad equilibrium must contain only bad equilibria.

Next suppose ϕ is subanalytic, suppose $\Gamma \in \mathcal{G}(f^\star, \mathcal{D})$, and suppose a positive limit set L^+ consists of bad equilibria. We can write L^+ as the product $L^+ = L_0^+ \times \{0\} \times \{0\}$, where $L_0^+ \subset \mathcal{D} \cap \mathrm{Crit}(J)$. Because ϕ is subanalytic, so is J, and thus by Theorem **??**, J is locally constant on $\mathrm{Crit}(J)$. Hence every point $p \in L_0^+$ has an open neighborhood N_p such that J is constant on the set $\mathrm{cl}(N_p) \cap \mathrm{Crit}(J)$. Define the open set $N \triangleq \bigcup\{N_p : p \in L_0^+\}$; then the set $\mathcal{O} = N \times \mathbb{R}^{mn} \times \mathbb{R}^{\ell \times n}$ is an open neighborhood of L^+. Also, it follows from Lemma 10 that J is constant on $\mathrm{cl}(N) \cap \mathrm{Crit}(J)$. Let $\mathcal{U}$ be any open set such that $L^+ \cap \mathcal{U} \neq \varnothing$, and fix $(\bar{p}, 0, 0) \in L^+ \cap \mathcal{U}$. By assumption $\bar{p}$ is not a local minimum of J, which means there exists a point $(p_0, 0, 0) \in \mathcal{U}$ such that $J(p_0) < J(\bar{p})$ and therefore also $\Upsilon(p_0, 0, 0) < \Upsilon(\bar{p}, 0, 0)$. Let $L_1^+ \times \{0\} \times \{0\}$ denote the positive limit set of the trajectory starting from the state $(p_0, 0, 0)$. Then J is constant on L_1^+, and because Υ is nonincreasing along trajectories, the value of J on L_1^+ is strictly less than its value on $\mathrm{cl}(N) \cap \mathrm{Crit}(J)$. Thus because $L_1^+ \subset \mathrm{Crit}(J)$ we have $L_1^+ \cap \mathrm{cl}(N) = \varnothing$, and it follows that the trajectory starting from $(p_0, 0, 0)$ eventually leaves the open neighborhood $\mathcal{O}$ of L^+ forever. We conclude that L^+ is strongly unsteady.

APPENDIX D

Proof of Theorem 9

The derivative of the storage function (C.13) is

$$\dot{V} = \sum_{i=1}^{n} \Big[-\dot{p}_i^T \big[B + B^T \big] \dot{p}_i - 2c\zeta(p_i) |\dot{p}_i|^2 - z_i^T \big[\Lambda + \Lambda^T \big] z_i + 2 z_i^T \Gamma e_i \Big] \tag{D.1}$$

with z_i and e_i as in (C.4)–(C.5). We may write the collection of PI estimators (6.21)–(5.27) in matrix form as

$$\dot{Y} = -Y \big[\gamma I + L_P(p) \big] + W L_I(p) + \gamma \Phi(p) \tag{D.2}$$

$$\dot{W} = -Y L_I(p) \tag{D.3}$$

with Y and Φ from (C.6) and (C.8), and with

$$W = \begin{bmatrix} w_1 & \dots & w_n \end{bmatrix} . \tag{D.4}$$

Defining E and Z as in (C.9)–(C.10) we obtain

$$\begin{aligned} \dot{E}\mathbf{1} = Z\mathbf{1} - \dot{Y}\mathbf{1} &= Z\mathbf{1} - \gamma \big[\Phi(p)\mathbf{1} - Y\mathbf{1} \big] \\ &= -\gamma E\mathbf{1} + Z\mathbf{1} \end{aligned} \tag{D.5}$$

$$\dot{W}\mathbf{1} = 0 \, . \tag{D.6}$$

From (5.16) we have $L_P \equiv L_P SS^T$ and $L_I \equiv L_I SS^T$, which means we can multiply both sides of (D.2)–(D.3) from the right by S to obtain

$$\dot{Y}S = -YS\big[\gamma I + L_P^\star(p)\big] + WSL_I^\star(p) + \gamma\Phi(p)S \tag{D.7}$$

$$\dot{W}S = -YSL_I^\star(p)\,. \tag{D.8}$$

With the change of variables

$$H = WS + \gamma\Phi(p)S\big[L_I^\star(p)\big]^{-1} \tag{D.9}$$

$$\Omega = \begin{bmatrix} YS & H \end{bmatrix} \tag{D.10}$$

the equations (D.7)–(D.8) become

$$\dot{\Omega} = \Omega F^T + NG^T\,, \tag{D.11}$$

where

$$F = \begin{bmatrix} -\gamma I - L_P^\star(p) & L_I^\star(p) \\ -L_I^\star(p) & 0 \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ I \end{bmatrix}, \tag{D.12}$$

$$N = \gamma ZS\big[L_I^\star(p)\big]^{-1} + \gamma\Phi(p)S\frac{d}{dt}\big[L_I^\star(p)\big]^{-1}\,. \tag{D.13}$$

We will write N as the sum

$$N = \gamma\sum_{i=0}^{n} N_i \tag{D.14}$$

where

$$N_0 = ZS\big[L_I^\star(p)\big]^{-1}$$

$$N_i = -\Phi(p)S\big[L_I^\star(p)\big]^{-1}\sum_{k=1}^{m}\frac{\partial L_I^\star(p)}{\partial p_i(k)}\big[L_I^\star(p)\big]^{-1}\dot{p}_i(k) \tag{D.15}$$

$$\text{for } 1 \leqslant i \leqslant n \tag{D.16}$$

and $p_i = [p_i(1) \ldots p_i(m)]^T \in \mathbb{R}^m$. We now derive bounds on these matrices N_i. First, using (5.30) we obtain

$$N_0 N_0^T = ZS\big[L_I^\star(p)\big]^{-2}S^TZ^T \leqslant \frac{1}{\varepsilon^2}\,ZSS^TZ^T \tag{D.17}$$

Next, using (5.23) and our assumption that $p(t) \in \mathfrak{C}$ for all $t \geqslant t_0$, we obtain

$$\begin{aligned}|\Phi(p)S|_F^2 &= \Big|\big[\Phi(p) - \phi(p_i)\mathbf{1}^T\big]S\Big|_F^2 \leqslant \big|\Phi(p) - \phi(p_i)\mathbf{1}^T\big|_F^2\\ &\leqslant \sum_{j\neq i}|\phi(p_j) - \phi(p_i)|^2\\ &\leqslant (n-1)\mathfrak{a}(\mathfrak{d}(n))\zeta(p_i)\,.\end{aligned} \tag{D.18}$$

It follows from (5.30) and the fact that b has bounded partial derivatives that there exists a constant $k > 0$ such that

$$|N_i|_F^2 \leqslant k\zeta(p_i)|\dot{p}_i|^2 \tag{D.19}$$

for $1 \leqslant i \leqslant n$. This constant k depends on n, ε, and the bounds on the partial derivatives of b. Let σ be a constant such that $0 < \sigma < 1$ and

$$\sigma \leqslant \frac{\varepsilon(\gamma+\rho)}{(\gamma+\bar{\rho}^2)+2\varepsilon\bar{\varepsilon}}\,. \tag{D.20}$$

Then the positive definite matrices

$$P = \begin{bmatrix} I & -\sigma I \\ -\sigma I & I \end{bmatrix}, \qquad Q = \begin{bmatrix} (\gamma+\rho)I & 0 \\ 0 & \sigma\varepsilon I \end{bmatrix} \tag{D.21}$$

satisfy

$$(1-\sigma)I \leqslant P \leqslant (1+\sigma)I \tag{D.22}$$

and

$$\begin{aligned} &PF + F^T P + Q = \\ &\begin{bmatrix} -2L_P^\star(p) + (\rho-\gamma)I + 2\sigma L_I^\star(p) & \sigma\gamma I + \sigma L_P^\star(p) \\ \sigma\gamma I + \sigma L_P^\star(p) & -2\sigma L_I^\star(p) + \sigma\varepsilon I \end{bmatrix} \\ &\leqslant -\sigma \cdot \underbrace{\begin{bmatrix} \left(\frac{1}{\sigma}(\gamma+\rho) - 2\bar{\varepsilon}\right)I & -\gamma I - L_P^\star(p) \\ -\gamma I - L_P^\star(p) & \varepsilon I \end{bmatrix}}_{R(p)} \leqslant 0 \end{aligned} \tag{D.23}$$

because (D.20) implies that $R(p) \geqslant 0$. Let $\kappa > 0$ be such that

$$\kappa < \min\Big\{\frac{\gamma+\rho}{\sigma},\, \sigma\varepsilon\Big\}\,. \tag{D.24}$$

Then we have

$$PGG^TP = \begin{bmatrix} \sigma^2 I & -\sigma I \\ -\sigma I & I \end{bmatrix}$$

$$= P - \begin{bmatrix} (1-\sigma^2)I & 0 \\ 0 & 0 \end{bmatrix} \tag{D.25}$$

and thus also

$$\begin{aligned} Q - \frac{\kappa}{1+\sigma} PGG^TP & \\ = \begin{bmatrix} \big[\gamma + \rho + \kappa(1-\sigma)\big] I & 0 \\ 0 & \sigma\varepsilon I \end{bmatrix} & - \frac{\kappa}{1+\sigma} P \\ \geqslant \min\big\{\gamma + \rho + \kappa(1-\sigma),\, \sigma\varepsilon\big\} I - \kappa I & = \alpha I \,, \end{aligned} \tag{D.26}$$

where $\alpha = \min\{\gamma + \rho - \sigma\kappa,\, \sigma\varepsilon - \kappa\}$. It now follows that

$$PF + F^TP + \frac{\kappa}{1+\sigma} PGG^TP \leqslant -\alpha I \,. \tag{D.27}$$

We define the matrix

$$\Psi = \Omega P \Omega^T + \beta E \mathbf{1}\mathbf{1}^T E^T + W \mathbf{1}\mathbf{1}^T W^T \tag{D.28}$$

where $\beta > 0$ is a constant parameter. Defining

$$\xi = \frac{\gamma^2 (n+1)(\sigma+1)}{\kappa} \,, \tag{D.29}$$

we use (5.16), (D.5), (D.6), (D.11), (D.14), (D.17), and (D.27) to obtain

$$\dot{\Psi} = \Omega\big[PF + F^TP\big]\Omega^T + \gamma\sum_{i=0}^{n}\big[N_iG^TP\Omega^T + \Omega PGN_i^T\big]$$

$$- 2\beta\gamma E\mathbf{1}\mathbf{1}^TE^T + \beta Z\mathbf{1}\mathbf{1}^TE^T + \beta E\mathbf{1}\mathbf{1}^TZ^T \tag{D.30}$$

$$\leqslant \Omega\big[PF + F^TP + \frac{\kappa}{1+\sigma}\,PGG^TP\big]\Omega^T - \beta\gamma E\mathbf{1}\mathbf{1}^TE^T$$

$$+ \xi\sum_{i=0}^{n}N_iN_i^T + \frac{\beta}{\gamma}\,Z\mathbf{1}\mathbf{1}^TZ^T \tag{D.31}$$

$$\leqslant -\alpha\Omega\Omega^T - \beta\gamma E\mathbf{1}\mathbf{1}^TE^T + \max\big\{\frac{n\beta}{\gamma}, \frac{\xi}{\varepsilon^2}\big\}\,ZZ^T + \xi\sum_{i=1}^{n}N_iN_i^T\,. \tag{D.32}$$

Because $ES = -YS$ we also have

$$\Omega\Omega^T = YSS^TY^T + HH^T = ESS^TE^T + HH^T \tag{D.33}$$

and therefore

$$\dot{\Psi} \leqslant -\nu_1EE^T - \alpha HH^T + \nu_2\,ZZ^T + \xi\sum_{i=1}^{n}N_iN_i^T\,, \tag{D.34}$$

where

$$\nu_1 = \min\big\{\alpha,\, n\beta\gamma\big\} \tag{D.35}$$

$$\nu_2 = \max\big\{\frac{n\beta}{\gamma}, \frac{\xi}{\varepsilon^2}\big\}\,. \tag{D.36}$$

Defining the storage function $U = \mathrm{Tr}(\Psi)$ we see that

$$\dot{U} \leqslant -\alpha|H|_F^2 + \sum_{i=1}^{n}\Big[-\nu_1|e_i|^2$$

(D.37) $$+ \nu_2 |z_i|^2 + \xi k \zeta(p_i) |\dot{p}_i|^2 \Big] .$$

Furthermore, we can use (C.20) to bound (D.1) from above as

(D.38)
$$\dot{V} \leqslant \sum_{i=1}^{n} \Big[-\dot{p}_i^T \big[B + B^T \big] \dot{p}_i - 2c\zeta(p_i) |\dot{p}_i|^2 - \frac{1}{2} z_i^T \big[\Lambda + \Lambda^T \big] z_i + 2 e_i^T \Gamma \big[\Lambda + \Lambda^T \big]^{-1} \Gamma e_i \Big] .$$

We assume that (5.31) holds with

(D.39) $$\delta_1 = \frac{1}{2} \sqrt{\frac{\nu_1}{\nu_2}} , \qquad \delta_2 = \frac{4 \delta_1 \nu_2}{\xi k} ,$$

and we choose $\mu > 0$ so that

(D.40) $$\Gamma < \frac{\mu \nu_1}{2 \delta_1} I < \delta_1 \big[\Lambda + \Lambda^T \big] .$$

Upon taking inverses and then multiplying by Γ from the left and the right, we obtain

(D.41) $$\frac{1}{\delta_1} \Gamma \big[\Lambda + \Lambda^T \big]^{-1} \Gamma < \Gamma .$$

In particular, (D.40) and (D.41) imply the existence of a scalar constant $\nu > 0$ such that

(D.42) $$\frac{1}{2} \big[\Lambda + \Lambda^T \big] - \mu \nu_2 I \geqslant \nu I$$

(D.43) $$\mu \nu_1 I - 2 \Gamma \big[\Lambda + \Lambda^T \big]^{-1} \Gamma \geqslant \nu I .$$

Furthermore, (5.31) and (D.40) imply

(D.44) $$2c \geqslant \mu \xi k .$$

We then define $\Upsilon(p, \dot{p}, E, W) = V + \mu U$ and use (D.37), (D.38), (D.42), (D.43), and (D.44) to obtain

$$\dot{\Upsilon} \leqslant -\alpha\mu|H|_F^2 + \sum_{i=1}^{n}\left[-\dot{p}_i^T\left[B + B^T\right]\dot{p}_i - \nu|z_i|^2 - \nu|e_i|^2\right].$$

The rest of the proof mimics the proof of Theorem 8 above.

APPENDIX E

Stability of the Active Localization Example

Applying the design theory developed in section 2.3, we show the feedback system is stable for the one-time measurement approach using P estimators.

E.1. Choosing the global information f

For the one-time measurement approach, the global uncertainty matrix looks like

$$\begin{aligned}\frac{1}{n}P_{\text{global}}^{-1} &= \frac{1}{n}\sum_{i=1}^{n} T_i R_i^{-1} T_i^T \\ &= \frac{1}{n}\sum_{i=1}^{n}\begin{bmatrix} \frac{\cos^2(\theta_i)+\frac{1}{\alpha}\sin^2(\theta_i)}{f_r(r_i)} & -\frac{\sin(2\theta_i)}{2f_r(r_i)}(1-\frac{1}{\alpha}) \\ -\frac{\sin(2\theta_i)}{2f_r(r_i)}(1-\frac{1}{\alpha}) & \frac{\sin^2(\theta_i)+\frac{1}{\alpha}\cos^2(\theta_i)}{f_r(r_i)} \end{bmatrix}.\end{aligned}$$

Now we define the global information $f = [f^1\, f^2\, f^3]^T$ with

$$f^1 = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{f_r(r_i)} \tag{E.1}$$

$$f^2 = \frac{1}{n}\sum_{i=1}^{n}\frac{\cos(2\theta_i)}{f_r(r_i)} \tag{E.2}$$

$$f^3 = \frac{1}{n}\sum_{i=1}^{n}\frac{\sin(2\theta_i)}{f_r(r_i)} \tag{E.3}$$

Using triangular equalities, we obtain:

$$\frac{1}{n}P_{\text{global}}^{-1} = \begin{bmatrix} \frac{\alpha+1}{2\alpha}f^1 + \frac{\alpha-1}{2\alpha}f^2 & \frac{1-\alpha}{2\alpha}f^3 \\ \frac{1-\alpha}{2\alpha}f^3 & \frac{\alpha+1}{2\alpha}f^1 - \frac{\alpha-1}{2\alpha}f^2 \end{bmatrix} \tag{E.4}$$

Then the cost function can be written as

$$\begin{aligned} J(f,\beta) &= \det(P_{\text{global}}) \\ &= \frac{1}{\frac{n(\alpha-1)^2}{4\alpha^2}[(\frac{\alpha+1}{\alpha-1}f^1)^2 - (f^2)^2 - (f^3)^2]}. \end{aligned} \tag{E.5}$$

E.2. Bounding the global information f

From the sensor model it is easy to see $f_r(r_i) > a_0$. Therefore

$$\|f\|_2 \leq \sqrt{3}|f^1| \leq \frac{\sqrt{3}}{a_0} \tag{E.6}$$

So we can apply Theorem 3 to find the nonlinear damping gain Λ_i and next we give a conservative lower bound on Λ_i.

E.3. A lower bound on the nonlinear damping gain

We start by giving an alternative strategy to bound the matrix product $C_i C_i^T$ without solving any optimization problem. Given a square matrix $C_i \in \mathbb{R}^{m\times m}$, let λ_i^* be its eigenvalue with the largest absolute value. It is easy to verify that $C_i C_i^T < (\lambda_i^*)^2 I$. Furthermore, we have

$$|\lambda_i^*| < \|C_i\|_\infty = \max_j |\left.\frac{\partial^2 J}{\partial f_j \partial f}\right|_{f=\widetilde{f}_{ij}}|_\infty$$

$$\leq \sup_{f\in\Psi} \max_j \left| \frac{\partial^2 J}{\partial f_j \partial f}\bigg|_{f=f} \right|_\infty = \sup_{f\in\Psi} \left\| \frac{\partial^2 J}{\partial f^2} \right\|_\infty \tag{E.7}$$

From that we obtain the following theorem:

Proposition 4. *Using local information, each agent i can calculate a lower bound of the Hessian matrix:*

$$\left\| \frac{\partial^2 J}{\partial f^2} \right\|_\infty \leq \frac{n(\alpha^2-1) f_r^2(r_i)[n(3\alpha^2-4\alpha+1) f_r(r_i) + \alpha a_0^2]}{2\alpha a_0^2} \tag{E.8}$$

Proof. See Section E.4. □

Applying Theorem 3, we use the following control law:

$$\begin{bmatrix} \widetilde{u}_i^r \\ \widetilde{u}_i^\theta \end{bmatrix} = [I + \frac{(1+\mu+\eta_i^2) D_i^T K_p^{-1} D_i}{\lambda_{\min}}]^{-1} \begin{bmatrix} u_i^r \\ u_i^\theta \end{bmatrix} \tag{E.9}$$

with

$$\begin{aligned} \eta_i &= \frac{n(\alpha^2-1) f_r^2(r_i)[n(3\alpha^2-4\alpha+1) f_r(r_i) + \alpha a_0^2]}{2\alpha a_0^2} \\ u_i^r &= \det(x_i) \cdot \mathrm{tr}(2a_2(r_i - a_1) T_i R_i^{-2} \begin{bmatrix} 1 & 0 \\ 0 & \alpha \end{bmatrix} T_i^T x_i) \\ u_i^\theta &= \det(x_i) \cdot \mathrm{tr}(\big(A_i + A_i^T\big) x_i)/r_i \end{aligned}$$

Theorem 3 guarantees the stability of the coupled system.

E.4. A Lower Bound on the Hessian

Here we derive an upper bound of $\left\| \frac{\partial^2 J}{\partial f^2} \right\|_\infty$.

From (E.5), straightforward calculation gives :

$$\frac{\partial^2 J}{\partial f^2} = 8\rho^2 J^3 v v^T - 2\rho J^2 K \tag{E.10}$$

with

$$\rho = \frac{n(\alpha-1)^2}{4\alpha^2} \tag{E.11}$$

$$v = \begin{bmatrix} \frac{\alpha+1}{\alpha-1} f_1 & -f_2 & -f_3 \end{bmatrix}^T \tag{E.12}$$

$$K = \begin{bmatrix} \frac{\alpha+1}{\alpha-1} & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}. \tag{E.13}$$

Therefore,

$$\left\| \frac{\partial^2 J}{\partial f^2} \right\|_\infty \leq 8\rho^2 J^3 \|vv^T\|_\infty + 2\rho J^2 \|K\|_\infty \tag{E.14}$$

From (E.13) we know

$$\|K\|_\infty = \frac{\alpha+1}{\alpha-1}. \tag{E.15}$$

Using the fact $|f_i| \leq \frac{1}{a_0}$, we get

$$\|vv^T\|_\infty \leq \frac{(\alpha+1)(3\alpha-1)}{(\alpha-1)^2 a_0^2}. \tag{E.16}$$

To bound the global uncertainty, each agent can relate J to its own measurement uncertainty. Based on (6.23) we have

$$P_{\text{global}}^{-1} > T_i R_i^{-1} T_i^T > \frac{1}{\alpha f_r(r_i)} I. \tag{E.17}$$

Therefore

$$J = \det(P_{\text{global}}) < \alpha^2 f_r^2(r_i). \tag{E.18}$$

Plugging (E.15), (E.16) and (E.18) into (E.19) gives

$$\left\| \frac{\partial^2 J}{\partial f^2} \right\|_\infty \leq \frac{n(\alpha^2 - 1) f_r^2(r_i)[n(3\alpha^2 - 4\alpha + 1) f_r(r_i) + \alpha a_0^2]}{2\alpha a_0^2}. \tag{E.19}$$

For each agent i, this bound only requires local information to calculate.