Foundations and Trends® in Systems and Control

On the Control of Multi-Agent Systems: A Survey

Suggested Citation: Fei Chen and Wei Ren (2019), "On the Control of Multi-Agent Systems: A Survey", Foundations and Trends® in Systems and Control: Vol. 6, No. 4, pp 339–499. DOI: 10.1561/2600000019.

Fei Chen
State Key Laboratory of Synthetical Automation for Process Industries
Northeastern University, China
and
School of Control Engineering
Northeastern University at Qinhuangdao, China
fei.chen@ieee.org

Wei Ren
Department of Electrical and Computer Engineering
University of California, Riverside, USA
ren@ee.ucr.edu

now
the essence of knowledge
Boston — Delft

Contents

1 Introduction **341**

2 Basic concepts and definitions on multi-agent systems (MASs) **346**

2.1 What is an agent? 346
2.2 What is autonomy? 347
2.3 What is a multi-agent system (MAS)? 348
2.4 Network topologies for information exchange 348
2.5 Centralized vs. decentralized vs. distributed 352

3 Agent models **355**

3.1 Linear agent models 355
3.2 Nonlinear agent models 360

4 Cooperative tasks **367**

4.1 Consensus 367
4.2 Leader-following coordination 374
4.3 Flocking 377
4.4 Formation control 379
4.5 Coverage control 384
4.6 Distributed average tracking 386
4.7 Distributed estimation 390

4.8 Containment control and surrounding control 394
4.9 Distributed optimization 399

5 Research issues **403**
5.1 Network issues 403
5.2 Noise or disturbance 410
5.3 Connectivity maintenance 413
5.4 Time-triggered measurement (sampled data) 418
5.5 Event-triggered measurement 420

6 Algorithms **424**
6.1 Proportional (P) control 424
6.2 Proportional-integral (PI) control 426
6.3 Adaptive control 429
6.4 Model predictive control 431
6.5 Passivity-based control 433
6.6 Sliding-mode control 436
6.7 Finite-time and fixed-time control 438

7 Applications **441**
7.1 Multi-robot systems 441
7.2 Sensor networks 445
7.3 Smart grids 452
7.4 Machine learning 456
7.5 Social networks 458
7.6 Task migration of many-core microprocessors 462
7.7 Coordination of the charging of multiple electric vehicles . 464
7.8 Distributed heating, ventilation, and air-conditioning (HVAC) optimization 466

8 Conclusions and prospects **470**

References **475**

On the Control of Multi-Agent Systems: A Survey

Fei Chen[1,2] and Wei Ren[3]

[1] *State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, China; fei.chen@ieee.org*
[2] *School of Control Engineering, Northeastern University at Qinhuangdao, China*
[3] *Department of Electrical and Computer Engineering, University of California, Riverside, USA; ren@ee.ucr.edu*

ABSTRACT

Recent years have witnessed a trend to use networked multiple autonomous agents to accomplish complex tasks arising from space-based applications, smart grids, and machine learning. Such systems are referred to as multi-agent systems, where a global objective is achieved via the local interactions among the agents. The recent two decades have witnessed a rapid development of MASs in automatic control, but their root can be traced back much earlier. This paper reviews the research progress of MASs in past years. After briefly introducing the basic concepts and definitions, we discuss agents' dynamic models, including both linear and nonlinear models, describe different cooperating tasks, such as consensus, coordinating tracking, formation control, distributed average tracking, distributed estimation, containment control, surrounding control, and distributed optimization. We introduce various research issues for MASs, including time-delays, noise or disturbance, quantization, connectivity maintenance,

Fei Chen and Wei Ren (2019), "On the Control of Multi-Agent Systems: A Survey", Foundations and Trends® in Systems and Control: Vol. 6, No. 4, pp 339–499. DOI: 10.1561/2600000019.

and event-triggered control. We present different MAS control algorithms, including proportional control, proportional-integral control, adaptive control, model predictive control, passivity-based control, and nonsmooth control, and their applications in multi-robot systems, sensor networks, smart grid, machine learning, social networks, and many-core microprocessors.

1

Introduction

Nature has created a large number of multi-agent systems (MASs), where local interaction rules/mechanisms are exploited at different levels by groups of agents to achieve a common group objective. Schools of fish and flocks of birds are typical examples of MASs, which have fascinated scientists from rather diverse disciplines, such as physics, biology, and computer sciences. Thanks to the parallel characteristics, MASs can be used to solve engineering problems that are difficult or impossible for a single agent to accomplish. For example, in a large area, it is not possible to use a camera to cover the whole area; while a network of multiple collaborating cameras can be used to achieve the purpose. MASs are more robust — the malfunction of one agent or a small portion of agents typically will not affect the functionality of the system; MASs are scalable — no matter what the size of the system is, the computation and communication costs of MASs are kept at a reasonably low level.

Although the study of MASs can be traced back long ago [1–3], it was only at the beginning of the 21st century that MASs emerged as a separate research field. In 2005, the paper "coordination of groups of mobile autonomous agents using nearest neighbor rule" won the

prestigious George S. Axelby outstanding paper award from IEEE Transactions on Automatic Control, a top-notched journal in systems and control. After that, the International Federation of Automatic Control (IFAC) and American Automatic Control Council (AACC) organized a series of conferences/workshops focusing on MASs. IFAC established several technical committees having close ties with MASs. New journals with a primary interest in MAS or network systems are established, e.g., IEEE Transactions on Control of Network Systems and IEEE Transactions on Network Sciences and Engineering.

As a critical enabler of several research fields, MASs are quintessentially multidisciplinary. In the search for theories and applications, physicists, computer scientists, biologists, and others have all contributed to the development of MASs. Craig Reynold proposed three rules that lead to simulated flocking: collision avoidance, velocity matching, and flock centering [4]. Tamas Vicsek et al. introduced a model where the velocity of the particles is determined by a simple rule with random fluctuations, to investigate the emergence of group behavior [5]. On the other perspective, the study on MASs also provides insights for the related fields, leading to new developments in these fields. A hallmark of MAS control is the mathematical rigor where convergence/stability analysis plays a vital role, perhaps an inheritance from control theory.

Although MAS theory shares some similarities with other branches of natural sciences, there exist some fundamental difference. In natural sciences, methodological reductionism plays a central role and has achieved great success, which attempts to explain entire systems in terms of their components. Reductionism assumes that the interactions among system components (subsystems) are not essential and their effect is negligible. However, the assumption failed to work for MASs. The goal of MAS study is to understand and exploit the local interaction rules among agents, from which a global behavior can emerge. As a result, methodological reductionism becomes less useful for MAS study. Compared with single agent control where there already exists interaction among different systems components, e.g., sensing component and control component, MASs add another layer of interactions at a higher level. How to understand and exploit these interactions is the key to the success of MASs.

Many survey or tutorial papers [6–10] on MASs have been presented, along with a number of special issues and books [11–22]. The Proceedings of IEEE and the IEEE Control System Magazine publishes several overviews on MASs, including information consensus in multivehicle cooperative control [23], motion coordination with distributed information [24], consensus and cooperation in networked multi-agent systems [25], interconnected dynamic systems—an overview on distributed control [26], oscillator models and collective motion [27], motion coordination with distributed information [24], and collective motion, sensor networks, and ocean sampling [28]. Springer Encyclopedia and Wiley Encyclopedia publish a series of tutorial papers on multi-agent systems (e.g., [29–44]). Numerous special issues are published by the IEEE Transactions on Automatic Control [45], the Proceedings of the IEEE [46], and the IEEE Transactions on Robotics and Automation [47].

The first decade of the 21st century has witnessed a very dynamic development of the MAS theory, with rapid growth of applications. The research topics of MASs have been extended from consensus [48–51], formation control [52, 53], and flocking [54] to distributed estimation [55–57] and distributed optimization [58, 59]. The courses on MAS control also appear both at the undergraduate level and graduate level. In many engineering fields, it is not unusual to see that multiple agents work cooperatively to accomplish a complex task. The examples include distributed reconfigurable sensor networks, space-based interferometers, combat, surveillance, and reconnaissance systems, smart grids, and distributed machine learning. Although the problems arise from diverse application domains, they share some fundamental characteristics. First, agents have simple sensing, communication, and computation capabilities and function in a fully autonomously way; second, there is not a central decision maker or coordinator, and each agent makes its own decision by its local information, i.e., the system is distributed. Today, the research scope of MASs is still expanding.

Organization

The remainder of this paper is organized as follows.

- We begin in Chapter 2 with a brief introduction of basic concepts and definitions on MASs, where we will introduce the definitions of agents, autonomy, as well as multi-agent systems. Additionally, we will delineate the preliminaries on graph theory that is relevant in characterizing the network topologies among agents. Among other things, we will point out the difference among the three terms "centralized", "decentralized", and "distributed".

- In Chapter 3, we introduce the models describing the dynamics of agents. The linear models include first-integrator dynamics, double-integrator dynamics, and generic linear systems, while the nonlinear models involve Lagrangian systems, unicycle systems, along with attitude dynamics of rigid bodies.

- In Chapter 4, we describe different multi-agent cooperative tasks, including consensus, leader-following coordination, flocking, formation control, coverage control, distributed average tracking, distributed estimation, containment control and its inverse problem surrounding control, as well as distributed optimization.

- In Chapter 5, we present various research issues of MASs, including time-delays, quantization, packet loss, noise and disturbance, connectivity maintenance, sampled-data control, along with event-triggered control.

- In Chapter 6, we introduce different types of multi-agent control algorithms, including proportional control, proportional-integral control, adaptive control, model predictive control, passivity-based control, sliding-mode control, finite-time control, as well as fixed-time control.

- In Chapter 7, we summarize the applications of MASs in multi-robot systems, sensor networks, smart grids, machine learning, social networks, task migration of multi-core microprocessors, coordination of the charging of multiple electric vehicles, distributed heating, ventilation, and air conditioning optimization.

- Finally, in Chapter 8, we give a conclusion and list some unsolved problems of MASs from the authors' perspective.

The synopsis of this article falls between a survey paper and a book, which is reflected by the length of the article. Compared with a regular survey, our article provides a more thorough coverage on the topics of multi-agent systems, as each one of Chapters 3–7 can serve independently as a survey paper. It is noted that most multi-agent books in the market are monographs, focusing on the specific research interests of the authors. In comparison, our article offers a wider range of coverage, but it omits cumbersome technical details, such as proofs, in order to let the reader get a full picture of the field promptly.

2

Basic concepts and definitions on multi-agent systems (MASs)

2.1 What is an agent?

MASs is an interdisciplinary subject, which receives research interests from a variety of disciplines, such as computer science, social science, control engineering, and software engineering. Due to the gap between different research disciplines, it would be difficult to provide a unified definition of an agent that works for all subjects. The following definition is adapted from [60].

Definition 1 (Agent)**.** An agent is a system with the following features.

- Autonomy: it is at least partially autonomous;
- Perception: it is able to perceive the environment;
- Communication: it is able to communicate with other agents;
- Computation: it is able to do some simple computation.

Example 1 (Multi-robot rendezvous)**.** Consider a group of n mobile robots moving in a two-dimensional plane (see Fig. 2.1). Let p_i denote the position of robot i. Each robot is equipped with a GPS device

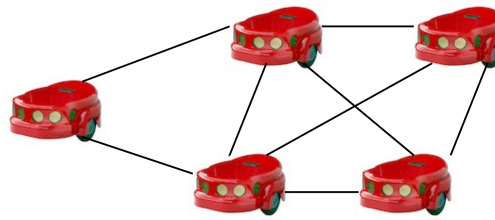


Figure 2.1: Multiple robots moving in a two-dimensional space [61].

and a communication device, which allows it to locate its position and communicate with neighboring robots. Robot i calculates the center of its position plus the positions of its neighboring robots and moves towards the center. By this way, all the robots will eventually rendezvous at a single location.

Depending on the research problem, an agent might represent a robot, a sensor node, or even a computer process. For instance, in Example 1, each robot can be viewed as an agent. In this example, a robot gets its position via GPS (perception), communicates the position information with neighboring robots (communication), calculates the center (computation), and makes its own control decision (autonomy).

2.2 What is autonomy?

The word "autonomous" is a combination of the Greek words "auto" (self) and "nomous" (rule) [62]. In MASs, agents have certain kind of "autonomy", which means that agents make their own control decisions on how to respond to its environment and the other agents without the direct interference from a supervisor. Instead of following commands like "move from place A to place B", a typical agent usually receives command like "move to the center of local neighboring agents". However, how to determine the neighboring agents and what is the local center are totally determined by the agent.

Autonomy is an essential characteristic of MASs. This is most desirable when there is no complete information of the operating environment

of MASs, or one does not have full control over the agents. For example, when sending a robot to space, the control command might not be sent/received in a timely manner due to significant time-delays.

2.3 What is a multi-agent system (MAS)?

A MAS is a system composed of two or more agents which work together in an environment to achieve a global task/objective via local interactions. Some desirable features of a MAS are summarized below.

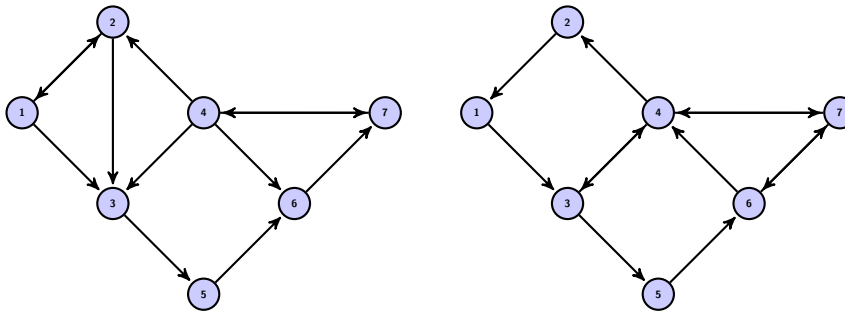
- Global task/objective: the global task/objective is beyond the capacity of a single agent and may not be known to all the agents.
- Local interaction: each agent can only use the information from its local neighbors. The neighboring relationships are normally determined by spatial proximity. For instance, in Example 1, a robot can use the information of the robots that are within its communication range.
- Robustness: if one or a few agents fail, the other agents should quickly adapt to the situation and respond accordingly such that the global task/objective is still achieved as desired.
- Scalability: if the number of agents increases dramatically, the computing, communication, and control cost of the agents do not increase dramatically.

MASs are capable of solving problems that are difficult or impossible for a single agent to complete. They arise from a wide spectrum of research fields, such as computer sciences, control engineering, physics, chemistry, biology, to name just a few.

2.4 Network topologies for information exchange

Basic definitions

It is typical to model the information exchange among agents by a graph, which is represented by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the node set and $\mathcal{E}$ is the edge set. In this graph, each node represents an agent.



(a) Unbalanced digraph (b) Balanced digraph

Figure 2.2: Illustration of digraphs.

A graph can be categorized as either directed or undirected. In a directed graph (digraph), an edge (i, j) is an ordered pair of nodes and indicates that information flows from agent i to j, but not vice versa (see Fig. 2.2). If $(j, i) \in \mathcal{E}$, then node j is called an in-neighbor of node i. We use $\mathcal{N}_i^{\text{in}} \triangleq \{j \in \mathcal{V} \,|\, (j, i) \in \mathcal{E}\}$ to denote the set of *in-neighbors* of node i. The *in-degree* of node i is defined as the cardinality of the set $\mathcal{N}_i^{\text{in}}$. Similarly, we can define the set of *out-neighbors* of node i as $\mathcal{N}_i^{\text{out}} \triangleq \{j \in \mathcal{V} \,|\, (i, j) \in \mathcal{E}\}$ and the *out-degree* as the cardinality of $\mathcal{N}_i^{\text{out}}$. A digraph is called *balanced* if for each node the in-degree equals the out-degree.

Connectedness notions

A *directed path* of length k is a sequence of directed edges of the form $(v_1, v_2), (v_2, v_3), \ldots, (v_k, v_{k+1})$. Next, we introduce a few key connectedness notions for digraphs.

Definition 2. A digraph is *strongly connected* if there is a directed path from every node to every other node.

Definition 3. We say a digraph has a *directed spanning tree* if there exists one node, called *root*, having a directed path to every other node.

Note that a strongly connected graph must contain a directed spanning tree, since by definition each node is a root.

Fig. 2.2(a) gives a digraph, of which the node set is $\mathcal{V} = \{1, 2, 3, 4, 5, 6, 7\}$ and the edge set is $\mathcal{E} = \{(1,2), (1,3), (2,1), (2,3), (3,5), (4,2), (4,3), (4,6), (4,7), (5,6), (6,7), (7,4)\}$. For node 1, its in-neighbor set is $\mathcal{N}_1^{\text{in}} = \{2\}$ and out-neighbor set is $\mathcal{N}_1^{\text{out}} = \{2, 3\}$. Its in-degree and out-degree are, respectively, 1 and 2. It can be verified that the digraph in Fig. 2.2(a) is strongly connected. As a result, it must contain a directed spanning tree (in fact, every node is a root for the digraph). Fig. 2.2(b) shows a balanced digraph. Unlike Fig. 2.2(a), the in-degree of each node in Fig. 2.2(b) equals its out-degree.

Matrices associated with a digraph

The adjacency matrix of a digraph is defined by $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ where $a_{ij} = 1$ if $(j, i) \in \mathcal{E}$, and $a_{ij} = 0$ otherwise. The in-degree matrix of a digraph is defined as $D^{\text{in}} \triangleq \text{diag}([d_1^{\text{in}}, \ldots, \text{diag}_n^{\text{in}}])$, where $d_i^{\text{in}} \triangleq \sum_{j=1}^{n} a_{ij}$. The Laplacian matrix of the digraph is then defined by

$$L = D^{\text{in}} - A,$$

Note that the row sums of L are all zeros, implying that zero is an eigenvalue of L with the corresponding eigenvector $\mathbf{1}$. The incidence matrix $E = [e_{ij}] \in \mathbb{R}^{n\times m}$ of a directed graph is a matrix such that $e_{ij} = 1$ if the edge e_j leaves node i, 1 if it enters node i, and 0 otherwise (the opposite sign convention is also employed in the literature). Next, we present an example to illustrate the definition of the Laplacian matrix.

Example 2. Consider the following digraph consisting of four nodes and five directed edges (see Fig. 2.3). The adjacency matrix of the above graph is given by

$$A = \begin{bmatrix} 0 & 0 & 0.5 & 0 \\ 2 & 0 & 1.5 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0.5 & 0 & 0 \end{bmatrix},$$

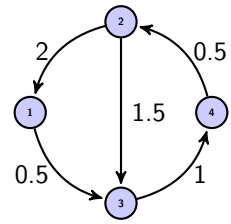


Figure 2.3: Illustration of a digraph.

while the Laplacian matrix is

$$L = \begin{bmatrix} 0.5 & 0 & -0.5 & 0 \\ -2 & 3.5 & -1.5 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & -0.5 & 0 & 0.5 \end{bmatrix}.$$

If we order the edges as follows: $(1,3)$, $(3,4)$, $(4,2)$, $(2,1)$, and $(2,3)$, the incidence matrix is a 4×5 matrix given by

$$E = \begin{bmatrix} -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 & 0 \end{bmatrix}.$$

Undirected graphs

In an undirected graph, an edge (i,j) is an unordered pair of nodes, which indicates that information both flows from agent i to agent j and vice versa (see Fig. 2.4). An *undirected path* in an undirected graph is defined analogously. An undirected graph is *connected* if there is an undirected path between every pair of distinct nodes.

Time-varying graphs

In what follows, we define several connectedness notions related to time-varying digraphs.

Definition 4 (Jointly strongly connected)**.** Let $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$, $k = 1, \ldots, p$, denote all possible network topologies of a time-varying digraph $\mathcal{G}(t)$

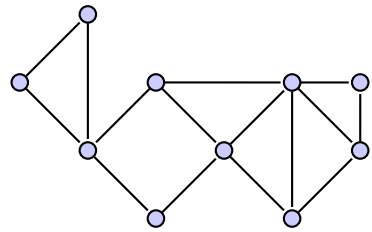


Figure 2.4: Illustration of a connected undirected graph.

during the time interval $[t_1, t_2]$. We say $\mathcal{G}(t)$ is jointly strongly connected in $[t_1, t_2]$, if the graph union $(\mathcal{V}, \cup_{k=1}^{p} \mathcal{E}_k)$ is strongly connected.

Definition 5 (joint directed spanning tree)**.** We say $\mathcal{G}(t)$ has a directed spanning tree jointly during $[t_1, t_2]$, if the graph union $(\mathcal{V}, \cup_{k=1}^{p} \mathcal{E}_k)$ has a directed spanning tree.

Similarly, we can extend the above definitions to undirected graphs.

2.5 Centralized vs. decentralized vs. distributed

A centralized multi-agent system is illustrated in Fig. 2.5. Apparently, there is a central node/agent, the red one, in the MAS, since it has access to the information of all the other agents. To break the connectivity of the MAS, one can take down the red node. In this way, the MAS is split into four separated agents, and no cooperation can be achieved. Thus, one of the main drawbacks of a centralized MAS is that it is not robust to the attack or the failure on the central node. On the other hand, the central node needs to communicate/process the information of all the other agents. When the system is of a small size, i.e., the number of the agents is small, this would not be a problem. However, as the size of the system increases dramatically, this will add a lot of communication and computation burden to the central node, which is usually difficult to implement. Therefore, the second problem of centralized MASs is the poor scalability properties, resulting from the complexity of communication and computation.

Fig. 2.6 gives an example of a decentralized MAS. In this architecture, the network can be split into several "communities", in which there is a

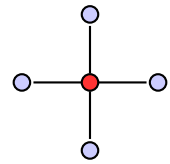


Figure 2.5: Illustration of a centralized MAS.

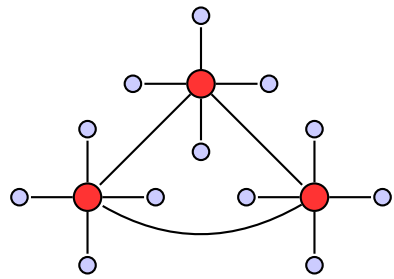


Figure 2.6: Illustration of a decentralized MAS.

local central node which can communicate or interact with the rest of the nodes in the community. This kind of network structure is better than the centralized one since the failure of a single agent can only affect the connectivity of its community and does not affect the other parts of the network.

Fig. 2.4 illustrates an example of a distributed MAS. In this architecture, one can not figure out which node to attack. Taking down any node/agent will not affect the functionality of the whole MAS since no agent plays a central role in the system. The distributed property is a critical requirement when we design MASs. The key difference between a distributed and a decentralized MAS is that a decentralized MAS has locally centralized agents (see the red node in Fig. 2.6), while a distributed MAS has no centralized agents at all, no matter globally or locally. It is worth pointing out that these two terms are sometimes used interchangeably in the context of multi-agent control.

In summary, while parallelism can be attained by assigning a complex task to different agents, robustness is a primary benefit of MASs that have redundant agents. MASs can tolerate failures caused by one or

more agents, which are lacked in centralized systems, for which if the centralized agent fails, the entire system crashes.

3

Agent models

In the field of automatic control, an agent model refers to a mathematical description of the dynamics of the agent. An agent model helps us understand how the agent moves and evolves, in particular how the control input is injected in the agent. The models of an agent are not unique. Depending on the assumptions and the problem to be addressed, one should select an appropriate model for the agents. It is worth pointing out that the model of an agent is not accurate. It is only an approximation of the underlining dynamics of the agents. In the remainder of this chapter, we will use $n \in \mathbb{R}^+$ to denote the size of a MAS, i.e., the number of agents of the MAS. We use x_i and u_i to denote, respectively, the state and control input of the ith agent, where $i \in \{1, \ldots, n\}$ is the index of the agent.

3.1 Linear agent models

3.1.1 First-order systems

A first-order system is described by the following differential equation:

$$\dot{x}_i = a_i x_i + b_i u_i, \tag{3.1}$$

where $x_i \in \mathbb{R}^p$ is the state of agent i, $u_i \in \mathbb{R}^q$ the control input, and $a_i \in \mathbb{R}$ and $b_i \in \mathbb{R}$ two constant parameters. In the special case $a_i = 0$ and $b_i = 1$, the model reduces to

$$\dot{x}_i = u_i, \tag{3.2}$$

which is the so-called single integrator model. Depending on the problems on hand, x_i and u_i might have a different interpretation. In the following, we give examples to demonstrate the application of the first-order models.

Example 3 (Coordinated tracking [63]). Let $p_i \in \mathbb{R}^p$, $i = 0, \ldots, n$, be the position of the ith vehicle moving in the $\mathbb{R}^p$ space. In addition, we assume that p_0 is a leader. The dynamics of the ith agent can be described by

$$\dot{p}_i = v_i, \quad i = 0, \ldots, n \tag{3.3}$$

where v_i is the velocity of the ith agents. If the leader is static and its position is available to some vehicles, i.e., $v_0 \equiv 0$, then it is possible to design a linear control algorithm such that all the vehicles will track the leader asymptotically. If v_0 is nonzero but bounded, it is possible to design nonlinear control algorithm to achieve the same goal.

Example 4 (Velocity consensus of vehicles [64]). Let v_i be the velocity of the ith vehicle, m_i the mass, and f_i the force generated by the engine. It follows from Newton's second law that

$$m_i \dot{v}_i = f_i. \tag{3.4}$$

By viewing v_i as the state and $\frac{1}{m_i} f_i$ as the control input, the model reduces to the single-integrator model (3.2). By designing a consensus input $f_i = \sum_{j \in \mathcal{N}_i} (v_j - v_i)$, where $\mathcal{N}_i$ denotes a subset of the vehicles whose positions can be sensed by the ith vehicle.

Example 5 (Multi-species growth [65]). Let x_i denote the population of the ith specie. A simple model to describe the growth of the specie is given by

$$\dot{x}_i = b_i x_i - m_i x_i + u_i = (b_i - m_i) x_i + u_i, \tag{3.5}$$

where b_i is the birth rate, m_i is the mortality rate, and $u_i \triangleq u_{i,S} + u_{i,H}$ with $u_{i,S}$ denoting the effect from other species and $u_{i,H}$ the effect from human beings. Let x_i^* denote the ideal population of the ith specie. If $u_{i,S}$ is a linear function of x_j, then a linear feedback $u_i = -a_i(x_i - x_i^*)$ can drive the populations of all the species to their desired states.

3.1.2 Double integrator

Double-integrator dynamics is one of the most fundamental systems in control applications [66–68]. Practical examples of double-integrator systems include single-axis spacecraft rotation and rotary crane motion, to name just a few. The dynamics of double-integrator systems is given by

$$\ddot{x}_i = u_i \tag{3.6}$$

where $x_i \in \mathbb{R}^p$ denotes the position and $u_i \in \mathbb{R}^p$ is the control inputs. Double-integrator dynamics can be used to model one-dimensional translational motion. Let x_i denote the position of a vehicle with mass m_i moving in a one-dimensional space and f_i be the force added on the car. According to Newton's second law, we have

$$f_i = m_i a_i = m_i \ddot{x}_i \tag{3.7}$$

where a_i is the acceleration. Eq. (3.7) can be rewritten as

$$\ddot{x}_i = \frac{f_i}{m_i} \triangleq u_i.$$

Double-integrator dynamics can also be used to model one-dimensional rotational motion. In this case, the model becomes

$$\ddot{\theta}_i = \frac{\tau_i}{I_i} \triangleq u_i, \tag{3.8}$$

where $\ddot{\theta}_i \in \mathbb{R}$ denotes the angular acceleration, $\tau_i \in \mathbb{R}$ the external torque, and $I_i \in \mathbb{R}$ the moment of inertia.

Example 6 (Flocking [54])**.** Consider a group of dynamic particles with the following equation of movements:

$$\ddot{p}_i = u_i, \tag{3.9}$$

where p_i denotes the position of particle i, and u_i its control input. Design the control inputs

$$u_i = \sum_{j \in \mathcal{N}_i} (\phi_a(\|\dot{p}_j - \dot{p}_i\|_\sigma)\mathbf{n}_{ij} + \sum_{j \in \mathcal{N}_i} a_{ij}(q)(p_j - p_i), \tag{3.10}$$

where $\phi_a(\cdot)$ is a smooth potential function depicted by Fig. 3.1. It is shown that under (3.10), the agent will generate the flocking behavior.

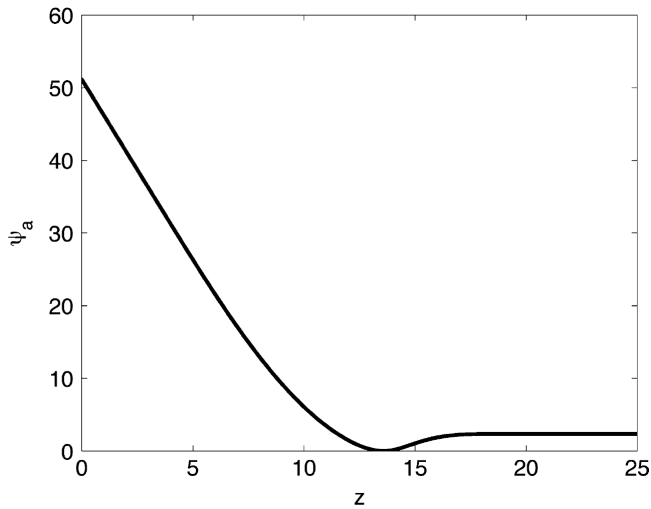


Figure 3.1: Smooth potential $\phi_a(\cdot)$ [54].

3.1.3 General linear systems

A general linear model is given by

$$\dot{x}_i = Ax_i + Bu_i, \tag{3.11}$$

where $x_i \in \mathbb{R}^p$ is the state, $u_i \in \mathbb{R}^q$ the control input, $A \in \mathbb{R}^{p \times p}$ the state matrix, and $B \in \mathbb{R}^{p \times q}$ the input matrix. The single-integrator model and double-integrator model are all special cases of (3.11). When $A = 0$ and B=1, implying $p = q = 1$, it reduces to a single integrator. For the double-integrator model (3.6), define $x = [x_1, x_2]$, where $x_1 = x$ and $x_2 = \dot{x}$. Eq. (3.6) can be written in the form of (3.11), where

$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ and $B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. When system output is included, (3.11) becomes

$$\begin{aligned} \dot{x}_i &= Ax_i + Bu_i, \\ y_i &= Cx_i + Du_i, \end{aligned} \tag{3.12}$$

where $y_i \in \mathbb{R}^o$ denotes the output, $C \in \mathbb{R}^{o\times p}$, and $D \in \mathbb{R}^{o\times q}$.

Example 7 (Consensus via dynamic output feedback [69])**.** Suppose n linear dynamic systems:

$$\begin{aligned} \dot{x}_i &= Ax_i + Bu_i, \\ y_i &= Cx_i, \end{aligned} \tag{3.13}$$

where (A, B, C) is stabilizable and detectable. Agent i collects its neighboring output information

$$z_i = \sum_{j=1}^{n} a_{ij}(y_j - y_i). \tag{3.14}$$

The information z_i is filtered by a stable filter $\mathcal{K}(s)$ and is fed back to agent i by

$$\mathcal{U}_i(s) = \mathcal{K}(s)\mathcal{Z}_i(s), \tag{3.15}$$

where $\mathcal{K}(s) = C_\mathcal{K}(sI - A_\mathcal{K})^{-1}B_\mathcal{K}D_\mathcal{K}$, and $\mathcal{U}_i(s)$ and $\mathcal{Z}_i(s)$ are, respectively, the Laplace transforms of u_i and z_i. The state-space form for $\mathcal{K}(s)$ is given by

$$\begin{aligned} \dot{x}_\mathcal{K} &= A_\mathcal{K}x_\mathcal{K} + B_\mathcal{K}z_\mathcal{K}, \\ u_\mathcal{K} &= C_\mathcal{K}x_\mathcal{K} + D_\mathcal{K}z_\mathcal{K}, \end{aligned}$$

with $A_\mathcal{K}$ being Hurwitz. Define $\bar{A} = \begin{bmatrix} A & BC_\mathcal{K} \\ \mathbf{0} & A_\mathcal{K} \end{bmatrix}$, $\bar{B} = \begin{bmatrix} BD_\mathcal{K} \\ B_\mathcal{K} \end{bmatrix}$, and $\bar{C} = [C, \mathbf{0}]$. It is shown in [69] that the control input (3.15) guarantees an asymptotic consensus if all eigenvalues of the matrix A lies in the closed left half complex plane, and the communication graph of the agents contains a directed spanning tree.

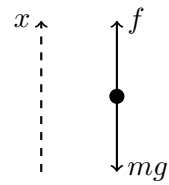


Figure 3.2: A massive point subjected to the gravitational force and an external force.

3.2 Nonlinear agent models

3.2.1 Lagrangian systems

A Lagrangian system is a system which is described by the Euler–Lagrange equations, a set of nonlinear ordinary differential equations. The Euler–Lagrange equations are derived from the energy of the system. They are a powerful modeling tool and describe a wide variety of physical systems, particularly robotic systems.

We first show how to write Newton's second law in the form of Euler–Lagrange equations. Consider a massive point of m subjected to the gravitational force mg and a external force u_i (see Fig. 3.2). According to Newton's second law, we have

$$m\ddot{x}_i = u_i - mg. \tag{3.16}$$

The kinetic energy of the mass is given by

$$K_i = \frac{1}{2} m\dot{x}_i^2,$$

and the potential energy is given by

$$P_i = mgx_i.$$

Then,

$$L_i = K_i - P_i = \frac{1}{2} m\dot{x}_i^2 - mgx_i, \tag{3.17}$$

is called the Lagrangian of the system. Using

$$m\ddot{x}_i = \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial K_i}{\partial \dot{x}_i},$$
$$mg = \frac{\partial P_i}{\partial x_i}, \tag{3.18}$$

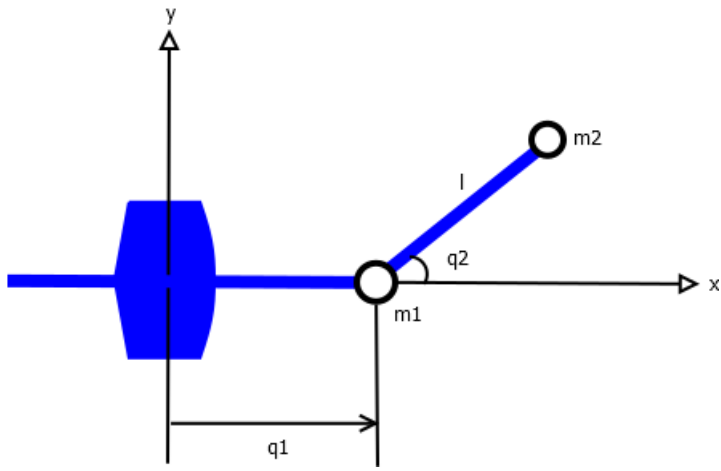


Figure 3.3: Illustration of a robot arm.

and noticing $\frac{\partial P_i}{\partial \dot{x}_i} = \frac{\partial K_i}{\partial \dot{x}_i}$ and $\frac{\partial L_i}{\partial x_i} = -\frac{\partial P_i}{\partial x_i}$, we can rewrite (3.16) as

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L_i}{\partial \dot{x}_i} - \frac{\partial L_i}{\partial x_i} = u_i. \tag{3.19}$$

Eq. (3.19) is the so-called Euler–Lagrange equation of the system.

Next, we show how to apply Euler–Lagrange equations to a simple robotic system. Consider a robot arm with two massive points m_1 and m_2 (see Fig. 3.3). The kinetic energies of the two points are given, respectively, by

$$\begin{aligned} K_1 &= \frac{1}{2} m_1 \dot{l}_1, \\ K_2 &= \frac{1}{2} m_2 \left(\frac{d}{dt}(l_1 + l_2 \cos\theta_2) \right)^2 + \frac{1}{2} m_2 \left(\frac{d}{dt}(l_2 \sin\theta_2) \right)^2 \\ &= \frac{1}{2} m_2 \left(\dot{l}_1^2 + l_2^2 \dot{\theta}_2^2 - 2 l_2 \dot{l}_1 \dot{\theta}_2 \sin\theta_2 \right). \end{aligned} \tag{3.20}$$

Let $q_1 = l_1$ and $q_2 = \theta_2$. The kinetic energy can be rewritten as

$$\begin{aligned} K_1 &= \frac{1}{2} m_1 \dot{q}_1, \\ K_2 &= \frac{1}{2} m_2 (\dot{q}_1^2 + l_2^2 \dot{q}_2^2 - 2 l_2 \dot{q}_1 \dot{q}_2 \sin q_2). \end{aligned} \tag{3.21}$$

The potential energies are given by

$$\begin{aligned} P_1 &= 0, \\ P_2 &= m_2 g l_2 \sin q_2. \end{aligned} \tag{3.22}$$

The Laplacian of the system is given by

$$\begin{aligned} L &= \sum_{i=1,2} K_i - \sum_{i=1,2} P_i \\ &= \frac{1}{2} m_1 \dot{q}_1 + \frac{1}{2} m_2 (\dot{q}_1^2 + l_2^2 \dot{q}_2^2 - 2 l_2 \dot{q}_1 \dot{q}_2 \sin q_2) - m_2 g l_2 \sin q_2. \end{aligned} \tag{3.23}$$

The Euler–Lagrange equation of the system is

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = u_i, \quad i = 1, 2, \tag{3.24}$$

where u_i is the force on q_i. Substituting (3.23) into (3.24) yields

$$\begin{aligned} (m_1 + m_2)\ddot{q}_1 - m_2 l_2 \ddot{q}_2 \sin q_2 - m_2 l_2 \dot{q}_2^2 \cos q_2 &= u_1, \\ m_2 l_2^2 \ddot{q}_1 - m_2 l_2 \ddot{q}_1 \sin q_2 + m_2 g l_2 \cos q_2 &= u_2, \end{aligned}$$

which can be written equivalently as

$$D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = u, \tag{3.25}$$

where $q = [q_1, q_2]^T$ is the generalized coordinate, $D(q)$ is the inertia matrix for the system, $C(q, \dot{q})$ represents the Coriolis forces as well as the damping, $G(q)$ gives the forces due to potential energy, and u is the generalized applied force.

Example 8 (Cooperative robotic arm control)**.** Suppose that there are n robot arms, whose dynamics is described by

$$D_i(q_i)\ddot{q}_i + C_i(q_i, \dot{q}_i)\dot{q}_i + G_i(q_i) = u_i, \quad i = 1, \ldots, n. \tag{3.26}$$

where q_i denotes the generalized coordinate of arm i. We are interested in moving objects via the cooperation of the robot arms (see Fig. 3.4 for illustration). Compared with single-arm systems, the benefits of using multiple arms include easier locomotion and higher precision.

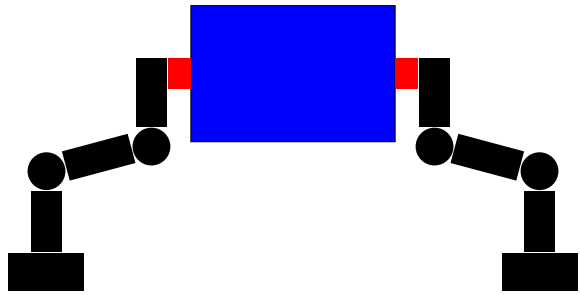


Figure 3.4: Cooperative arm control.

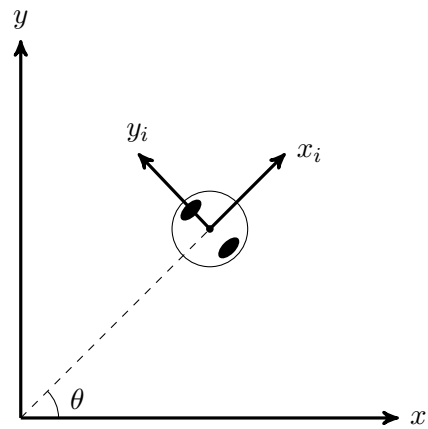


Figure 3.5: The unicycle model.

3.2.2 Unicycle model

The unicycle model (see Fig. 3.5) is given by

$$\begin{aligned} \dot{x}_i &= v_i \cos\theta_i, \\ \dot{y}_i &= v_i \sin\theta_i, \\ \dot{\theta}_i &= w_i, \end{aligned} \tag{3.27}$$

where (x_i, y_i) denotes the coordinates of robot i in the plane, θ_i is the angle between the heading direction and the x-axis, v_i and w_i are, respectively, the linear and angular velocities. The model describes a wide variety of kinematics of mobile robots, which include front-wheel drive automobiles, automated guided vehicles (AGVs) with a drive and

steering wheel, and mobile robots with two independent drive wheels. It follows from the model that $\frac{\mathrm{d}y_i}{\mathrm{d}x_i} = \tan\theta_i$, which leads to

$$\frac{\dot{y}_i}{\dot{x}_i} = \frac{\sin\theta_i}{\cos\theta_i}. \tag{3.28}$$

This is equivalent to

$$\dot{y}_i \cos\theta_i - \dot{x}_i \cos\theta_i = 0, \tag{3.29}$$

which is the nonholonmic constraint of the model.

Example 9 (Stabilization of unicycles to a line [70])**.** Suppose there are n wheeled robots with the unicycle model (3.27). We want to stabilize the robots such that they form a straight line. Let robots 1 and n be the edge robots, which denotes the two ends of the line. The following control algorithm

$$\begin{aligned} v_i(t) &= k \sum_{j=i-1,i+1} \begin{bmatrix} x_j - x_i \\ y_j - y_i \end{bmatrix}^T \begin{bmatrix} \cos\theta_i \\ \sin\theta_i \end{bmatrix}, \\ \omega_i(t) &= \cos(t) \end{aligned}$$

guarantees that all robots converge to a uniform distribution on the line segment specified by the two edge robots for sufficiently small $k > 0$.

3.2.3 Attitude dynamics of rigid bodies

According to Euler's rotation theorem, any rotation in the $\mathbb{R}^3$ space of a rigid body is equivalent to a rotation by a given angle ϕ (called the Euler angle) about a unit vector $u = [u_1, u_2, u_3]$ (called the Euler axis). Unit quaternions give a simple way to encode this axis-angle representation in four numbers. A unit quaternion is defined as

$$Q = [q^T, \eta]^T \in \mathbb{R}^4, \tag{3.30}$$

where $q = u\sin(\frac{\phi}{2}) \in \mathbb{R}^3$ and $\eta = \cos\frac{\phi}{2} \in \mathbb{R}$. A unit quaternion is not unique because Q and $-Q$ represent the same attitude. However, the uniqueness can be achieved by restricting the angle ϕ in the range $0 \leq \phi \leq \pi$. The attitude dynamics of a rigid body using unit quaternions

is described by

$$\begin{aligned}\dot{q} &= -\frac{1}{2}\omega \times q + \frac{1}{2}\eta\omega,\\ \dot{\eta} &= -\frac{1}{2}\omega \cdot q,\\ J\dot{\omega} &= -\omega \times (J\omega) + \tau,\end{aligned} \tag{3.31}$$

where ω is the angular velocity, J the inertia tensor, and τ the control torque.

Example 10 ([71]). Consider a group of n rigid bodies, where the motion of the ith rigid body is governed by

$$\dot{Q}_i = \frac{1}{2}T(Q_i)w_i, \quad I_{f_i}\dot{w}_i = \Gamma_i - S(w_i)I_{f_i}w_i, \quad i = 1, \ldots, n, \tag{3.32}$$

where $w_i \in \mathbb{R}^3$ is the angular velocity of the ith rigid body, $I_{f_i} \in \mathbb{R}^{3\times 3}$ is the inertial matrix of the ith rigid body, and Γ_i is the external torque input. The unit quaternion $Q_i = [q_i^T, \eta_i]$ describes the attitude of the rigid bodies. The matrix $T(Q_i) \in \mathbb{R}^{4\times 3}$ is given by

$$T(Q_i) = \begin{bmatrix} \eta_i I_3 + S(q_i) \\ -q_i^T \end{bmatrix},$$

where $S(x)$ is the skew-symmetric matrix satisfying $S(x_1)x_2 = x_1 \times x_2$ for any vector $x_1, x_2 \in \mathbb{R}^3$ with $\times$ denoting the vector cross product. Associate each rigid body with the following virtual system

$$\dot{Q}_{v_i} = \frac{1}{2}T(Q_{v_i})w_{v_i}, \tag{3.33}$$

where $Q_{v_i} = [q_{v_i}^T, \eta_{v_i}^T]$ is the unit quaternion representing the attitude of the virtual system, and w_{v_i} is the virtual angular velocity input to be designed. For two unit-quaternion $Q_i = [q_i^T, \eta_i]^T$ and $Q_j = [q_j^T, \eta_j^T]^T$, we define their multiplication as

$$Q_i \bigodot Q_j \triangleq ((\eta_i q_j + \eta_j q_i + S(q_i)q_j)^T, \eta_i\eta_j - q_i^T q_j)^T.$$

Let Q_i^e be the discrepancy between the attitudes of the ith rigid body and its virtual system, i.e., $Q_i^e = Q_{v_i}^{-1} \bigodot Q_i$, where

$$\dot{Q}_i^e = \frac{1}{2}T(Q_i^e)w_i^e, \quad w_i^e = w_i - R(Q_i^e)w_{v_i}, \tag{3.34}$$

with $R(Q_i^e)$ being the rotational matrix related to Q_i^e. Design the input torque as

$$\Gamma_i = H_i(\dot{w}_{v_i}, w_{v_i}, Q_i^e) - k_i^p q_i^e - k_i^d \tilde{q}_i^e,$$

where $H_i(\dot{w}_{v_i}, w_{v_i}, Q_i^e) = (I_{f_i} R(Q_i^e)\dot{w}_{v_i} + S(R(Q_i^e)w_{v_i})I_{f_i}R(Q_i^e)w_{v_i})$, k_i^p and k_i^d are strictly positive scalar gains, q_i^e is the vector part of Q_i^e and $\tilde{q}_i^e$ is the vector part of $\tilde{Q}_i^e = P_i^{-1} \bigodot Q_i^e$, with

$$\dot{P}_i = \frac{1}{2} T(P_i)\beta_i, \quad \beta_i = \lambda_i \tilde{q}_i^e,$$

and $\lambda_i > 0$. Suppose the communication among rigid bodies is delayed. Let $\bar{Q}_{v_{ij}} = Q_{v_j}^{-1}(t - \tau_{ij}) \bigodot Q_{v_i}$ denote the relative attitude between the ith and jth virtual system, where τ_{ij} denotes the time-delay. Design the virtual angular velocity as

$$\dot{w}_{v_i} = -k_i^w w_{v_i} - \sum_{j=1}^{n} a_{ij} \bar{q}_{ij},$$

where $k_i^w > 0$, a_{ij} is the (i,j)th entry of the adjacency matrix, and $\bar{q}_{ij}$ is the vector part of $\bar{Q}_{ij}$. It is shown that if the delays are bounded, then attitude synchronization can be achieved.

4

Cooperative tasks

4.1 Consensus

In a network of agents, consensus means that the states of agents converge to the same value. Depending on the physical background, an agent might represent a robot, a sensor node, or even a computer process. Accordingly, the state of an agent might denote the position of a robot, the local time clock of a sensor node, or the workload of a computer process. The value to which all the agents converge is the consensus value. A consensus algorithm is a protocol or a law which specifies how the agents utilize their collected information to reach a consensus. If a consensus algorithm can be implemented in such a way that each agent only uses the information of its local neighbors in the network, then the consensus algorithm is distributed.

The research on consensus might be traced back to 1959, where a consensus model is built to describe how a group of individuals reach a consensus on the probability distribution for an unknown parameter [1]. Consensus serves as a fundamental problem in distributed computing, where computer processes need to reach a consensus on certain data value during computation [72, 73]. Examples of consensus in distributed computing include whether to commit a transaction to a database,

agreeing on the identity of a leader, etc. For consensus, the research focus was initially placed on the convergence analysis of consensus under certain graph assumptions [48–52], and later has been extended from various perspectives, such as agent dynamics [66, 68, 69, 74, 75] and nonlinear control strategies [76–79]. The material in this section is most adopted from [33].

4.1.1 A typical consensus algorithm

Consider the following multi-agent system with the single integrator dynamics

$$\dot{x}_i = u_i, \quad i = 1, \ldots, n, \tag{4.1}$$

where x_i is the state of agent i, u_i the control input, n the number of agents, and $\dot{x}_i$ the derivative of x_i. The consensus algorithm is typically designed as follows:

$$u_i = \sum_{j=1}^{n} a_{ij}(x_j - x_i), \tag{4.2}$$

where a_{ij} is the (i, j)th entry of the adjacency matrix of a graph $\mathcal{G}$ that describes the information flows among the agents. That is, $a_{ij} > 0$ if agent i can utilize the information of agent j, and $a_{ij} = 0$ otherwise. Here, if $a_{ij} > 0$, then agent j is a neighbor of agent i. The graph $\mathcal{G}$ is referred to as the network topology of the multi-agent system. The basic idea behind eq. (4.2) is to drive the state of each agent towards the states of its neighbors such that all the agents' states will converge to a common value. This common value is the consensus value. The algorithm eq. (4.2) is distributed because each agent only uses the information of its local neighbors. For a multi-agent system, a distributed algorithm is preferable to a centralized one due to its scalability. The consensus closed-loop system is then given by

$$\dot{x}_i = \sum_{j=1}^{n} a_{ij}(x_j - x_i). \tag{4.3}$$

In the discrete-time case, the differential equation eq. (4.3) becomes the following difference equation:

$$x_i(k+1) = x_i(k) + \epsilon \sum_{j=1}^{n} a_{ij}[x_j(k) - x_i(k)], \tag{4.4}$$

where $x_i(k)$ denotes the state of agent i at time k, and $\epsilon > 0$ is the step size. Let $p_{ij} = \epsilon a_{ij}$ and $p_{ii} = 1 - \epsilon \sum_{j=1}^{n} a_{ij}$. Eq. (4.4) can be rewritten as

$$x_i(k+1) = \sum_{j=1}^{n} p_{ij} x_j(k). \tag{4.5}$$

It can be verified that if ϵ is sufficiently small, then $p_{ii} > 0$. In addition, the row sums $\sum_{j=1}^{n} p_{ij} = 1, \forall i = 1, \ldots, n$, which implies that $P = [p_{ij}]$ is a row stochastic matrix with positive diagonal entries (A row stochastic matrix is a real nonnegative matrix with each row summing to one). The idea behind eq. (4.5) is that each agent updates its next state to be the weighted average of its own and neighbors' current states.

4.1.2 Consensus examples in practical systems

Example 11 (Multi-robot Rendezvous [80])**.** Consider a group of n mobile autonomous robots moving in a two-dimensional plane. Let p_i denote the position of robot i and let S_i denote its sensing region. Each robot can sense the positions of all other agents within its sensing region. The objective of rendezvous is to design local control strategies such that all the robots can eventually rendezvous at a single location, i.e., reaching a consensus on their positions p_i.

Example 12 (Clock Synchronization in Wireless Sensor Networks [81, 82])**.** In a wireless sensor network, each sensor node maintains a local time clock

$$c_i = a_i t + b_i,$$

where a_i is the skew rate of the clock that determines the clock speed, and b_i is the offset of the clock function which is primarily caused by inaccuracy when first setting the clock. To reach clock synchronization

in the sensor network, all the nodes must compensate for their clock parameters a_i and b_i so that all clocks have zero offset error and all tick at the same rate. That is, a consensus can be reached for a_i and b_i respectively, where the consensus value for the offset b_i is zero.

Example 13 (Workload Balancing for Distributed Computing [83])**.** Consider a distributed computing system consisting of n computer processes, which collaborate with each other, and a set of tasks (workloads), which have to be executed in the system. Let $q_i(t)$ denote the queue length of the tasks of process i at time t. Denote $p_i(t)$ the productivity of process i, which determines process i's speed of executing tasks. The objective of workload balancing is to balance the fraction $q_i(t)/p_i(t)$ such that the execution time of the tasks can be minimized. The problem can be solved effectively by running a consensus algorithm on $q_i(t)/p_i(t)$.

Example 14 (Target Tracking in Distributed Camera Networks [84, 85])**.** Consider a network of multiple cameras monitoring an area which contains several moving targets. The state (position) of all targets at time k is represented by a vector $x(k)$. Each camera has a specific detection area and can detect some or none of the targets. The particular targets identified by each camera determines whether $z_i(k)$, the measurement of camera i, consists of multiple or no measurements of the components of $x(k)$. At each time k, camera i uses a consensus-based Kalman filter and the measurement $z_i(t)$ to update its estimate, denoted by $\hat{x}_i(k)$, of $x(k)$, where the consensus algorithm is incorporated with the Kalman filter to keep the estimates at different cameras cohesive.

Example 15 (Altitude Alignment of Unmanned Air Vehicles (UAVs) [86])**.** Suppose that a team of UAVs is initialized at different altitudes. The center of the UAVs is defined as the average of all UAVs' altitudes and is not known *a priori*. Each UAV controls its vertical climb rate using the relative altitudes of its neighbors such that all the UAVs will reach a consensus on their altitudes while keeping the center unchanged. It turns out that the above altitude alignment problem can be solved by an average consensus problem.

4.1.3 Characterizations of consensus

Let span($\mathbf{1}$) denote the space spanned by the vector with all ones. A formal definition of consensus is given below.

Definition 6 (Consensus)**.** The system (4.3) is said to reach a consensus if and only if $\lim_{t\to\infty}[x_i(t) - x_j(t)] = 0$ for all $i \neq j$, or equivalently the stack vector x converges to span($\mathbf{1} \otimes I_p$), with p being the dimension of x_i.

Note that the row sums of the Laplacian matrix are all ones, indicating that a nonzero vector belonging to span($\mathbf{1}$) must be an eigenvector of the Laplacian matrix corresponding to the zero eigenvalue. Thus, using the properties of the Laplacian matrix, an important result [48, 49] can be stated as follows: if the undirected graph (respectively, digraph) is connected (respectively, has a directed spanning tree), then

$$Lx = 0 \quad \text{iff} \quad x \in \text{span}(\mathbf{1}). \tag{4.6}$$

That is, Lx is a measurement of the consensus error and $Lx = 0$ can be used to characterize consensus.

Recall the definition of E as the incidence matrix of the graph $\mathcal{G}$ (cf. Section 2.4). The term $E^T x$ is a vector which contains the entry $x_i - x_j$ for $(i, j) \in \mathcal{E}$. It thus follows that

$$E^T x = 0 \quad \text{iff} \quad x_i - x_j = 0,\ \forall (i, j) \in \mathcal{E}. \tag{4.7}$$

If the undirected graph (respectively, digraph) is connected (respectively, has a directed spanning tree), then eq. (4.7) can be further written as

$$E^T x = 0 \quad \text{iff} \quad x_i - x_j = 0,\ \forall i \neq j.$$

That is, $E^T x$ is another measurement of the consensus error and can be used to characterize consensus.

An important result regarding Laplacian matrices is given as follows.

Theorem 1 ([87])**.** For an undirected graph, the Laplacian matrix L is symmetric and positive semi-definite. Zero is a simple eigenvalue of L if and only if the graph is connected.

In addition, L can be diagonalized as

$$L = U^T \Lambda U, \tag{4.8}$$

where $U \in \mathbb{R}^{n\times n}$ is a unitary matrix satisfying $U^T U = UU^T = I$, and $\Lambda \triangleq \text{diag}([\lambda_1, \ldots, \lambda_n])$ with $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ denoting the eigenvalues of L. Recall the definition of U and Λ in eq. (4.8) for an undirected graph. Let $\tilde{U} \in \mathbb{R}^{(n-1)\times n}$ be the matrix constructed from U by removing the first row and let $\tilde{\Lambda} = \text{diag}([\lambda_2, \ldots, \lambda_n])$. It can be verified that $L = \tilde{U}^T \tilde{\Lambda} \tilde{U}$ by using $\lambda_1 = 0$. If $\tilde{U}x = 0$, then $Lx = U^T \Lambda U x = \tilde{U}^T \tilde{\Lambda} \tilde{U} x = 0$. If $Lx = 0$, then $U^T \Lambda U x = 0$, which yields that $x^T U^T \Lambda U x = 0$. This is equivalent to $\tilde{U}x = 0$ for an undirected and connected graph. The above statement can be summarized as follows: if the undirected graph $\mathcal{G}$ is connected, then

$$\tilde{U}x = 0 \quad \text{iff} \quad Lx = 0. \tag{4.9}$$

That is, $\tilde{U}x$ is a measurement of the consensus error for undirected graphs.

4.1.4 Consensus for double-integrator systems

Consider a multi-agent system governed by the double-integrator dynamics:

$$\begin{aligned} \dot{x}_i &= v_i, \\ \dot{v}_i &= u_i, \quad i = 1, \ldots, n, \end{aligned} \tag{4.10}$$

where x_i is the position of agent i, v_i is its velocity, and u_i is the control input to be designed. A consensus algorithm for (4.10) is proposed in [88] as

$$u_i = -\sum_{j=1}^{n} a_{ij}((r_i - r_j) + \gamma(v_i - v_j)), \tag{4.11}$$

where a_{ij} is the (i, j)th entry of the adjacency matrix, and γ is a positive gain.

Note that (4.11) does not explicitly take into account actuator saturation. To tackle the issue, Ref. [89] proposes the following consensus

algorithm with a bounded control input:

$$u_i = -\sum_{j=1}^{n}(a_{ij}\tanh(K_x(x_i - x_j)) + b_{ij}\tanh(K_v(v_i - v_j))), \quad i = 1, \ldots, n, \tag{4.12}$$

where K_x and K_v are positive-definite diagonal matrices, a_{ij} and b_{ij} are, respectively, the (i, j)th entry of the adjacency matrices A and B for positions and velocities, and tanh is the hyperbolic tangent function defined component-wise. Here, A and B can be chosen differently. It is shown in [89] that if the interaction graphs for A and B are both undirected and connected, then under (4.12), a consensus can be reached asymptotically for the double-integrator system (4.10).

At the end of this section, we highlight some unsolved challenges and opportunities for consensus.

- *Network topology in control.* Currently, the network topology is mainly used to describe the information flows among agents. Its role in control algorithm design has not been fully appreciated. Sometimes it might be desirable to design or choose different control schemes or parameters for a multi-agent system according to its network topology. This leads to two subproblems: one is how to characterize the network topology, and the other is how to define the control law according the chosen characterization of the network. It might be possible to develop a totally new network-dependent control scheme for multi-agent systems, which can be a good complement to control theory.

- *Consensus in data mining.* Data mining is a subfield of computer science which aims at leaning information from data and transforming it into an appropriate structure for further use. Currently, most data mining algorithms are centralized. It is possible to borrow the idea of consensus to redesign the centralized data mining algorithms such that the newly designed algorithms can be implemented in a distributed manner, while maintaining satisfactory performance. For instance, consensus might be used in aggregation of clustering, whose objective is to find a single clustering that is better than the existing ones.

4.2 Leader-following coordination

4.2.1 Coordinated tracking (leader-following consensus)

In coordinated tracking, the final consensus value is determined by a leader agent, labeled as 0 without loss of generality. The leader agent can be a physical or virtual one. All the other agents are called followers. The state of the leader might be time-invariant or time-varying and is available to only a portion of the followers. The objective of coordinated tracking is that the states of the followers should finally track the state of the leader. Let r_0 denote the state of the leader. If r_0 is time-invariant (e.g., the leader is governed by the single integrator dynamics $\dot{r}_0 = u_0$ with $u_0 = 0$), then the consensus algorithm (4.2) can be slightly modified to reach coordinated tracking. The coordinated tracking algorithm for (4.1) takes the following form:

$$u_i = \sum_{j=1}^{n} a_{ij}(x_j - x_i) + a_{i0}(r_0 - x_i), \tag{4.13}$$

where $a_{i0} > 0$ if follower i can access the state information of the leader, and $a_{i0} = 0$ otherwise. The coordinated tracking algorithm (4.13) can be seen as a special case of the normal consensus algorithm (4.2), where the leader is viewed as an agent without neighbors.

Define the coordinated tracking error for each agent as $e_i = x_i - r_0$. Eq. (4.13) can be written in terms of e_i as

$$\dot{e}_i = \sum_{j=1}^{n} a_{ij}(e_j - e_i) + a_{i0}e_i. \tag{4.14}$$

Let $A_0 = \mathrm{diag}([a_{10}, \ldots, a_{n0}])$ and $e = [e_1^T, \ldots, e_n^T]^T$. We can write (4.14) in a vector form as

$$\dot{e} = -(L + A_0)e. \tag{4.15}$$

The dynamical behavior of (4.15) is totally determined by the eigenvalues of the matrix $L + A_0$. The network topology $\mathcal{G}$ describes the information flows among the followers, but it does not contain the information about the leader agent. In order to include the leader information, we need to define an extended graph $\bar{\mathcal{G}} = (\bar{\mathcal{V}}, \bar{\mathcal{E}})$, where

$\bar{\mathcal{V}} = \mathcal{V} \cup \{0\}$ with node 0 denoting the leader agent and $\bar{\mathcal{E}}$ is the set of the edges which includes the information flows from the leader to the followers. If the graph $\bar{\mathcal{G}}$ has a directed spanning tree (or equivalently the leader has directed paths to all followers), then all the eigenvalues of $L + A_0$ are negative [48] and the system (4.15) is asymptotically stable, i.e., $\lim_{t\to\infty} e(t) = 0$. That is, the coordinated tracking is reached.

Other works on coordinating tracking include coordinated tracking for robots modeled by Lagrangian dynamics [53], coordinated tracking under switching interconnection topology [90], coordinated tracking subject to time-varying delays [91], coordinated tracking for heterogeneous systems [92, 93], coordinated tracking via asymptotic decoupling [53], and finite-time coordinated tracking via terminal sliding-mode control [94]. Future research perspectives might include the application of coordinated tracking techniques in multi-camera systems, simultaneous detection and tracking, vision-based coordinated tracking, and coordinated tracking in cluttered environment.

4.2.2 Controllability for leader-follower consensus networks

Controllability is a fundamental property for multi-agent systems, which addresses the important question that whether the system can be driven from any initial state to any target state in a finite time. One conventional approach to controlling network systems, e.g., a group of mobile robots, is to construct or select some leaders, possibly virtual, who can induce the other agents to generate a desired global behavior. There are three elements that affect the controllability of a leader-follower network: the selection of leader nodes, the interacting dynamics of the agents, as well as the network topology.

The agent dynamics

Consider a network of n agents. We divide the vertex set $\mathcal{V} = \mathcal{V}_L \cup \mathcal{V}_F$ into two disjoint sets, the leader set $\mathcal{V}_L$ and the follower set $\mathcal{V}_F$. Assume that the followers are governed by the following consensus dynamics:

$$\dot{x}_i = -\sum_{j=1}^{n} a_{ij}(x_i - x_j), \quad i \in \mathcal{V}_F,$$

where x_i is the state of agent i, and a_{ij} is the (i,j)th entry of the adjacency matrix. Additionally, we assume that the control signals are injected to the leaders via

$$x_i = u_i, \quad i \in \mathcal{V}_L.$$

Suppose that the agents are indexed in such a way that the first n_f agents are followers, and the last n_l agents are leaders, where $n_l + n_f = n$. If the interaction topology among the agents is undirected, we can decompose the Laplacian matrix as

$$L = -\begin{bmatrix} A & B \\ B^T & C \end{bmatrix},$$

where $A = A^T \in \mathbb{R}^{n_f \times n_f}$, $B \in \mathbb{R}^{n_f \times n_l}$, and $C \in \mathbb{R}^{n_l \times n_l}$. For notational convenience, we consider the one dimensional case and can write the network dynamics as (see [95])

$$\dot{x} = Ax + Bu. \tag{4.16}$$

The objective here is to characterize network controllability using purely graphic language instead of using the standard rank tests.

External equitable partitions

A partition of a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is a map $\pi : \mathcal{V} \to \{C_1, \ldots, C_k\}$, where $\pi(i)$ denotes the cell to which node i belongs. The inverse operator $\pi^{-1}(C_i) \triangleq \{j \in \mathcal{V} \mid \pi(j) = C_i\}$ returns the set of nodes belonging to the cell C_i. Define the node-to-cell degree

$$\deg_\pi(i, C_j) = |\{k \in \mathcal{V} \mid \pi(k) = C_j,\ (i,k) \in \mathcal{E}\}|.$$

A partition π is called an external equitable partition (EEP) if for all $C_i, C_j \in \text{range}(\pi)$, where $i \neq j$,

$$\deg_\pi(k, C_j) = \deg_\pi(l, C_j), \quad \forall k, l \in \pi^{-1}(C_i).$$

A necessary condition for single-leader networks

Consider the case of a single leader. We are interested in those EEPs placing the leader node in a single cell, i.e., $\pi^{-1}(\pi(N)) = \{N\}$, and

we refer such EEPs as leader-invariant. We say a leader-invariant is maximal if its codomain has the smallest cardinality. The maximal leader-invariant always exists and is unique [96], and we use π^* to denote it. The following gives a key result characterizing a necessary condition for complete controllability in terms of π^*.

Theorem 2 ([97]). The networked system in (4.16) is completely controllable only if $\mathcal{G}$ is connected, and $\pi^{*-1}(\pi^*(i)) = \{i\}$, for all $i \in \mathcal{V}$.

Other works on controllability of MASs include structural controllability [98], the effect of time-delays [99], the effect of correlations [100], controllability transition and nonlocality [101], robustness analysis of network controllability [102], controllability of networks of networks [103], and minimal edge addition [104]. Despite the aforementioned works, there is much room for the study of controllability of MASs, since the analytical relationships between controllability and the structural characteristics of a random network model are far from clear. This could be a future research topic.

4.3 Flocking

Consider a group of dynamic agents with the following second-order dynamics:

$$\begin{aligned} \dot{q}_i &= p_i, \\ \dot{p}_i &= u_i, \end{aligned}$$

where q_i, p_i, and u_i are, respectively, the position, velocity, and control input of agent i. Let $r > 0$ denote the common sensing/communication radius of the agents. A proximity net is then defined by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and

$$\mathcal{E} = \{(i, j) \in \mathcal{V} \times \mathcal{V} \,|\, \|q_i - q_j\| < r,\ i \neq j\}.$$

In flocking, one seeks a configuration where each agent is equally distanced from its neighbors, i.e.,

$$\|q_i - q_j\| = d,\ j \in \mathcal{N}_i(q), \tag{4.17}$$

where $\mathcal{N}_i(q) = \{j \in \mathcal{V} \,|\, \|q_i - q_j\| < r,\ i \neq j\}$.

In [54], the authors proposes two flocking algorithms. The first algorithm is given by

$$u_i = \sum_{j\in\mathcal{N}_i} \phi_\alpha(\|q_j - q_i\|_\alpha)n_{ij} + \sum_{j\in\mathcal{N}_i} a_{ij}(q)(p_j - p_i),$$

where $\|z\|_\alpha = \frac{1}{\epsilon}(\sqrt{1+\epsilon\|z\|^2} - 1)$ with $\epsilon > 0$, $\phi_\alpha(z)$ is a potential with a global minimum at $z = d$, and n_{ij} is a vector along the line connecting q_i and q_j. Under the first algorithm it can be shown that the system almost converges to a configuration satisfying (4.17) with the same velocity.

The second algorithm is designed as

$$\begin{aligned} u_i = &\sum_{j\in\mathcal{N}_i} \phi_\alpha(\|q_j - q_i\|_\alpha)n_{ij} + \sum_{j\in\mathcal{N}_i} a_{ij}(q)(p_j - p_i) \\ &- c_1(q_i - q_r) - c_2(p_i - p_r), \end{aligned}$$

where q_r and p_r denote, respectively, the position and velocity of a leader agent, whose dynamics are governed by

$$\begin{aligned} \dot{q}_r &= p_r, \\ \dot{p}_r &= f_r(q_r, p_r). \end{aligned} \tag{4.18}$$

The function $f_r(\cdot,\cdot)$ determines how the leader agent moves. For instance, if $f_r \equiv 0$, then the leader will move along a straight line with a constant velocity. It is proved that the algorithm (4.18) embodies all the three rules of Reynold, i.e., flocking centering, collision avoidance, and velocity matching.

Algorithm (4.18) requires that all the agents have the access to the leader agent's information. This restriction is relaxed in [105] by considering the algorithm

$$\begin{aligned} u_i = &\sum_{j\in\mathcal{N}_i} \phi_\alpha(\|q_j - q_i\|_\alpha)n_{ij} + \sum_{j\in\mathcal{N}_i} a_{ij}(q)(p_j - p_i) \\ &- h_i[c_1(q_i - q_r) + c_2(p_i - p_r)], \end{aligned} \tag{4.19}$$

where $h_i = 1$ if agent i is informed and $h_i = 0$ otherwise. For a leader with constant velocity, it is shown that the distance between informed agents and the leader is upper bounded, and the velocities of the agents approach to the leader's velocity asymptotically.

The stability properties of a distributed flocking algorithm over networks that change arbitrarily fast (no dwell time between consecutive switches) is presented in [106] via nonsmooth analysis. The flocking behavior of multiple agents which have significant inertias and evolve on a balanced information graph is analyzed in [64]. It is shown that flocking algorithms that neglect agents' inertial effect can cause unstable group behavior. To incorporate this inertial effect, the passive decomposition technique is employed, which decomposes the closed-loop group dynamics into two decoupled systems: a shape system representing the internal group shape and a locked system describing the motion of the center-of-mass.

4.4 Formation control

The objective of formation control is to drive a group of agents to form and maintain a pre-specified geometric pattern. Usually, the geometric pattern is described by a set of constant vectors p_{ij} ($(i,j) \in \mathcal{E}$), which specifies the desired relative position (state) between agents i and j. To reach a formation, we need to design control algorithms such that $x_j - x_i - p_{ji} \to 0$, where x_i denotes the position of agent i. There are various approaches to formation control, such as consensus-based approaches [107], behavior-based approaches [108], and virtual structure-based approaches [109]. The information used in formation control can be categorized as absolute information, relative information, and distance information.

4.4.1 Absolute information

Consider a multi-agent system with the following dynamics:

$$\dot{p}_i = u_i, \tag{4.20}$$

where p_i is the position and u_i is the control input. We assume that each agent has access of its absolute position information with respect to a global coordinate frame. The desired formation is denoted by $p^* = [p_1^*, \ldots, p_n^*]$, where p_i^* is the desired position of agent i. A straightforward

approach is to control each agent separately using feedback control, i.e.,

$$u_i = -(p_i - p_i^*). \tag{4.21}$$

It can be shown that each agent will converge to p_i^* asymptotically. Even though the collaboration between agents is not necessary in this case, the introduction of agent collaboration can be beneficial. We can design the new control input as

$$u_i = -(p_i - p_i^*) - \sum_{j=1}^{n} a_{ij} \left((p_i - p_i^*) - (p_j - p_j^*) \right). \tag{4.22}$$

The collaboration term $-\sum_{j=1}^{n} a_{ij} \left((p_i - p_i^*) - (p_j - p_j^*) \right)$ can increase the convergence speed of the system.

Ren and Atkins study a formation control problem based on absolute information for double-integrator agent dynamics and derives necessary and sufficient conditions for convergence [88]. Similar ideas are also employed for nonholonomic agents, such as the unicycle model [110, 111].

4.4.2 Relative information

Consider the formation control system

$$\dot{x}_i = \sum_{j=1}^{n} a_{ij}(x_j - x_i - p_{ji}). \tag{4.23}$$

Suppose that the formation is feasible, i.e., there exist x_i^*, $i = 1, \ldots, n$, satisfying $x_j^* - x_i^* = p_{ji}$ for $(j, i) \in \mathcal{E}$. Define the new state $\tilde{x}_i = x_i - x_i^*$. Eq. (4.23) can then be rewritten as

$$\dot{\tilde{x}}_i = \sum_{j=1}^{n} a_{ij}(\tilde{x}_j - \tilde{x}_i), \tag{4.24}$$

which is a consensus system for $\tilde{x}_i$. It is worth pointing out some nontrivial facts in the above transformation. First, in the formation control law (4.23), the formation is specified by relative positions. Second, the formation control law only uses relative position information. These two facts play a critical role in the feasibility of transforming a formation control algorithm to a consensus algorithm.

A more complex dynamics is considered in [112]. Consider the following agent dynamics:

$$\dot{x}_i = P_A x_i + P_B u_i, \tag{4.25}$$

where x_i and u_i are agents' states and control inputs, and P_A and P_B are two matrices. Each agent receives the following information

$$\begin{aligned} y_i &= P_{C_1} x_i, \\ z_i &= \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} z_{ij}, \\ z_{ij} &= P_{C_2}(x_i - x_j), \end{aligned} \tag{4.26}$$

where $\mathcal{N}_i$ denotes the neighbors of agent i. A dynamic controller is proposed as follows:

$$\begin{aligned} \dot{v}_i &= K_A v_i + K_{B_1} y_i + K_{B_2} z_i, \\ u_i &= K_C v_i + K_{D_1} y_i + K_{D_2} z_i. \end{aligned} \tag{4.27}$$

The closed-loop system can be rewritten in a compact way as

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix}, \tag{4.28}$$

where

$$\begin{aligned} A_{11} &= I_N \otimes (P_A + P_B K_{D_1} P_{C_1}) + (I_N \otimes P_B K_{D_2} P_{C_2})(L \otimes I_n), \\ A_{12} &= I_N \otimes P_B K_C, \\ A_{21} &= I_N \otimes K_{B_1} P_{C_1} + (I_N \otimes K_{B_2} P_{C_2})(L \otimes I_n), \\ A_{22} &= I_N \otimes K_A, \end{aligned} \tag{4.29}$$

and x and v are, respectively, the stack vectors of x_i and v_i and L is the Laplacian matrix. It is shown in [112] that a local controller K can stabilize the formation dynamics in (4.28) iff it simultaneously stabilizes the set of N systems

$$\begin{aligned} \dot{x}_i &= P_A x_i + P_B u_i, \\ y_i &= P_{c_1} x_i, \\ z_i &= \lambda_i P_{C_2} x_i, \end{aligned} \tag{4.30}$$

where λ_i are the eigenvalues of the Laplacian matrix. Formation control with relative information has been studied for single-integrator dynamics [113, 114], double-integrator dynamics [115, 116], general linear dynamics [117], and nonlinear dynamics [118, 119].

4.4.3 Distance information

In this case, the desired formation is given by the desired distances between agents. Consider the single-integrator multi-agent system (4.20). The desired formation is defined as

$$\|p_i - p_j\| = d_{ij}, \tag{4.31}$$

where d_{ij} is the desired distance between agents i and j. Define the potential function for agent i as

$$\phi_i(p) \triangleq \frac{k_p}{2}\gamma_{ij}(\|p_i - p_j\|), \tag{4.32}$$

where $\gamma_{ij}(\|p_i - p_j\|) = k_p\left(\|p_i - p_j\|^2 - d_{ij}^2\right)^2$ with $k_p > 0$. Then, a gradient descent control law [120] can be designed as

$$u_i = -\frac{\phi_i(p)}{\partial p_i}. \tag{4.33}$$

4.4.4 Rigid formation control for double-integrator dynamics

Consider a multi-agent system whose interaction topology is described by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with n nodes and m edges. Let $x_i \in \mathbb{R}^d$, $d \in \{2, 3\}$, denote the position of agent i. The vector

$$z = (\mathcal{I} \otimes I_d)x$$

describes the relative positions between the agents, where $z = [z_1^T, \ldots, z_m^T]$ with z_k denoting the relative position corresponding to the kth edge. Define the following distance map

$$r_{\mathcal{G}} = (\|x_i - x_j\|^2)_{(i,j)\in\mathcal{E}} = Z^T z,$$

where $Z \triangleq \text{diag}(z_1, z_2, \ldots, x_m) \in \mathbb{R}^{dm\times m}$. The notion of rigidity matrix plays a central role in studying graph rigidity (see [121] for the definition

of rigid formation), which is defined as

$$R(x) = \frac{1}{2}\frac{\partial r_{\mathcal{G}}(x)}{\partial x} = Z^T(\mathcal{I} \otimes I_d).$$

It is useful to note that the entries of $R(x)$ only involve relative position information. As a result, we will sometimes write $R(x)$ as $R(z)$. Particularly, if $R(x) = dn - d(d+1)/2$, we call the formation infinitesimally rigid.

In [122], the authors investigate rigid formation control problems whose target formation is infinitesimally rigid. Let $d_{k_{ij}}$ denote the desired length of edge k linking agents i and j. Similarly, let

$$e_{k_{ij}} = \|p_i - p_j\|^2 - d_{ij} = \|z_k\|^2 - d_{k_{ij}}^2 \tag{4.34}$$

denote the squared distance error for edge k. The distance error vector is then given by $e = [e_1, \ldots, e_m]^T$. Ref. [122] proposes two kinds of rigid formation control systems. One uses a velocity damping term, which aims at stabilizing a rigid formation while achieving a stationary formation, i.e., all agents should come to a rest finally. The system takes the following form:

$$\begin{aligned} \dot{x}_i &= v_i, \\ \dot{v}_i &= -k_i v_i - \sum_{j \in \mathcal{N}_i} \left(\|x_i - x_j\|^2 - d_{k_{ij}}^2 \right) (p_i - p_j), \end{aligned} \tag{4.35}$$

where k_i is a positive gain that can be freely selected. Define

$$\psi(x, v) \triangleq \frac{1}{2}\sum_{i \in \mathcal{V}} \|v_i\|^2 + \frac{1}{4}\sum_{(i,j) \in \mathcal{E}} (\|x_i - x_j\|^2 - d_{k_{ij}}^2)^2.$$

The system (4.35) can be rewritten in a compact way as

$$\begin{aligned} \dot{x} &= \nabla_v \psi = v, \\ \dot{v} &= -(K \otimes I_d)\nabla_v \psi - \nabla_p \psi = -(K \otimes I_d)v - R^T(z)e(z). \end{aligned} \tag{4.36}$$

The other system is to achieve a flocking behavior with both velocity consensus and formation stabilization. The overall system can be described by the following equation:

$$\begin{aligned} \dot{x}_i &= v_i, \\ \dot{v}_i &= \sum_{j \in \mathcal{N}_i} (v_j - v_i) - \sum_{j \in \mathcal{N}_i} (\|x_i - x_j\|^2 - d_{k_{ij}}^2)(x_i - x_j). \end{aligned} \tag{4.37}$$

Similarly, (4.37) can be rewritten as

$$\begin{aligned} \dot{x} &= \nabla_v \psi = v, \\ \dot{v} &= -(\mathcal{L} \otimes I_d) - R^T(z)e(z), \end{aligned} \tag{4.38}$$

where $\mathcal{L}$ is the Laplacian matrix. It has been revealed in [122] that both systems are capable of achieving local formation stabilization.

A graph theoretical framework which allows a formal definition of formation and the issues arising from the uniqueness of graph realizations is discussed in [123]. In [124], the authors propose algorithms to coordinate a formation of mobile agents when the agents can measure the distances to their respective neighbors. It is shown that the proposed stop-and-go strategy can stabilize a generically minimally rigid formation. Ref. [125] uses potential functions to design formation control laws where the target formation is specified by an undirected infinitesimally rigid graph. A new technique based on complex Laplacian is introduced in [126] to address the problems of which formation shapes determined by inter-agent relative positions can be formed. A global stability analysis of the closed-loop formation control dynamics is addressed in [127], where a differential geometric approach is pursued, and purely algebraic conditions for local stability of invariant embedded submanifolds are derived. The formation control problem with collision avoidance can also be posed as a nonlinear differential game [128]. Future perspectives of formation control include task-triggered formation control, rigid formation control for the three-dimensional space, fault-tolerant formation control, bearing-only formation control, as well as bounded assignment formation control.

4.5 Coverage control

Let Q be a convex polytope in $\mathbb{R}^n$. Let $\phi : Q \mapsto \mathbb{R}^+$ be a density function, which represents the probability of some event taking place over Q. Let $P = [P_1, \ldots, P_n]$ denote the location of n moving sensors. A function $f(\|q - p_i\|)$ evaluates the sensing performance of the location q provided by sensor i. Let $W = \{W_1, \ldots, W_n\}$ be a partition of Q, i.e., $\text{int}(Q_i) \cap \text{int}(Q_j) = \emptyset$ and $\sum_{i=1}^{n} W_i = Q$. The objective of coverage

control is to minimize the following cost function:

$$H(P,W)=\sum_{i=1}^{n}\int_{W_i} f(\|q-p_i\|)\phi(q)\mathrm{d}q, \tag{4.39}$$

where sensor i is responsible for measurement over the domain W_i. In (4.39), there are two optimization variables: one is the locations of the sensors and the other is the partition W.

Voronoi partitions and centroid Voronoi partitions

It can be verified that the optimal partition is the Voronoi partition $V(P)=\{V_1,\ldots,V_n\}$, where

$$V_i=\{q\in Q\,|\,\|q-p_i\|\leq\|q-p_j\|,\forall j\neq i\}.$$

When two Voronoi regions are adjacent, we call p_i a neighbor of p_j. Let $H_V(P)=H(P,V(P))$. It follows that

$$H_V(P)=\int_Q \min_{i\in\{1,\ldots,n\}} f(\|q-p_i\|)\phi(q)\mathrm{d}q.$$

Define, respectively, the mass, centroid and polar moment of inertial as

$$\begin{aligned}M_V&=\int_V\phi(q)\mathrm{d}q,\\ C_V&=\frac{1}{M_V}\int_V q\rho(q)\mathrm{d}q,\\ J_{V,p}&=\|q-p\|^2\rho(q)\mathrm{d}q.\end{aligned}$$

It can be shown that

$$H_V(P)=\sum_{i=1}^{n}J_{V_i,C_{V_i}}+\sum_{i=1}^{n}M_{V_i}\|p_i-C_{V_i}\|^2$$

and

$$\frac{\partial H_V}{\partial p_i}(P)=2M_{V_i}(p_i-C_{V_i}),$$

indicating that the local minimum points for H_V are the centroids of their Voronoi cells. Accordingly, we call the critical partition and points

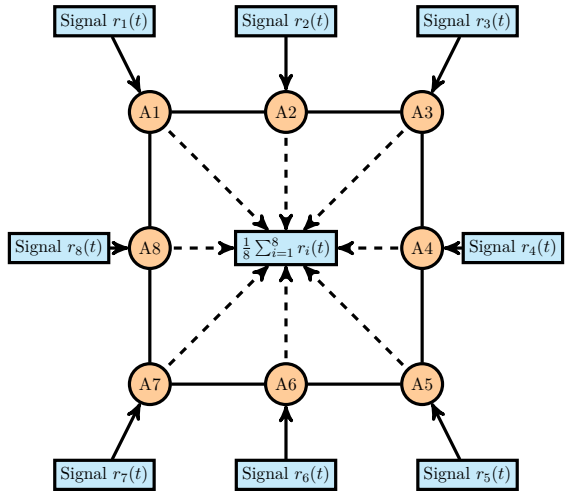


Figure 4.1: Illustration of dynamic average consensus.

for H the centroidal Voronoi partition. Based on the above discussion, a continuous-time coverage control algorithm [129] is proposed as

$$\dot{p}_i = -k(p_i - C_{V_i}) \tag{4.40}$$

where $k > 0$ is a control gain. The coverage control laws based on Voronoi partition are also studied in [130–134].

4.6 Distributed average tracking

In distributed average tracking, each agent i is associated with a reference signal r_i. Here, r_i is generally time-varying. The objective of distributed average tracking is for the agents to reach a consensus on the average of the references (see Fig. 4.1).

Incorporating the derivatives of the references into the consensus algorithm (4.2) yields the following dynamic average consensus algorithm [55] for (4.1)

$$u_i = \sum_{j=1}^{n} a_{ij}(x_j - x_i) + \dot{r}_i, \tag{4.41}$$

where $\sum_{j=1}^{n} a_{ij}(x_j - x_i)$ is a proportional term and $\dot{r}_i$ is a derivative term. For (4.1) using (4.41), if the network topology is undirected, then

$$\sum_{i=1}^{n} \dot{x}_i(t) = \sum_{i=1}^{n} \dot{r}_i(t), \quad \forall t \geq 0.$$

Thus, if the following initialization condition holds

$$\sum_{i=1}^{n} x_i(0) = \sum_{i=1}^{n} r_i(0),$$

then

$$\sum_{i=1}^{n} x_i \equiv \sum_{i=1}^{n} r_i, \tag{4.42}$$

where the triple bar symbol "$\equiv$" implies that the equality holds for all $t \geq 0$. Eq. (4.42) indicates that if the consensus problem is solved, i.e., $x_i(t) = x_j(t)$ for all $i \neq j$, which implies that $x_i(t) = \frac{1}{n}\sum_{j=1}^{n} x_j(t)$, then the dynamic average consensus problem is solved. It has been shown that the algorithm (4.41) can reach dynamic average consensus if the references have steady states. That is, the references will finally converge to certain constant values.

In [56], the Laplace transform $\phi_i(s)$ of the dynamic signals is assumed to have the following form

$$\phi_i(s) = \frac{c_i(s)}{d(s)}, \tag{4.43}$$

where $c_i(s)$ and $d(s)$ are coprime. Here, $d(s)$ can be viewed as the model information of the signals. The state equations of the designed averaging algorithm take the following form

$$\begin{aligned} \dot{X}_i^1 &= A_1 X_i^1 + B_1 \Big[\phi_i - k_p \sum_{j\in\mathcal{N}_i} a_{ij}(v_i - v_j) - k_I \sum_{j\in\mathcal{N}_i} a_{ij}(\eta_i - \eta_j)\Big], \\ v_i &= C_1 X_i^1 + D_1 \Big[\phi_i - k_p \sum_{j\in\mathcal{N}_i} a_{ij}(v_i - v_j) - k_I \sum_{j\in\mathcal{N}_i} a_{ij}(\eta_i - \eta_j)\Big] \end{aligned} \tag{4.44}$$

and

$$\dot{X}_i^2 = A_2 X_i^2 + B_2 \Big[k_I \sum_{j \in \mathcal{N}_i} a_{ij}(v_i - v_j) \Big],$$
$$\eta_i = C_2 X_i^2 + D_2 \Big[k_I \sum_{j \in \mathcal{N}_i} a_{ij}(v_i - v_j) \Big], \tag{4.45}$$

where k_p and k_I are positive gains, v_i agent i's estimate of the average of ϕ_i, X_i^j $(j = 1, 2)$ and η_i the internal states, $\mathcal{N}_i$ the set of neighbors of agent i, A_j, B_j, C_j matrices of appropriate sizes, and D_j a scalar. The parameters A_j, B_j, C_j, D_j, k_p, and k_I are to be designed.

Taking the Laplace transform of (4.44) and (4.45) and solving for $v(s)$ yield that

$$v(s) = P(s)h(s)\phi(s) + P(s)\xi^1(s) - k_I P(s)Lh(s)\xi^2(s), \tag{4.46}$$

where $h(s) = C_1(sI - A_1)^{-1}B_1 + D_1$, $g(s) = C_2(sI - A_2)^{-1}B_2 + D_2$, $P(s) = (I + k_I^2 h(s)g(s)L^2 + k_p h(s)L)^{-1}$, and $\xi^j(s)$ $(j = 1, 2)$ are related to the initial states $X_i^j(0)$. It follows from (4.46) that the following conditions are sufficient to guarantee averaging on the dynamic signals (4.43).

(a) The steady state of $P(s)h(s)\phi(s)$ is the average of $\phi_i(t)$;

(b) $P(s)\xi^1(s)$ and $P(s)Lh(s)\xi^2(s)$ have a steady state of zero.

The above conditions act as a guideline in the design of the parameters of (4.44) and (4.45). It is shown that condition (a) holds if $d(s)|(n_h(s) - d_h(s))$ and $d(s)|d_g(s)$, where $n_f(s)$ and $d_f(s)$ denote the numerator and denominator of f and $a(s)|b(s)$ means that there exists a real polynomial $p(s)$ such that $b(s) = p(s)a(s)$. Therefore, $g(s)$ and $h(s)$ both contain the model information $d(s)$, which suggests that the internal model principle has been applied. Condition (b) is guaranteed if the roots of $d_h(s) = 0$ and $d_g(s)d_h(s) + n_g(s)n_h(s)k_I^2\lambda_i^2 + d_g(s)n_h(s)k_p\lambda_i = 0$ are in the open left half plane. It follows from (b) that the steady-state behavior of (4.44) and (4.45) is not affected by the initial states $X_i^j(0)$, which indicates that the system is robust to initialization errors.

For more general reference signals, a nonsmooth dynamic average consensus algorithm is given in [57] as follows:

$$u_i(t) = \alpha \sum_{j=1}^{n} a_{ij} \mathrm{sgn}(x_j - x_i) + \dot{r}_i \tag{4.47}$$

where $\alpha > 0$ is a control gain, $x_i(0) = r_i(0)$, and $\mathrm{sgn}(\cdot)$ is the signum function defined componentwise. It has been shown that for arbitrary references with bounded derivatives if α is sufficiently large and the graph is undirected and connected, then dynamic average consensus is reached by the proposed algorithm (4.47).

A proportional algorithm and a proportional-integral algorithm are proposed in [135]. It is shown that the more complex proportional-integral algorithm has performance benefits over, the simpler proportional algorithm. A class of discrete-time DAT algorithms is proposed in [136], where the convergence results rely on the input-to-output stability properties of static average consensus algorithms. It is shown in [137] that the proposed DAT algorithm preserves the privacy of the local input of each agent. Ref. [138] addresses the nonlinear DAT problem for networks of dynamic agents. In [139], under an extended proportional-integral control scheme, a new class of DAT algorithms has been developed for three different kinds of references: references with steady states, references with bounded derivatives, and references with a common derivative. Ref. [140] studied the DAT problem for reference signals with bounded accelerations. Ref. [141] proposes a new class of discrete time algorithms that are able to track the average of the signals with an arbitrarily small steady-state error and with robustness to initialization errors. Finally, a connection between dynamic region-following formation control and distributed average tracking is established in [142].

It is noteworthy that most existing works on distributed average tracking focus on undirected graphs or balanced digraphs. Therefore, a future direction is to solve the distributed average tracking problem under a generic digraph. Additionally, since distributed average tracking can serve as estimation algorithms, it would be interesting to investigate the interplay between distributed average tracking and other coordination tasks from which the reference signals are generated.

4.7 Distributed estimation

Distributed parameter estimation

Consider a sensor network which measures an environment parameter θ as follows [143, 144]:

$$y_i = \theta + v_i, \tag{4.48}$$

where $v_i \in N(0, \sigma_i^2)$ is independent random variable with covariance σ_i^2. The estimate with minimum variance is given by

$$\hat{\theta}_{mv} = \frac{\sum_{i=1}^n \frac{1}{\sigma_i^2} y_i}{\sum_{i=1}^n \frac{1}{\sigma_i^2}} = \frac{\frac{1}{n}\sum_{i=1}^n \frac{1}{\sigma_i^2} y_i}{\frac{1}{n}\sum_{i=1}^n \frac{1}{\sigma_i^2}}, \tag{4.49}$$

which is the quotient of two averaged value. Hence, the optimal estimation can be obtained by designing two average consensus algorithms for $\frac{1}{\sigma_i^2} y_i$ and $\frac{1}{\sigma_i^2}$ respectively.

Distributed data regression

Let $f_\theta(x) = \sum_{i=1}^M \theta_i g_i(x)$ be a parameterized function, where $g_i(x)$ are known functions and θ_i are unknown parameters to be determined. Let $\{x_i, y_i\}$ be a data set. The purpose of distributed data regression [145, 146] is to find the optimal value of θ to solve the following problem

$$\min_\theta \sum_{i=1}^n (y_i - f_\theta(x_i))^2. \tag{4.50}$$

Let $G = [g_1, \ldots, g_M]^T$ with $g_i = [g_1(x_i), \ldots, g_M(x_i)]$ and let $y = [y_1, \cdots, y_M]^T$. If $(G^T G)^{-1}$ exists, the optimal θ that minimizes (4.50) is given by

$$\theta^* = \arg\min_\theta \|y - G\theta\|^2 = \left(\frac{1}{n}\sum_{i=1}^n g_i g_i^T\right)^{-1} \left(\frac{1}{n}\sum_{i=1}^n g_i y_i\right), \tag{4.51}$$

whose value can be calculated by two average consensus algorithms.

Distributed Kalman filtering

Consider the following dynamic process [147, 148]

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1},$$

where x_k is the state at time k, u_{k-1} is the input at time $k-1$, and w_{k-1} are independent process noise satisfying

$$p(w) \sim N(0, Q) \tag{4.52}$$

with Q being the covariance matrix. Let

$$z_k = Hx_k + v_k$$

be the measurement. The measurement noise v_k satisfies

$$p(v) \sim N(0, R),$$

where R is the covariance matrix. Let $\hat{x}_k^-$ be the a priori state estimate and $\hat{x}_k$ be the a posteriori state estimate. Define a priori and a posterior estimate errors as

$$e_k^- \triangleq x_k - \hat{x}_k^-, \quad e_k \triangleq x_k - \hat{x}_k.$$

Let

$$P_k^- = E[e_k^-(e_k^-)^T], \quad P_k = E[e_k e_k^T]$$

be, respectively, the a priori and a posterior estimate error covariances. The discrete-time Kalman filter consists the following two steps:

1. Predict:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}, \tag{4.53}$$
$$P_k^- = AP_{k-1}A^T + Q. \tag{4.54}$$

2. Correction:

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-), \tag{4.55}$$
$$K_k = P_k^- H^T(HP_k^- H^T + R)^{-1}, \tag{4.56}$$
$$P_k = (I - K_k H)P_k^-. \tag{4.57}$$

The predict step (4.53) uses the a posterior estimate at the last step to provide a priori estimate at the current step. Eq. (4.54) holds since

$$\begin{aligned} P_k^- &= E[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T] \\ &= E[(A(x_{k-1} - \hat{x}_{k-1}) + w_{k-1})(A(x_{k-1} - \hat{x}_{k-1}) + w_{k-1})^T] \\ &= E(Ae_{k-1}e_{k-1}^T A^T + Ae_{k-1}w_{k-1} + w_{k-1}Ae_{k-1} + w_{k-1}w_{k-1}^T) \\ &= AP_{k-1}A^T + Q, \end{aligned} \tag{4.58}$$

where the last equality uses the fact that e_{k-1} and w_{k-1} are independent. The correction step (4.55) uses the measurement z_k to correct the a priori estimate, where the gain K_k is chosen as in (4.56) such that the estimate error covariance $E(e_k^T e_k)$ is minimized. We show this in the following. According to its definition,

$$\begin{aligned} E(e_k^T e_k) &= E((x_k - \hat{x}_k)^T (x_k - \hat{x}_k)) \\ &= E((x_k - \hat{x}_k^-) - K(z_k - H\hat{x}_k^-))^T ((x_k - \hat{x}_k^-) - K(z_k - H\hat{x}_k^-)) \\ &= E((x_k - \hat{x}_k^-)^T (x_k - \hat{x}_k^-)) + E((x_k - \hat{x}_k^-)^T (-K)(z_k - H\hat{x}_k^-)) \\ &\quad + E((z_k - H\hat{x}_k^-)^T (-K)^T (x_k - \hat{x}_k^-)) \\ &\quad + E((z_k - H\hat{x}_k^-)^T (-K)^T (-K)(z_k - H\hat{x}_k^-)). \end{aligned}$$

Calculating the partial derivative of $E(e_k^T e_k)$ with respect to K and setting it to zero yields

$$(HP_k^- H^T + R)K = P_k^- H^T, \tag{4.59}$$

which is equivalent to (4.56). Eq. (4.57) holds since

$$\begin{aligned} P_k &= E(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T \\ &= (I - K_k H)P_k^- (I - K_k H)^T + KRK^T \\ &= (I - K_k H)P_k^-, \end{aligned} \tag{4.60}$$

where the last equality holds since

$$KRK^T - (I - K_k H)P_k^- (K_k H)^T = 0.$$

Assume that there is a sensor network with n sensors described by the following sensing model [147]:

$$z_i(k) = H_i x(k) + v_i(k).$$

Let $z_c = [z_1^T, \ldots, z_n^T]$, $v_c = [v_1^T, \ldots, v_n^T]$, and $H_c = [H_1; \ldots, H_n]$. We have

$$z_c(k) = H_c x(k) + v_c(k).$$

Let

$$R_c = \text{diag}(R_1, \ldots, R_n) \tag{4.61}$$

denote the covariance of v_c. The state propagation equation is given by

$$\hat{x} = \bar{x} + M(H_c^T R_c^{-1} z_c - H_c^T R_c^{-1} H_c \bar{x}),$$

where

$$M = (P + H_c^T R_c^{-1} H_c)^{-1}.$$

Define the inverse-covariance matrix

$$S = \frac{1}{n} H_c^T R_c^{-1} H_c = \frac{1}{n} \sum_{i=1}^{n} H_i^T R_i^{-1} H_i$$

and the average measurement $y_i = H_i^T R_i^{-1} z_i$ and $y = \frac{1}{n} \sum_{i=1}^{n} y_i$. The Kalman state update equation is given by

$$\hat{x} = \bar{x} + M_\mu (y - S\bar{x}),$$

with a micro-Kalman gain of $M_\mu = nM$. In addition, one has

$$M_\mu = nM = ((nP)^{-1} + S)^{-1}. \tag{4.62}$$

Let $P_\mu = nP$ and $Q_\mu = nQ$, we have

$$P_\mu^+ = AM_\mu A^T + BQ_\mu B^T.$$

It is shown in [147] that if the network is connected, then the distributed Kalman filter

$$\begin{aligned} M_\mu &= (P_\mu^{-1} + S)^{-1}, \\ \hat{x} &= \bar{x} + M_\mu (y - S\bar{x}), \\ P_\mu^+ &= AM_\mu A^T + BQ_\mu B^T, \\ \bar{x}^+ &= A\hat{x} \end{aligned}$$

gives an estimate identical to the one obtained via a central Kalman filter.

Ref. [148] introduces three novel distributed Kalman filtering (DKF) algorithms for sensor networks, discusses communication complexity and packet-loss issues, and demonstrates the performance and effectiveness of these distributed Kalman filtering algorithms on a target tracking task. Ref. [149] proposes a two-stage estimation strategy: the first is a Kalman-like measurement update and the second is an estimate fusion using a consensus matrix. Ref. [150] designs the optimal decentralized Kalman-consensus filter and shows that its computational and communication costs are not scalable in the network size. Ref. [151] focuses on diffusion strategies, where nodes communicate with their direct neighbors only, and the information is diffused across the network through a sequence of Kalman iterations and data-aggregation. Ref. [152] presents the gossip interactive Kalman filter (GIKF) for distributed Kalman filtering for networked systems and sensor networks. Ref. [153] applies a DAT algorithm to distributed Kalman filtering, where the resulting distributed Kalman filter and the embedded DAT estimator update at the same frequency. Ref. [154] gives a frequency-domain characterization of the distributed estimator's steady-state performance. Ref. [155] designs a distributed Kriged Kalman filter for predictive inference of a random field and its gradient.

Future works might include the optimal gain design for distributed Kalman filter, quantized distributed Kalman filter, and the applications of distributed Kalman filter in big data.

4.8 Containment control and surrounding control

4.8.1 Containment control

In order to describe the problem, we employ the notation used in [156]. Let $f : V \mapsto \mathbb{R}^d$ be a function defined over a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The partial derivative of f is defined as

$$\partial_y f(x) = f(y) - f(x).$$

The Laplacian of f is given by

$$\Delta f(x) = \sum_{y\in\mathcal{V},(x,y)\in\mathcal{E}} \partial_y f(x).$$

In containment control, the agents are required to move towards the convex hull formed by a group of leaders (references). Let $\mathcal{V}_L$ denote the set of leaders and $\mathcal{V}_F$ denote the set of followers. Let $r(x,t)$ be the position of agent x at time t. The dynamics of followers is governed by

$$\dot{r}(x,t) = u(x,t), \quad x \in \mathcal{V}_F \tag{4.63}$$

where $u(x,t)$ is the control input. In [156], the control input is designed as

$$u(x,t) = \Delta r(x,t) = \sum_{(y,x)\in\mathcal{E}} (r(y,t) - r(x,t)).$$

Let the leader be fixed, i.e., $r(x,t) = \bar{r}(x)$ for $x \in \mathcal{V}_L$, where $\bar{r}$ is a constant vector. The equilibria of (4.63) are given by

$$\Delta h(x) = 0, \quad x \in \mathcal{V}_F. \tag{4.64}$$

It is shown in [157] that if the graph $\mathcal{G}$ is connected, the partial difference equation (4.64) has a unique solution $h(x)$. Additionally, $\forall x \in \mathcal{V}_F$, $h(x)$ lies in the convex hull of the leaders position.

Sometimes, it is convenient to write the containment control system (4.63) as

$$\dot{x}_i = -\sum_{j\in\mathcal{V}_F} a_{ij}(x_i - x_j) - \sum_{j\in\mathcal{V}_L} a_{ij}(x_i - r_j), \tag{4.65}$$

where x_i is the state of the ith agent, r_j the state of the jth leader, $\mathcal{V}_F$ the set of the followers, and $\mathcal{V}_L$ the set of the leaders. Define $x = [x_1, \ldots, x_{|\mathcal{V}_F|}]^T$ and $r = [r_1, \ldots, r_{|\mathcal{V}_L|}]$, where $|\mathcal{V}_F|$ and $|\mathcal{V}_L|$ denote, respectively, the numbers of the followers and the leaders. Eq. (4.65) can be rewritten in a matrix form as

$$\dot{x} = -L_1 x - L_2 r. \tag{4.66}$$

Here, $L_1 = L + A_0$, where L is the Laplacian matrix of the followers, and $A_0 = \mathrm{diag}([a_{10}, \ldots, a_{|\mathcal{V}_F|0}])$ with $a_{i0} = \sum_{j\in\mathcal{V}_L} a_{ij} > 0$ if agent i can access at least one leader. The matrix L_2 denotes the coupling between the followers and the leaders.

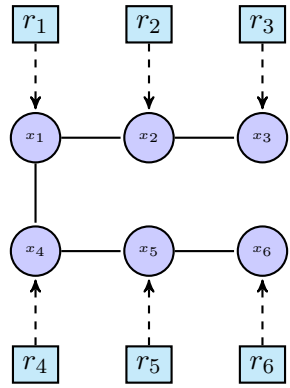


Figure 4.2: A containment control example.

Connections with distributed average tracking

Since containment control and distributed average tracking all involve multiple leaders (references), one might wonder whether these two problems are equivalent. It turns out that the containment control problem has an essential difference with the distributed average tracking problem. To illustrate, let's see a containment control example given by Fig. 4.2. Here, for comparison, we make the number of leaders equal to the number of followers. But in general, the number of leaders can be different from that of followers. In Fig. 4.2, x_i denotes follower i and r_i denotes leader (reference) i. For convenience, we consider constant references, i.e., r_i is not time-varying. In this example, the matrices defined in (4.66) are given by $L_1 = L + I$ with

$$L \triangleq \begin{bmatrix} 2 & -1 & 0 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix},$$

and $L_2 = -I$. In (4.66), x will converge to $-L_1^{-1}L_2 r$, where $-L_1^{-1}L_2$ is a row-stochastic matrix. Suppose that $-L_1^{-1}L_2 = \frac{1}{N}\mathbf{1}\mathbf{1}^T$. Using $L_2 = -I$, it follows that $L_1^{-1} = \frac{1}{N}\mathbf{1}\mathbf{1}^T$, which is not possible because $\frac{1}{N}\mathbf{1}\mathbf{1}^T$ is not invertible. It thus follows that $-L_1^{-1}L_2 \neq \frac{1}{N}\mathbf{1}\mathbf{1}^T$, which indicates that

x_i will converge to the convex hull formed by the leaders but not the average of the leaders. Thus, there exists essential difference between containment control and distributed average tracking.

It turns out that some containment control problems can be transformed to coordinated tracking problems. In containment control, the tracking error is defined as $e = x + L_1^{-1}L_2 r$. Note that L_1 is invertible. Thus, the derivative of e is given by

$$\begin{aligned} \dot{e} &= -L_1 x - L_2 r \\ &= -L_1(x + L_1^{-1}L_2 r) \\ &= -(L + A_0)e, \end{aligned} \tag{4.67}$$

which has the same form to that of coordinated tracking (4.15). Thus, we can conclude that for constant references the closed-loop system (4.65) for containment control can be reduced to the closed-loop system for coordinated tracking.

In [156], the authors exploit the theory of partial difference equations and propose hybrid control schemes based on stop-go rules for the leader-agents. Non-Zenoness, liveness, and convergence of the resulting system are also analyzed. In [158], a leader based containment control strategy for multiple unicycle agents is introduced. It is shown that the followers converge to the same velocities and orientations as the leaders, with the same control law that is used for the followers in the initial containment control problem. Ref. [159] presents a distributed containment control approach for uncertain nonlinear strict-feedback systems with multiple dynamic leaders under a directed graph topology. The containment control problem of heterogeneous linear multi-agent systems based on output regulation framework is also considered in [160].

4.8.2 Surrounding control

Suppose a team of unmanned ground vehicles (UGVs) is sent to detect and establish a corridor through hostile terrain. To protect the UGVs from potential threats, another group of armed robotic vehicles (ARVs) is dispatched to provide ground cover for the UGVs. That is, ARVs must surround the UGVs. The problem is referred to as a surrounding

control problem. The surrounding control problem can be considered as the inverse problem of containment control.

When the geometric center of the leader $\bar{x} \triangleq \frac{1}{n}\sum_{j\in V_L} x_j$ is available to all the followers, a surrounding control algorithm is given in [161] as

$$u_i = \kappa_1 \sum_{j\in N_i^s} (x_i - x_j) + \kappa_2 \left(\bar{x} + \xi \mathrm{sgn}\left\{ \sum_{j\in N_i^s(0)} [x_i(0) - x_j(0)] \right\} - x_i \right) \tag{4.68}$$

where κ_1, κ_2, and ξ are positive constants, and $\mathrm{sgn}(\cdot)$ is the signum function. In the control input, the first term is an repulsive force between follower i and its neighbors. This term is used to enlarge the convex hull $co(V_F)$ formed by the followers. The second term is an attractive force, which is used to drive the followers to $\bar{x}+\xi\mathrm{sgn}\left\{\sum_{j\in N_i^s(0)}[x_i(0) - x_j(0)]\right\}$. It is shown that the algorithm (4.68) has the ability of collision avoidance, i.e., if $x_i(0) > x_j(0)$, $i, j \in V_F$, then $x_i(t) > x_j(t)$ for all $t \geq 0$. The results have been extended to the case of balanced surrounding control where the additional constraint that the followers form a regular polytope with a geometric center at $\bar{x}$ is imposed.

The distributed surrounding control problem of a convex target is addressed in [162]. Consider an n-agent system described by the first-integrator dynamics:

$$\dot{x}_i = u_i, \quad i = 1, \ldots, n, \tag{4.69}$$

where x_i and u_i are the state and input of agent i. Let X be a convex set to be surrounded, which is bounded and closed. A complex-valued adjacency matrix $A = [a_{ij}] \in \mathbb{C}^{n\times n}$ is used to describe the desired relative angles of projections for agents to X, which is designed as follows: $a_{ii} = 0$ and either $|a_{ij}| = 1$ or $a_{ij} = 0$ for $i \neq j$. We say distributed set surrounding is achieved for (4.69) if

$$\lim_{t\to\infty} a_{ij}(x_j - P_X(x_j)) - (x_i - P_X(x_i)) = 0, \quad \forall (i,j) \in \mathcal{E}.$$

The control input u_i is designed as

$$u_i = \sum_{j\in\mathcal{N}_i} \left(a_{ij}(x_j - P_X(x_j)) - (x_i - P_X(x_i))\right). \tag{4.70}$$

It is shown that if the directed cycles of the graph are consistent [162], and the graph is strongly connected, the distributed set surrounding problem is solved.

A distributed controller is proposed in [162] to surround a given set with the same distance and desired projection angles specified by a complex-value adjacency matrix. In [163], the surrounding control problem of second-order multi-agent systems is studied, where the targets can be either stationary or moving. Ref. [164] investigates the surrounding control problem of a group of non-identical agents, where an adaptive design method is presented. A decentralized control algorithm is proposed in [165] for a collection of nonholonomic vehicles to achieve collective circular motion behavior. A repulsion mechanism is employed to improve the distribution evenness of the agents'circular motion phases and hence to avoid collision.

4.9 Distributed optimization

4.9.1 Unconstrained distributed optimization

A distributed optimization problem takes the following form:

$$\min_{x\in\mathbb{R}^m} \sum_{i=1}^{n} f_i(x), \tag{4.71}$$

where $f_i : \mathbb{R}^m \mapsto \mathbb{R}$ is the objective function of agent i. Let f^* denote the optimal value of the objective function, which is finite. The optimal solution set is then given by

$$X^* = \left\{ x \in \mathbb{R}^n : \sum_{i=1}^{n} f_i(x) = f^* \right\}. \tag{4.72}$$

Let $x_i \in \mathbb{R}^m$ denote the estimate of agent i's estimate of the optimal solution of (4.71). It turns out that the optimal solution of the unconstrained problem (4.71) is equivalent to the following constrained optimization problem:

$$\begin{aligned} &\min_{x_i\in\mathbb{R}^m} \sum_{i=1}^{n} f_i(x_i) \\ &\text{s.t. } x_i = x_j, \quad i \neq j. \end{aligned} \tag{4.73}$$

A subgradient algorithm is proposed in [58] as

$$x_i(k+1) = \sum_{j=1}^{n} a_{ij}(k)x_j(k) - \alpha d_i(k), \tag{4.74}$$

where a_{ij} is the (i,j)th entry of the adjacency matrix that describes the interactions between the agents, α is the stepsize for all agents, and $d_i(k)$ is a subgradient of f_i at $x_i(k)$, which satisfies

$$f_i[x_i(k)] + [d_i(k)]^T[x - x_i(k)] \leq f_i(x),\ \forall x.$$

It has been shown that under the subgradient algorithm (4.74), the optimization error is upper bounded and can be tuned by the stepsize α. That is, a smaller stepsize guarantees a smaller error.

Remark 1. The problem (4.71) can be viewed as a cooperative game, where each agent is willing to share its decision, i.e., x_i, with its neighbors.

4.9.2 Constrained distributed optimization

In this section, the agents aim at solving the constrained optimization problem:

$$\min_x \sum_{i=1}^{n} f_i(x)$$
$$\text{subject to } x \in \bigcap_{i=1}^{n} X_i,$$

where X_i represents the constrained set of each agent. In [59], the authors propose the following projected subgradient algorithm

$$x_i(k+1) = P_{X_i}\left[\sum_{j=1}^{m} a_{ij}(k)x_j(k) - \alpha_k d_i(k)\right], \tag{4.75}$$

where $\alpha_k > 0$ and $P_{X_i}(\bar{x})$ is the projection of the vector x on the set X_i, i.e.,

$$P_{X_i}(\bar{x}) = \arg\min_{x_i \in X_i} \|x_i - \bar{x}\|.$$

Due to the nonlinear projection operator, the analysis of (4.75) is more complicated than that of (4.74). It has been shown that when the constraint sets X_i are identical, i.e., $X_i = X$ for all i, the algorithm guarantees that $x_i(k) \to x^*$ as $k \to \infty$, where x^* is an optimal point.

4.9.3 Time-varying cost function

Consider a multi-agent system with single-integrator dynamics:

$$\dot{x}_i(t) = u_i(t), \quad i = 1, \ldots, n, \tag{4.76}$$

where x_i and u_i denote, respectively, the position and control input of agent i. Suppose that each agent has a time-varying cost function $f_i(x_i, t)$, which is available only to agent i. The objective is to optimize the total cost function:

$$\min_{x_i} \sum_{i=1}^{n} f_i(x_i, t)$$
$$\text{subject to } x_i = x_j, \ \forall i \neq j$$

in a cooperative and distributed way. Here, the functions f_i are assumed to be twice continuously differentiable and have invertible Hessian. For the single integrator dynamics (4.76), the control input is given by

$$\begin{aligned} u_i &= -\sum_{j \in N_i} \beta_{ij} \text{sgn}(x_i - x_j) + \phi_i, \\ \dot{\beta}_{ij} &= \|x_i - x_j\|_1, \quad j \in N_i, \\ \phi_i &= -H_i^{-1}(x_i, t) \left[\nabla f_i(x_i, t) + \frac{\partial}{\partial t} \nabla f_i(x_i, t) \right], \end{aligned} \tag{4.77}$$

where ϕ_i is an internal signal, β_{ij} is a time-varying gain satisfying $\beta_{ij} = \beta_{ji} > 0$, $\text{sgn}(\cdot)$ is the signum function defined componentwise, and H_i is the Hessian matrix of f_i. Note that ϕ_i depends on only agent i's position. It has been shown that if $H_i(x_i, t) = H_j(x_j, t)$, then the time-varying optimization problem for single-integrator systems can be solved by the nonsmooth algorithm (4.77). For double-integrator systems

$$\begin{aligned} \dot{x}_i(t) &= v_i(t), \\ \dot{v}_i(t) &= u_i(t), \end{aligned}$$

the control input is proposed in [166] as

$$\begin{aligned} u_i = &- \sum_{j \in N_i} [\mu(x_i - x_j) + \alpha(v_i - v_j)] \\ &- \sum_{j \in N_i} \beta_{ij} \text{sgn}[\gamma(x_i - x_j) + \psi(v_i - v_j)] + \phi_i, \\ \dot{\beta}_{ij} = &\|\gamma(x_i - x_j) + \psi(v_i - v_j)\|_1, j \in N_i, \end{aligned} \tag{4.78}$$

where μ, α, γ and ψ are positive constants and

$$\begin{aligned} \phi_i \triangleq &-H_i^{-1}(x_i,t)\left[\frac{\partial}{\partial t}\frac{d}{dt}\nabla f_i(x_i,t) + \frac{d}{dt}\nabla f_i(x_i,t)\right] - H_i(x_i,t)\nabla f_i(x_i,t) \\ &+ \left\{H_i^{-1}(x_i,t)\left[\frac{d}{dt}H_i(x_i,t)\right]H_i^{-1}(x_i,t)\right\}\left[\frac{\partial}{\partial t}\nabla f_i(x_i,t) + \nabla f_i(x_i,t)\right] \end{aligned} \tag{4.79}$$

Ref. [167] devises two distributed primal-dual subgradient algorithms based on the characterization of the primal-dual optimal solutions as the saddle points of the Lagrangian and penalty functions. Contrary to what is known in the consensus literature, it is shown in [168] that the consensus-based dynamics that solves the continuous-time distributed optimization problem for undirected graphs fails to converge when transcribed to the directed setting. Ref. [169] incorporates the presence of a random communication graph between the agents as a more realistic abstraction of the gossip and broadcast communication protocols of a wireless network. In [170], the authors develop and analyze distributed algorithms based on dual subgradient averaging, and provide sharp bounds on their convergence rates as a function of the network size and topology.

5

Research issues

In this chapter, we discuss several research issues arising from the design and analysis of MASs. Specifically, we will consider communication time-delays, quantization, packet loss, noise and/or disturbance, connectivity maintenance, time-triggered control, as well as event-triggered control. It should be pointed out that the chosen topics in this section are by no means complete.

5.1 Network issues

5.1.1 Time-delays

Time-delays arise when information is exchanged between neighboring agents via a communication network. The effect of communication delays is often neglected for the ease of theoretical analysis. However, it is well known that a small communication delay can even destroy the stability of a system [171, 172].

Uniform time-delays

Assume that the network topology is undirected and connected, and the communication delays in all channels all equal to τ. The consensus

system (4.3) becomes

$$\dot{x}_i(t) = \sum_{j=1}^{n} a_{ij}[x_j(t-\tau) - x_i(t-\tau)]. \tag{5.1}$$

Taking the Laplace transform on both sides of (5.1) yields

$$sX_i(s) - x_i(0) = \sum_{j \in N_i} a_{ij} e^{-\tau s}[X_j(s) - X_i(s)], \tag{5.2}$$

where $X_i(s)$ denotes the Laplace transform of $x_i(t)$. Eq. (5.2) can be rewritten in a compact way as

$$X(s) = (sI + e^{-\tau s}L)^{-1}x(0), \tag{5.3}$$

where L is the Laplacian matrix. The convergence analysis of the time-delayed system (5.1) is hence reduced to the stability analysis of the MIMO system (5.3). It has been proved in [49] that consensus can be reached if the communication delay $\tau \in [0, \frac{\pi}{2\lambda_{\max}(L)})$, where $\lambda_{\max}(L)$ denotes the maximal eigenvalue of the Laplacian matrix L. One can conclude that the upper bound on the time-delay is proportional to $\lambda_{\max}(L)$. By the Gershgorin theorem, we have $\lambda_{\max} \leq 2d_{\max}(G)$, where $d_{\max}(G)$ denotes the maximal degree of the nodes in G. Hence, an alternative upper bound is given by

$$\tau \leq \frac{\pi}{4d_{\max}(G)}. \tag{5.4}$$

This indicates that the larger the maximum degree, the less robust of the MAS to the time-delay.

Non-uniform time-delays

Consensus with non-uniform delays and a switching directed network topology is investigated in [173] for the following system

$$\dot{x}_i = -\sum_{j=1}^{n} a_{ij}^{\sigma(t)} \kappa_{ij}[x_i - \mathbb{T}_{ij}(x_j)], \tag{5.5}$$

where x_i is the state of agent i and $a_{ij}^{\sigma(t)}$ the (i,j)th entry of the adjacency matrix with $\sigma(t)$ the switching signal of the network topology. The

coupling functions $\kappa_{ij}(z)$ are defined as $\kappa_{ij}(z) = \overline{\kappa_{ij}}(\|z\|)(z/\|z\|)$ with $\overline{\kappa_{ij}}(\|z\|)$ being nonlinear, continuous gains satisfying $\kappa_{ij}(0) = \overline{\kappa_{ij}}(0) = 0$ and the following sector condition:

$$\bar{\mathcal{K}} z \leq \overline{\kappa_{ij}}(z) \leq \underline{\mathcal{K}} z, \quad \forall z \geq 0, \tag{5.6}$$

with $\bar{\mathcal{K}}$ and $\underline{\mathcal{K}}$ positive constants. The symbol $\mathbb{T}_{ij}$ is a delay operator, which can represent the three most common delay models: constant delays $\mathbb{T}_{ij}(x_j) = x_j(t-\tau_{ij})$ for some $\tau_{ij} \in [0, \mathcal{T}]$ with $\mathcal{T}$ a positive constant, time-varying delays $\mathbb{T}_{ij}(x_j) = x_j(t - \tau_{ij}(t))$ with $\tau_{ij} : \mathbb{R} \mapsto [0, \mathcal{T}]$, and distributed delays $\mathbb{T}_{ij}(x_j) = \int_0^{\mathcal{T}} \phi_{ij}(\eta) x_j(t-\eta) d\eta$ with the delay kernel ϕ_{ij} satisfying $\phi_{ij}(\eta) \geq 0$ for all $\eta \in [0, \mathcal{T}]$ and $\int_0^{\mathcal{T}} \phi_{ij}(\eta) d\eta = 1$. It is shown that a consensus is reached for arbitrary bounded heterogeneous constant, time-varying, or distributed delays if the network topology has a jointly directed spanning tree (cf. Section 2.4).

A nonlinear consensus system with non-uniform delays is also given in [174] as follows:

$$\dot{x}_i(t) = k_i \sum_{j=1}^{n} a_{ij} f_{ij}[x_j(t - \tau_{ij}) - x_i(t)], \tag{5.7}$$

where τ_{ij} denotes the constant time-delay from agent j to agent i, k_i are constants, and f_{ij} are locally passive functions on $[-\sigma_{ij}^-, \sigma_{ij}^+]$ for some $\sigma_{ij}^- > 0$ and $\sigma_{ij}^+ > 0$. For the time-delayed system (5.7), the state space is infinite-dimensional and is given by the Banach space $\mathcal{C}([-\tau, 0], \mathbb{R}^n)$, where $\tau = \max_{i,j} \tau_{ij}$. Define $\gamma = \min_{i,j=1,\ldots,n}\{\sigma_{ij}^-, \sigma_{ij}^+\}$. It is shown that if the initial states of the agents are less than $\frac{\gamma}{2}$, consensus can be reached regardless of the size of the communication delays [174] if the network topology has a directed spanning tree.

It is noteworthy that time-delayed MASs are infinite-dimensional systems, i.e., the state of a time-delayed MAS is a function defined in an time interval, rather than a vector. It has been shown in [175] that the initial states of MASs have a significant effect on consensus equilibrium.

Ref. [176] presents a methodology for the stability analysis of linear consensus protocols with time-delayed communications, where uniform delays are considered. Ref. [177] focuses on integrator dynamics with time-delays: both constant and time-varying delays are considered, as

well as the uniform and non-uniform repartitions of the delays. Ref. [178] investigates the robustness of consensus schemes for linear MASs to feedback delays and develops a unified framework that considers linear MAS models with different feedback delays.

5.1.2 Quantization

For a real digital network, communication channels always have a finite communication capacity. That is, agents can only transmit a limited amount of information at each time step. This indicates that the agents cannot use the exact state information in their control inputs. The above observation motivates the study of quantization effects in consensus.

In what follows, two kinds of quantizers are presented. The first one is the so-called *uniform quantizer*, which is described as follows. For $L_q \in \mathbb{N}$ where $\mathbb{N}$ denotes the set of all natural numbers, define the uniform set of quantization levels [179]

$$S_{L_q} = \left\{ -1 + \frac{2l-1}{L_q} : l \in \{1, \ldots, L_q\} \right\} \cup \{-1\} \cup \{1\}. \tag{5.8}$$

The parameter L_q determines the number of quantization levels m in the following way: $m = L_q + 2$. The uniform quantizer is then defined as

$$\mathrm{unq}_{L_q}(x) = \begin{cases} -1 + \frac{2l-1}{L_q}, & \text{if } -1 + \frac{2(l-1)}{L_q} \le x \le -1 + \frac{2l-1}{L_q} \\ 1, & \text{if } x > 1 \\ -1, & \text{if } x < -1 \end{cases}$$

where a larger value of L_q corresponds to a more accurate uniform quantizer. Another quantizer that is commonly used is the so-called *logarithmic quantizer*. Given an accuracy parameter $\delta \in (0, 1)$, the logarithmic set of quantization levels is defined by

$$S_\delta = \left\{ \left(\frac{1+\delta}{1-\delta} \right)^l \right\}_{l \in \mathbb{Z}} \cup \{0\} \cup \left\{ -\left(\frac{1+\delta}{1-\delta} \right)^l \right\}_{l \in \mathbb{Z}}. \tag{5.9}$$

The corresponding logarithmic quantizer [180] is given by

$$\mathrm{lgq}_\delta(x) = \begin{cases} (\frac{1+\delta}{1-\delta})^l, & \text{if } \frac{(1+\delta)^{l-1}}{(1-\delta)^l} \le x \le \frac{(1+\delta)^l}{(1-\delta)^{l+1}} \\ 0, & \text{if } x = 0 \\ -\mathrm{lgq}_\delta(-x), & \text{if } x < 0 \end{cases} \tag{5.10}$$

where a smaller value of the parameter δ corresponds to a more accurate logarithmic quantizer.

In [181], a quantized average consensus problem is considered. The objective is to design distributed algorithms such that the value of each agent will finally converge to an integer approximation of the average of the initial states subject to the following constraints:

1. The state of each node is always an integer.

2. The sum of states in the network does not change with time.

Let $x(t) = [x_1(t), \ldots, x_n(t)]$ denote the state of the multi-agent system, where $x_i(t)$ is the state of agent i. A class of quantized gossip algorithms is proposed in [181] as follows. For a pair (i, j), if $|x_i(t) - x_j(t)| = 0$, then the states are left unchanged, namely, $x_k(t+1) = x_k(t)$ for $k = i, j$; if $|x_i(t) - x_j(t)| \geq 1$, it is required that

1. $x_i(t+1) + x_j(t+1) = x_i(t) + x_j(t)$,

2. if $|x_i(t) - x_j(t)| > 1$, then $|x_i(t+1) - x_j(t+1)| < |x_i(t) - x_j(t)|$, and

3. if $|x_i(t) - x_j(t)| = 1$ and (without loss of generality) $x_i(t) < x_j(t)$, then $x_i(t+1) = x_j(t)$ and $x_j(t+1) = x_i$.

It is proved that algorithms satisfying 1–3 can solve the quantized average consensus problem.

Note that in the above quantized average consensus problem, the agents converge to an integer approximation of the average of the initial states, instead of the exact average. What if we want to drive the agents to the exact average while considering quantization effects? It turns out that a dynamic coder or decoder scheme can be introduced to solve the average consensus problem under quantized communication. The coder is used to transform the state to the message to be transmitted to neighboring agents, while the decoder is used to transform the message received from neighboring agents to the state (estimated). For the logarithmic quantizer, the following coder/decoder scheme is studied

in [182]:

$$\begin{aligned} \xi(t+1) &= \xi(t) + \alpha(t), \\ \alpha(t) &= \mathrm{lgq}_\delta(x(t) - \xi(t)), \\ \hat{x}(t) &= \xi(t) + \alpha(t), \end{aligned} \tag{5.11}$$

where $\xi(t)$ is the coder/decoder state, $\alpha(t)$ is the coder, and $\hat{x}(t)$ is the decoder. It is shown that if the logarithmic quantizer is accurate enough, i.e., δ is small enough, then average consensus can be reached under quantized communication.

Ref. [181] studies the distributed averaging problem with the additional constraint that the value at each node is an integer, and derives bounds on the convergence time of these algorithms for fully connected networks and linear networks. It is shown in [183] that a quantized consensus can be reached for an arbitrary quantizer utilizing the proposed stochastic gossip algorithm. Ref. [184] derives a necessary and sufficient graphical condition to guarantee consensus subject to quantized information flow. The obtained graphical condition ensuring average consensus is weaker than those in the literature for either real-valued or quantized states. Ref. [185] studies the problem of distributed average consensus with quantized data and random link failures, where dither (small noise) is added to the sensor states before quantization. Ref. [186] presents an algorithm to solve an extended version of the quantized consensus problem over networks represented by Hamiltonian graphs. Ref. [187] considers continuous-time average consensus via uniform quantizers. Solutions to the resulting system are defined in the Krasowskii sense and are proven to converge to conditions of "practical consensus".

5.1.3 Packet loss

Consider a wireless sensor network with $N > 1$ nodes. At each time instant t, each agent takes a noisy measurement of a scalar signal $d(t)$, i.e.,

$$u(t) = d(t)\mathbf{1} + v(t),$$

where $u = (u_1, \ldots, u_n)^T$ is the sensor measurement vector, and $v = (v_1, \ldots, v_N)^T$ is the noise vector. It is assumed that $v(t) \in \mathcal{N}(0, \sigma^2 I)$.

The information flows among the sensors are described by an weighted undirected graph $\mathcal{G}(t) = (\mathcal{V}, \mathcal{E}(t))$, where $\mathcal{V} = \{1, \ldots, N\}$ is the node set, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set. It is noteworthy that even though the information flows are bidirectional, the weights assigning to the edges (i, j) and (j, i) can be different.

Each sensor computes its estimate $x_i(t)$ of $d(t)$ as follows:

$$x_i(t) = \sum_{j \in \mathcal{N}_i(t)} k_{ij}(t)\phi_{ij}(t)x_j(t-1) + \sum_{j \in \mathcal{N}_i(t)} h_{ij}(t)\phi_{ij}(t)u_i(t), \quad (5.12)$$

where $k_{ij}(t)$ and $h_{ij}(t)$ are weights assigned to the edge (j, i), and $\phi_{ij}(t)$ are binary random variables modelling package losses of the link. Specifically, $\Pr(\phi_{ij}(t) = 1) = p$ and $\Pr(\phi_{ij}(t) = 0) = 1 - p$, where $p \in [0, 1]$. One can write (5.12) in a matrix form as

$$\begin{aligned} x(t) &= (K(t) \circ \Phi(t))x(t-1) + (H(t) \circ \Phi(t))u(t) \\ &= K_{\phi(t)}(t)x(t-1) + H_{\phi(t)}u(t), \end{aligned}$$

where $x = (x_1, \ldots, x_N)^T$,

$$[K(t)]_{ij} = \begin{cases} k_{ij}(t), & \text{if} j \in \mathcal{N}_i(t); \\ 0, & \text{otherwise}, \end{cases}$$

and $\circ$ denotes the Hadamard (element-wise) product between two matrices. We define the matrices $H(t)$ and $\Phi(t)$ in a similar way.

Define the estimation error

$$e(t) = x(t) - d(t)\mathbf{1}.$$

Additionally, define $\delta(t) = d(t) - d(t-1)$. The error dynamics is then given by

$$\begin{aligned} e(t) = K_{\phi(t)}(t)e(t-1) + d(t)(K_{\phi(t)}(t) + H_{\phi(t)} - I)\mathbf{1} - \delta(t)K_{\phi(t)}(t)\mathbf{1} \\ + H_{\phi(t)}(t)v(t). \end{aligned} \quad (5.13)$$

A stability result for the system (5.12) is given as follows.

Theorem 3 ([188]). Consider the system (5.12). If

1. $((K(t) + H(t) - I) \circ \bar{\Phi}(t))\mathbf{1} = 0$ for any realization $\bar{\Phi}(t)$ of the stochastic process $\Phi(t)$;

2. $|\delta(t)| < \Delta$ for all t;

3. $\gamma(K(t)) \leq \gamma_{\max} M < 1$ for all t.

Then,

$$\lim_{t\to\infty} \|\mathbb{E}_\phi \mathbb{E}_v e(t)\| \leq \frac{\Delta\sqrt{N}\gamma_{\max}}{1-\gamma_{\max}}.$$

5.2 Noise or disturbance

5.2.1 Random noise

Additive noise

Let $w_{ij}(k)$ denote the communication noise between agents i and j at time k. The measurement by agent i of agent j's state is given by

$$y_{ij}(k) = x_j(k) + w_{ij}(k),$$

where $x_j(k)$ is the real state of agent j. In [189], the authors propose the stochastic consensus algorithm

$$x_i(k+1) = [1 - a(t)b_{ii}]x_i(t) + a(t)\sum_{j\in\mathcal{N}_i} b_{ij}y_{ij}(k), \quad k \geq 0, \tag{5.14}$$

where $a(t) > 0$ is a step size and b_{ij} is defined as follows:

$$\begin{cases} b_{ij} > 0, & \text{if } k \in \mathcal{N}_i, \\ b_{ij} = 0, & \text{if } k \notin \mathcal{N}_i \cup \{i\}, \\ b_{ii} = \sum_{k\in\mathcal{N}_i} b_{ik}, & \text{otherwise.} \end{cases}$$

Two different definitions for stochastic consensus are introduced in [189].

Definition 7 (Mean square consensus)**.** The agents are said to reach mean square consensus if for all i, $E|x_i(k)|^2 < \infty$ and there exists a random variable x^* such that $\lim_{k\to\infty} E|x_i(k) - x^*|^2 = 0$.

Definition 8 (Almost sure consensus)**.** The agents are said to reach almost sure consensus if there exists a random variable x^* such that $\lim_{k\to\infty} x_i(k) = x^*$ almost surely.

It turns out that the design of the step size $a(k)$ plays a central role in guaranteeing the convergence of the algorithm. A design criterion of $a(k)$ is: i) $a(k) > 0$; and ii) $\sum_{k=0}^{\infty} a(k) = \infty$ and $\sum_{k=0}^{\infty} a^2(k) < \infty$. It has been proved that if the design principle is met, then both mean square consensus and almost sure consensus are achieved asymptotically.

Multiplicative noise

In real applications, simple observation is that as the distance between two agents increases, the measurement noise becomes larger [190]. However, this effect cannot be modeled by additive noise. This motivates the study of multiplicative noise. Consider a multi-agent system with single-integrator dynamics:

$$\dot{x}_i(t) = u_i(t), \quad i = 1, \ldots, n. \tag{5.15}$$

Let the measurement of the state of agent j received by agent i be modelled by

$$y_{ij}(t) = x_j(t) + \sigma_{ji}|x_j(t) - x_i(t)|\xi_{ji}(t), \tag{5.16}$$

where $\xi_{ji}(t)$ are standard white noise, and $\sigma_{ji} \geq 0$ is the noise strength. Note that $y_{ii}(t) = x_i(t)$, which implies that agent i can measure its own state exactly. For the system (5.15), the control input with multiplicative noise is given by

$$u_i(t) = \alpha \sum_{j=1}^{n} a_{ij}[y_{ji}(t) - x_i(t)],$$

where $\alpha > 0$ is the consensus gain. The closed-loop system is given by

$$\dot{x}_i(t) = \alpha \sum_{j=1}^{n} a_{ji}[x_j(t) - x_i(t)] + \alpha \sum_{j=1}^{n} a_{ji}\sigma_{ji}|x_j(t) - x_i(t)|\xi_{ji}(t), \quad i = 1, \ldots, n.$$

It has been shown in [190] that if the consensus gain is small enough, then both mean-square consensus and almost sure consensus can be achieved asymptotically.

5.2.2 Bounded disturbance

Consider the first-order system

$$\dot{x}_i = u_i(x_i, y^{(i)}),$$

where $y^{(i)}$ is a vector containing the information of neighboring agents. The jth entry of $y^{(i)}$ is defined as

$$y_j^{(i)} = \begin{cases} y_{ij} & \text{if } j \in N_i, \\ 0 & \text{otherwise.} \end{cases} \tag{5.17}$$

Here, y_{ij} is a measure of x_j obtained by agent i as

$$y_{ij} = x_j + d_{ij},$$

with d_{ij} being a uniformly bounded disturbance, i.e., $-\xi \leq d_{ij} \leq \xi$. The disturbance d_{ij} is unknown; however, the bound ξ is known a priori. In addition, it is assumed that all the agents have perfect measurement of itself., i.e, $d_{ii} = 0$. The control input is given by

$$u_i(x_i, y^{(i)}) = \sum_{j=1}^{n} a_{ij}\phi(y_{ij} - x_i), \tag{5.18}$$

where $\phi : \mathbb{R} \mapsto \mathbb{R}$ is a generic nonlinear scalar function. Because of the presence of the unknown disturbance, the exact consensus is hard to be achieved by linear consensus algorithms. This indicates that for linear consensus algorithms, we can only guarantee that the consensus error is upper bounded. How large is the bound depends on the disturbance upper bound ξ.

In [191], the authors propose the following linear consensus algorithm

$$u_i(x_i, y^{(i)}) = \sum_{j=1}^{n} a_{ij}[\tilde{y}_{ij} - x_i],$$

where $\tilde{y}_{ij}$ is the estimate of state x_j by agent i. Apparently, the estimate $\tilde{y}_{ij}$ must belong to the interval

$$y_{ij} - \xi \leq \tilde{y}_{ij} \leq y_{ij} + \xi.$$

The question is how to select $\tilde{y}_{ij}$ from the above interval. Let $\tilde{y}^{(i)} = \sum_{j=1}^{n} a_{ij}\tilde{y}_{ij}$. The following lazy rule for estimating $\tilde{y}^{(i)}$ is proposed in

[191]:

$$\tilde{y}^{(i)} = \arg\min_{\tilde{y_{ij}}} \left| \sum_{j=1}^{n} a_{ij}(\tilde{y}_{ij} - x_i) \right|. \tag{5.19}$$

Eq. (5.19) is called lazy, since each agent makes the estimate to minimize its control effort $|\dot{x}_i|$.

Ref. [192] considers general state update systems susceptible to perturbations approached from a consensus perspective and derive consensus conditions on the system parameters such as nonstationary, random update and control matrices, and random perturbation vector. Ref. [193] proposes a distributed algorithm to solve the finite time consensus problem in a network of integrators affected by bounded disturbances with a directed communication topology. Using the ordinary differential equation method, it is proved in [194] that the updates converge almost surely to the consensus average for various models of perturbation of data exchanged between nodes. In [195], the authors investigate the properties of a distributed consensus algorithm for a network of continuous-time integrators subject to unknown-but-bounded time-varying disturbances.

5.3 Connectivity maintenance

5.3.1 Connectivity measurements

Let $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ denote the adjacency matrix of a graph $\mathcal{G}$, where $a_{ij} > 0$ if there exists a connection between agent i and agent j and $a_{ij} = 0$ otherwise. The following result characterizes the connectivity of the graph $\mathcal{G}$ by the adjacency matrix.

Theorem 4 ([196]). The graph $\mathcal{G}$ is connected iff all the entries of the matrix A^{n-1} are positive.

Alternatively, graph connectivity can be captured by the Laplacian matrix L. The Laplacian matrix of an undirected graph is positive semi-definite. Without loss of generality, let $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ denote the n eigenvalues of the Laplacian matrix. We have the following result.

Theorem 5 ([87]). If the graph $\mathcal{G}$ is undirected, then it is connected iff $\lambda_2 > 0$.

Theorems 4 and 5 can be proved with the aid of a blend of matrix theory and graph theory.

5.3.2 Connectivity maintenance via Laplacian matrix

Consider a multi-agent system with the edge set determined by the relative positions between the agents. Assign each edge (i, j) with the weight

$$w_{ij} \triangleq f(\|x_i - x_j\|),$$

for some function $f : \mathbb{R}^+ \mapsto \mathbb{R}^+$ with x_i denoting the state of agent i. Under the framework, the graph $\mathcal{G}$ is state-dependent and the Laplacian matrix is also state-dependent. The connectivity maintenance problem can be solved by considering the following optimization problem

$$\text{maximize } \lambda_2(L(x)) \tag{5.20}$$

where x is the stack vector of all the agents' states (Note that the graph is connected iff $\lambda_2(L) > 0$). To prevent the agents from getting arbitrarily close to each other, the following constraint is needed:

$$\|x_i - x_j\|^2 \geq \rho, \tag{5.21}$$

where ρ is a positive constant. The following result is key in transforming the problem (5.20) into semidefinite programming.

Theorem 6 ([197]). For a Laplacian matrix L,

$$\lambda_2(L) > 0$$

is equivalent to

$$P^T L P > 0,$$

where $P = [p_1, \ldots, p_{n-1}] \in \mathbb{R}^{n \times (n-1)}$ with each p_i chosen as $p_i^T \mathbf{1} = 0$ and $p_i^T p_j = 0$.

Using Theorem 6, the problem of maximizing (5.20) can be restated as

$$\text{maximize}_{x\in\mathbb{R}^n} \quad \gamma \tag{5.22}$$
$$\text{subject to } d_{ij} \triangleq \|x_i - x_j\|^2 \geq \rho, \tag{5.23}$$
$$P^T L(x) P \geq \gamma I_{n-1}. \tag{5.24}$$

Differentiating both sides of (5.23) with respect to time gives

$$2(\dot{x}_i(t) - \dot{x}_j(t))^T (x_i(t) - x_j(t)) = \dot{d}_{ij}(t). \tag{5.25}$$

Applying Euler's first discretization method to (5.25) yields

$$x(t) \to x(k), \quad \dot{x}(t) \to \frac{x(k+1) - x(k)}{\Delta t}. \tag{5.26}$$

Using (5.26), we can rewrite (5.25) as

$$2(x_i(k+1) - x_j(k+1))^T (x_i(k) - x_j(k)) = d_{ij}(k+1) + d_{ij}(k).$$

Similarly, we can discretize the Laplacian matrix as

$$[L(k)]_{ij} \triangleq \begin{cases} -w_{ij}(k) & \text{if } i \neq j, \\ \sum_{s\neq i} w_{is}(k) & \text{if } i = j. \end{cases}$$

The optimization problem (5.22)–(5.24) can then be rewritten as

$$\begin{aligned} &\text{maximize}_{x(k+1)} \quad \gamma \\ &\quad \text{subject to } 2(x_i(k+1) - x_j(k+1))^T (x_i(k) - x_j(k)) \\ &\qquad\qquad = d_{ij}(k+1) + d_{ij}(k), \\ &\qquad d_{ij}(k+1) \geq \rho, \\ &\qquad P^T L(k+1) P \geq \gamma I_{n-1}. \end{aligned}$$

The algorithm is initialized at time $k = 0$ with an initial graph $\mathcal{G}_0$, then it proceeds in an iterative way to find graphs that maximize $\lambda_2(L(k+1))$.

5.3.3 Connectivity maintenance via adjacency matrix

Consider the multi-agent system [198]

$$\dot{x}(t) = F(x(t), u(t)) \tag{5.27}$$

where $x(t) = [x_1^T(t), \ldots, x_n^T(t)]^T$ and $u(t) = [u_1^T(t), \ldots, u_n^T(t)]$. Let the state-dependent adjacency matrix be defined as $A(x) = [a_{ij}(x)]$ with

$$a_{ij}(x) = \hat{u}(\delta - \|x_i - x_j\|),$$

where $\delta > 0$ is a given constant and

$$\hat{u}(y) = \lim_{w\to\infty,\epsilon\to 0} \sigma_w(y - \epsilon) \to \begin{cases} 1 & \text{if } y > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Here, $\sigma_w(y) = \frac{1}{1+e^{-wy}}$ is the sigmoid function. The function $\hat{u}(y)$ is a continuous approximation to the step function. Apparently, $\|x_i - x_j\| \leq \delta$ iff $a_{ij}(x) = 1$. Define the k connectivity matrix

$$C_k(x) = I + A(x) + A^2(x) + \cdots + A^k(x),$$

where k is a positive integer. Define the matrix $H_k(x) = (h_{ij}^{(k)}(x))$ as

$$H_k(x) = \hat{u}(C_k(x)),$$

where the function $\hat{u}$ is applied to each entry of $C_k(x)$. To guarantee connectivity maintenance, a sufficient condition is

$$\begin{cases} \dot{h}_{ij}^{(k)}(x(t)) \geq 0, & \text{if } h_{ij}^{(k)}(x_0) = 1, \\ \dot{h}_{ij}^{(k)}(x(t)) < 0, & \text{if } h_{ij}^{(k)}(x_0) = 0. \end{cases} \tag{5.28}$$

Using the fact that

$$\dot{h}_{ij}(x) = \nabla_x h_{ij}(x)^T \dot{x} = \hat{u}(c_{ij}(x)) \nabla_x c_{ij}(x)^T \dot{x}, \tag{5.29}$$

(5.28) can be rewritten as

$$\begin{cases} \hat{u}'(c_{ij}(x)) \nabla_x c_{ij}(x)^T \dot{x} \geq 0, & \text{if } h_{ij}^{(0)} = 1, \\ \hat{u}'(c_{ij}(x)) \nabla_x c_{ij}(x)^T \dot{x} \leq 0, & \text{if } h_{ij}^{(0)} = 0, \end{cases} \tag{5.30}$$

where $\hat{u}'$ is the derive of $\hat{u}$. Eq. (5.30) can be rewritten in a matrix form as

$$G(x)\dot{x} \geq 0, \tag{5.31}$$

where $G(x)$ is a matrix depending on $\hat{u}'(c_{ij}(x))$ and $\nabla_x c_{ij}(x)^T$. The system dynamics satisfying connectivity maintenance can be rewritten as

$$\begin{aligned} \dot{x} &= F(x, u), \\ \text{s. t. } & G(x)F(x, u) \geq 0. \end{aligned}$$

5.3.4 Integration with multi-robot cooperative control

For multi-agent coordination, connectivity of the network topology describing sensing/communication among the agents is critical in guaranteeing the convergence of the cooperative control algorithms. However, in most cases, the connectivity among the agents is assumed a priori. It is shown in [199] that under the traditional consensus algorithm

$$u_i = -\sum_{j=1}^{n} a_{ij}(x_i - x_j) \tag{5.32}$$

the connectivity of the network might be lost even if the initial network is connected. To deal with the issue, the authors of [199] propose the following distributed consensus algorithm

$$\dot{x}_i = -\sum_{j=1}^{n} a_{ij} w(x_i - x_j)(x_i - x_j), \tag{5.33}$$

where $w(x_i - x_j)$ is a state-dependent weight given by

$$w(x_i - x_j) = \frac{2\delta - \|x_i - x_j\|}{(\delta - \|x_i - x_j\|)^2}$$

with $\delta > 0$ being a given constant. It is shown that under the algorithm (5.33), consensus is reached asymptotically while maintaining connectivity.

Ref. [200] proposes a general class of distributed potential-based control laws with the connectivity preserving property for single-integrator agents. The potential functions are smooth, which give bounded control inputs. Using the Rician fading model for the communication channel, Ref. [201] solves a connectivity maintenance problem that arises when two mobile autonomous agents navigate in an environment containing obstacles. A gradient-based control strategy is described in [202] that exploits decentralized estimation of the algebraic connectivity. Ref. [203] analyzes the problem of maintaining connectivity in the presence of a jammer, which can be seen as a variation of the standard connectivity maintenance problem. A novel passivity-based distributed strategy is proposed in [204] to enforce connectivity maintenance for a group of robots in a flexible way. Ref. [205] introduces a connectivity maintenance

strategy based on a distributed procedure for the estimation of the algebraic connectivity of the graph.

5.4 Time-triggered measurement (sampled data)

Consider a multi-agent system with the following double-integrator dynamics:

$$\begin{aligned} \dot{r}_i &= v_i, \\ \dot{v}_i &= u_i, \end{aligned} \tag{5.34}$$

where $r_i \in \mathbb{R}^m$ and $v_i \in \mathbb{R}^m$ are, respectively, the position and velocity of the ith agent, and u_i is the control input. Suppose the measurement can only be made at discrete times, and the control inputs are based on zero-order hold as

$$u_i(t) = u_i[k], \quad kT \leq t < (k+1)T, \tag{5.35}$$

where k denotes the discrete-time index, T denotes the sampling period, and $u_i[k]$ is the control input at $t = kT$. By using direct discretization [206], the continuous-time system can be discretized as

$$\begin{aligned} r_i[k+1] &= r_i[k] + Tv_i[k] + \frac{T^2}{2}u_i[k], \\ v_i[k+1] &= v_i[k] + Tu_i[k], \end{aligned}$$

where $r_i[k]$ and $v_i[k]$ denote, respectively, the position and velocity of the ith agent at $t = kT$.

Ref. [207] studies two kinds of consensus algorithms. One is

$$u_i[k] = -\sum_{j=1}^{n} a_{ij}(r_i[k] - r_j[k]) - \alpha v_i[k]. \tag{5.36}$$

Algorithm (5.36) is called an absolute damping algorithm due the absolute damping term $-\alpha v_i[k]$. This term will drive the velocity of all the agents to zero. The other is

$$u_i[k] = -\sum_{j=1}^{n} a_{ij}(r_i[k] - r_j[k]) - \alpha(v_i[k] - v_j[k]). \tag{5.37}$$

Compared with (5.36), the only difference is to replace the absolute damping term with the relative damping term $-\alpha(v_i[k]-v_j[k])$. The relative damping term will only drive the velocities of the agents to reach a consensus, not necessarily zero.

For the absolute damping algorithm (5.36), the main convergence results are given as follows. If the graph is undirected and the gain α and the sampling time T are chosen from the set

$$S_r = \left\{(\alpha, T) : -\frac{T^2}{2}\min_i \mu_i < \alpha T < 2\right\}, \tag{5.38}$$

where μ_i is the ith eigenvalue of the Laplacian matrix, then a consensus can be reached. In addition, condition (5.38) is also necessary. For directed graphs, the necessary and sufficient conditions are given as

1. $0 < \alpha T < 2$;
2. When $\mathrm{Re}(\mu_i) < 0$ and $\mathrm{Im}(\mu_i) = 0$, $(\alpha, T) \in S_r$, where S_r is defined in (5.38);
3. When $\mathrm{Re}(\mu_i) < 0$ and $\mathrm{Im}(\mu_i) \neq 0$, α and T satisfy $\alpha/T > -|\mu_i|^2/2\mathrm{Re}(\mu_i)$ and $T < \bar{T}_i$, where

$$\bar{T}_i = \frac{-2\alpha\mathrm{Re}(\mu_i)[\mathrm{Re}(\mu_i)+\alpha] - 2\sqrt{\alpha^2[\mathrm{Re}(\mu_i)]^2[\mathrm{Re}(\mu_i)+\alpha]^2 - \mathrm{Re}(\mu_i)|\mu_i|^2[\mathrm{Im}(\mu_i)]^2}}{\mathrm{Re}(\mu_i)|\mu_i|^2}.$$

For the relative damping algorithm (5.37), the main convergence results are stated as follows. If the graph is undirected, the necessary and sufficient condition guaranteing consensus is that

$$Q_r = \left\{(\alpha, T) : \frac{T^2}{2} < \alpha T < -\frac{2}{\min_i \mu_i}\right\}.$$

For directed graph, the necessary and sufficient condition is

$$(\alpha, T) \in Q_r \cap Q_c,$$

where

$$Q_c \triangleq \cap_{\forall \mathrm{Re}(\mu_i)<0 \text{ and } \mathrm{Im}(\mu_i)\neq 0}\left\{(\alpha, T) : \frac{1}{2} < \frac{\alpha}{T}, B_i < 0\right\}, \tag{5.39}$$

where $B_i = \left(\frac{4\mathrm{Re}(\mu_i)}{|\mu_i^2|T^2} + 2\frac{\alpha}{T}\right)\left(1-2\frac{\alpha}{T}\right)^2 + \frac{16\mathrm{Im}(\mu_i)^2}{|\mu_i|^4T^4}$.

5.5 Event-triggered measurement

Consider single-integrator multi-agent systems with x_i the state and u_i the input of agent i. Suppose that there is a sequence of event $t_0, t_1, \ldots$. The control input is updated only at the event times, i.e.,

$$u(t) = u(t_i), \quad \forall t \in [t_i, t_{i+1}).$$

Thus, the control input is piecewise constant between two consecutive event times. Define the state measurement error as

$$e(t) = x(t_i) - x(t), \quad t \in [t_i, t_{i+1}). \tag{5.40}$$

The event-triggered control law is given by

$$u(t) = -Lx(t_i), \quad t \in [t_i, t_{i+1}). \tag{5.41}$$

The closed-loop system is

$$u(t) = -L[x(t) + e(t)]. \tag{5.42}$$

Define the consensus Lyapunov function candidate $V(x) = \frac{1}{2}x^T Lx$, where L is the Laplacian matrix. The derivative of V is

$$\dot{V} = -\|Lx\|^2 - x^T L^2 e.$$

By imposing

$$\|e\| \leq \sigma \frac{\|Lx\|}{\|L\|}, \tag{5.43}$$

where $\sigma > 0$, we have $\dot{V} < 0$. Therefore, the event times are defined at the times satisfying

$$\|e\| = \sigma \frac{\|Lx\|}{\|L\|}. \tag{5.44}$$

In addition, it has been shown that the Zeno behavior can be avoided [208], i.e., there exists a strictly positive lower bound on the inter-event times. It is worth pointing out that the triggering condition (5.44) requires the global information of the MAS, since $\|x\|$ and $\|Lx\|$ need to be computed.

Distributed event-triggered approach

In order to implement the event-triggering condition (5.44), the agents need to know the global measurement error e, which makes the algorithm centralized. A distributed counterpart is given in [208] as

$$u_i(t) = -\sum_{j=1}^{n} a_{ij}[x_i(t_k^i) - x_j(t_{k'(t)}^j)],$$

where $k'(t) = \arg\min_{t \geq t_j^i}\{t - t_l^j\}$. The corresponding measurement error is then given by

$$e_i(t) = x_i(t_k^i) - x_i(t), \quad t \in [t_k^i, t_{k+1}^i). \tag{5.45}$$

For each agent, its control inputs are updated when

$$e_i^2 = \frac{\sigma_i a(1 - a|N_i|)}{|N_i|} z_i^2, \tag{5.46}$$

where $z_i = \sum_{j \in N_i}(x_i - x_j)$, $|N_i|$ denotes the number of neighbors of agent i, and $\sigma_i > 0$. However, for the distributed event-triggered condition (5.46), it can only guarantees that at least one agent has strictly positive inter-event time interval.

Except for the form in (5.45), the measurement error can be defined in different ways in order to gain additional benefits. The following combinational measurement error is proposed in [209]

$$e_i(t) = q_i(t_k^i) - q_i(t), \quad t \in [t_k^i, t_k^{i+1}) \tag{5.47}$$

to reduce the update frequency of the proposed controller. The same measurement error is also adopted in [210]. Meanwhile, there is also interest in using model-based measurement error:

$$e_i(t) = e^{A(t - t_k^i)} x_i(t_k^i) - x_i(t), \quad t \in [t_k^i, t_{k+1}^i). \tag{5.48}$$

A primary motivation of employing (5.48) is to tackle the difficulties raised by agents' general linear dynamics. Some recent works in this direction can be found in [211, 212].

Self-triggered control

In the centralized and distributed event-triggered control laws, it is apparent that continuous measurements of the measurement error are required for checking the event-triggered conditions. Thus, self-triggered is proposed to solve this issue. In self-triggered control, the next triggering time t_{k+1} is predetermined at the previous triggering time t_k and no measurement needs to be taken between two consecutive event times. In [208], the self-triggering law is given by

$$t_{k+1} - t_k \leq \frac{-2\sigma^2[Lx(t_k)]^T L^2 x(t_k) + \sqrt{\Delta}}{2(\|Lx(t_i)\|^2\|L\|^2 - \sigma^2\|L^2x(t_i)\|^2}, \tag{5.49}$$

where

$$\begin{aligned}\Delta = {} & 4\sigma^2\|(Lx(t_k))^T L^2 x(t_k)\|^2 \\ & + 4\sigma^2\|L^2x(t_k)\|^2\left(\|Lx(t_i)\|^2\|L\|^2 - \sigma^2\|L^2x(t_i)\|^2\right).\end{aligned} \tag{5.50}$$

For the distributed case, the self-triggered rule is defined in the following definition [208].

Definition 9. Let $\beta_i = (\sigma_i a(1 - a|N_i|)/|N_i|)$, $P_i = -|N_i|\rho_i + \sum_{j\in N_i}\rho_j$, and $\Phi_i = |N_i|\rho_i + \sum_{j\in N_i}(\rho_j(t_k^i - t_{k'}^j))$, where ρ_i are constants. If there is a $\xi_i \geq 0$ such that $|\rho_i|\xi_i = \sqrt{\beta_i}|P_i\xi_i + \Phi_i|$, then the next update time t_{k+1}^i takes place at most ξ_i times units after t_k^i. Agent i also checks this condition whenever its control law is updated due to an update of the error from one of its neighbors. Otherwise, if the inequality $|\rho_i|\xi_i \leq \sqrt{\beta_i}|P_i\xi_i + \Phi_i|$ holds for all $\xi \geq 0$, then agent i waits until the next update of the control law of its neighbors to recheck this condition.

Ref. [213] investigates distributed multi-agent coordination via event-triggered strategies, where the control input of each agent is piecewise constant with directed time-varying communication graphs under neighbor-synchronous and asynchronous updating protocols, respectively. Ref. [214] studies a distributed event-triggered communication and consensus strategy, where the proposed strategy prescribes isolated event times where both communication and controller updates occur. A novel distributed event-triggered algorithm for estimating the algebraic

connectivity is proposed in [215], which relies on the distributed computation of the powers of matrices. A self-triggered consensus algorithm for multi-agent systems is proposed in [216], where each agent receives the state information of its neighbors and computes the average state of its neighborhood. In [211], a novel event-triggered consensus protocol is proposed, where each agent implements a model of the decoupled dynamics of its neighbors. Ref. [217] studies the output consensus problem of networked passive systems with event-driven communication, in which the information exchange among the coupled agents are event-based rather than pre-scheduled periodically.

6

Algorithms

6.1 Proportional (P) control

6.1.1 P coordination algorithms

Proportional (P) control is a linear feedback control where the control input is proportional to the gradient of the quadratic error (the difference between desired value and current state value). Consider a single-integrator multi-agent system

$$\dot{x}_i = u_i, \tag{6.1}$$

where $x_i \in \mathbb{R}$ denotes the state of agent i and $u_i \in \mathbb{R}$ is its control input. A P-consensus input is given by

$$u_i = -\kappa_p \sum_{j=1}^{n} a_{ij}(x_i - x_j), \tag{6.2}$$

where $\kappa_p > 0$ is a control gain. Let $u \triangleq [u_1, \ldots, u_n]$. The control input can be rewritten as

$$u = -\kappa_p L x. \tag{6.3}$$

where L is the Laplacian matrix. It can be shown that for connected graph, $Lx = 0$ iff a consensus is reached. This suggests that Lx is

a consensus error. Hence, (6.3) is a P consensus algorithm. Let $x \triangleq [x_1, \ldots, x_n]^T$. The closed-loop system is

$$\dot{x} = -\kappa_p L x. \tag{6.4}$$

It is shown that as long as the P control gain κ_p is positive, a consensus is reached asymptotically.

When there exists a leader agent x_0, the P algorithm takes the following form

$$u_i = -\kappa_p \left[\sum_{j=1}^{n} a_{ij}(x_i - x_j) + a_{i0}(x_i - x_0) \right], \tag{6.5}$$

where $a_{i0} = 1$ if agent i has access to the state of the leader, and $a_{i0} = 0$ otherwise. The objective of (6.5) is to drive the agents to track the leader state x_0. Define the tracking error $e \triangleq [e_1, \ldots, e_n]$ with $e_i \triangleq x_i - x_0$. Eq. (6.5) can be written in a vector form as

$$u = -\kappa_p (L + A) e.$$

The matrix $(L + A)$ is positive definite provided that for any agent there is a path from the leader to the agent, i.e., any agent can access the information of the leader directly or indirectly. Hence, $(L + A)e$ is a measure of tracking error. If x_0 is fixed, the dynamics of the tracking error e is given by

$$\dot{e} = -\kappa_p (L + A) e.$$

In this case, all the followers will track the leader asymptotically. The result can be extended to the case where the leader is moving, but finally converges to some fixed point.

6.1.2 Drawback of P algorithms

P control has one drawback that when external input appears in closed-loop dynamics, a control error cannot be eliminated by proportional control alone. Recall the leader-following problem stated in the last subsection. If the leader x_0 is time-varying, the tracking error dynamics becomes

$$\dot{e} = -\kappa_p (L + A) e + \mathbf{1}_n \otimes u_0, \tag{6.6}$$

where $u_0 \in \mathbb{R}$ denotes the input of the leader agent. The tracking error e will never reach zero for the error dynamics (6.6). When the tracking error e is "large", the P control term $\kappa_p(L+A)e$ is also "large" and can "cover" the external input $\mathbf{1}_n \otimes u_0$. In this case, the tracking error is reduced. However, when the tracking error reduces to a certain level, it is not "large" enough to "cover" the external input. In this case, the tracking error cannot be reduced anymore. An intuition is that when the control κ_p increases, the stationary tracking error is reduced.

The above statements can be verified as follows. Define the Lyapunov function candidate $V = \frac{1}{2}e^T e$. The derivative of V is

$$\begin{aligned}\dot{V} &= e^T\left[-\kappa_p(L+A)e + \mathbf{1}_n \otimes u_0\right] \\ &= -\kappa_p e^T(L+A)e + e^T(\mathbf{1}_n \otimes u_0) \\ &\leq -\kappa_p \lambda_{\min}(L+A)\|e\|^2 + \|e\|\sqrt{n}|u_0| \\ &= -\|e\|\left(\kappa_p \lambda_{\min}(L+A)\|e\| - \sqrt{n}|u_0|\right).\end{aligned}$$

If $\|e\| > \frac{\sqrt{n}|u_0|}{\kappa_p \lambda_{\min}(L+A)}$, we have $\dot{V} \leq 0$. This indicates that the tracking error e will finally satisfy

$$\|e\| \leq \frac{\sqrt{n}|u_0|}{\kappa_p \lambda_{\min}(L+A)}. \tag{6.7}$$

It follows from (6.7) that the proportional gain κ_p can be used to tune the tracking error: the larger the proportional gain, the smaller the tracking error. However, this tracking error is also related to the number of agents n. For a network with a huge number of nodes, it might require a substantial proportional gain to get a small enough tracking error. This poses a challenge for proportional control of network systems.

6.2 Proportional-integral (PI) control

6.2.1 Single-order PI control

Let u_0 be a constant input in the leader-following tracking problem. As we have shown in the previous section, there always exists a steady-state tracking error under P control. In this section, we consider the following

PI control algorithm

$$\begin{aligned}\dot{x} &= -k_p(L+A)(x-\mathbf{1}x_0) - k_i(L+A)v,\\ \dot{v} &= (L+A)(x-\mathbf{1}x_0),\end{aligned} \tag{6.8}$$

which can be rewritten as

$$\begin{aligned}\dot{e} &= -k_p(L+A)e - k_i(L+A)v + (\mathbf{1}_n \otimes u_0),\\ \dot{v} &= (L+A)e,\end{aligned} \tag{6.9}$$

where $e \triangleq x-\mathbf{1}x_0$ and u_0 is a constant. Here, $v(t) = \int_{t_0}^{t}[-(L+A)e(\tau)]\mathrm{d}\tau$, which represents the integral information of the tracking error $(L+A)e(\tau)$, and $k_i > 0$ is the integral control gain. The equilibrium of the system (6.9) is

$$\bar{e} = 0, \quad \bar{v} = -\frac{1}{k_i}(\mathbf{1}_n \otimes u_0).$$

Let $e_v = v - \bar{v}$. The system (6.9) is rewritten as

$$\begin{aligned}\dot{e} &= -k_p(L+A)e - k_i(L+A)e_v,\\ \dot{e}_v &= (L+A)e,\end{aligned}$$

or equivalently

$$\begin{bmatrix}\dot{e}\\ \dot{e}_v\end{bmatrix} = \left\{\begin{bmatrix}-k_p & -k_i\\ 1 & 0\end{bmatrix} \otimes (L+A)\right\}\begin{bmatrix}e\\ e_v\end{bmatrix}. \tag{6.10}$$

Note that $(L+A)$ is positive definite, and $\begin{bmatrix}-k_p & k_i\\ -1 & 0\end{bmatrix}$ is Hurwitz as long as $k_p > 0$ and $k_i > 0$. It follows that the matrix $\left\{\begin{bmatrix}-k_p & -k_i\\ -1 & 0\end{bmatrix} \otimes (L+A)\right\}$ is Hurwitz as long as $k_p > 0$ and $k_i > 0$, which yields that $e \to 0$ and $e_v \to 0$, i.e., zero tracking error will be achieved asymptotically.

6.2.2 Higher-order PI control

In this subsection, we explore the possibility of using higher-order PI control, such that the exact tracking is achieved for the leader satisfying $x_0^{(p)} = 0$, where $x_0^{(p)}$ denotes the pth order derivative. For the ease of

discussion, we consider the case that $p = 3$. A similar result holds for the general case.

In this case, the control algorithm is defined as

$$\begin{aligned}\dot{x} &= -k_p(L+A)(x-x_0) - k_{iv}(L+A)v,\\ \dot{v} &= (L+A)(x-x_0) - k_{iw}(L+A)w,\\ \dot{w} &= (L+A)(v + k_{iv}^{-1}(L+A)^{-1}\mathbf{1}_n\dot{x}_0).\end{aligned} \tag{6.11}$$

Let $e \triangleq x - x_0$, $e_v \triangleq v + k_{iv}^{-1}(L+A)^{-1}\mathbf{1}_n\dot{x}_0$, and $e_w \triangleq w + k_{iw}^{-1}k_{iv}^{-1}(L+A)^{-2}\mathbf{1}_n\ddot{x}_0$. The error dynamics can be rewritten as

$$\begin{bmatrix}\dot{e}\\ \dot{e}_v\\ \dot{e}_w\end{bmatrix} = \left\{\begin{bmatrix}-k_p & -k_{iv}0 & \\ 1 & 0 & -k_{iw}\\ 0 & 1 & 0\end{bmatrix} \otimes (L+A)\right\}\begin{bmatrix}e\\ e_v\\ e_w\end{bmatrix}.$$

It can be shown that as long as $k_p > 0$, $k_{iv} > 0$, and $k_{iw} > 0$, the matrix $\left\{\begin{bmatrix}-k_p & -k_{iv}0 & \\ 1 & 0 & -k_{iw}\\ 0 & 1 & 0\end{bmatrix} \otimes (L+A)\right\}$ is Hurwitz. This also indicates that the system (6.11) is asymptotically stable, i.e., a zero-tracking error can be achieved. However, we notice that the higher-order PI control system (6.11) is not distributed. This can be verified by the dynamics of the higher-order integral variable w. For each component of w, the information of $\dot{x}_0$ is needed.

Ref. [135] analyzes two different estimation algorithms for distributed average tracking in sensing and communication networks, a proportional algorithm and a proportional-integral algorithm. Ref. [218] designs a clock synchronization algorithm based on a PI-like consensus protocol where the proportional part compensates the different clock speeds and the integral part eliminates different clock offsets. Ref. [219] investigates the use of distributed PID control to achieve consensus in networks of homogeneous and heterogeneous linear systems. Ref. [75] presents a proportional plus damping controller that can asymptotically drive a network of nonidentical Euler–Lagrange systems toward a consensus point. Ref. [220] investigates the use of distributed PI actions to achieve consensus and synchronization in complex networks.

6.3 Adaptive control

6.3.1 Direct adaptive control

Direct adaptive control is a scheme where we adapt the parameters of the controller. Consider the nonlinear multi-agent system

$$\dot{x}_i = f(x_i) + u_i \tag{6.12}$$

where $f(x_i)$ is a nonlinear function describing the dynamics of agent i and u_i is the control input. For system (6.12), the consensus input is given by

$$u_i = \sigma \sum_{j=1}^{n} a_{ij}(h(x_i) - h(x_j)), \tag{6.13}$$

where $h(x_i)$ is the output function and a_{ij} is the (i,j)th entry of the adjacency matrix. Here, the control gain σ should be large enough, which normally relies on the global network topology to ensure consensus. This makes the algorithm (6.13) not fully distributed. Adaptive control can be used to design fully distributed consensus algorithm. In [221], two different kinds of adaptive control algorithms are proposed. One approach is to use the following node-based adaptive gain

$$\dot{\sigma}_i = \left\| \sum_{j=1}^{n} a_{ij}(h(x_i) - h(x_j)) \right\|$$

to replace the global gain σ in (6.13). The other approach is to use the edge-based adaptive gain

$$\dot{\sigma}_{ij} = \|h(x_j) - h(x_i)\|.$$

6.3.2 Indirect adaptive control

Indirect adaptive control is a scheme where we estimate the unknown parameters of the model and use these estimates to update controller parameters. Consider the following nonlinear multi-agent systems [222]

$$\dot{x}_i = f_i(x_i) + u_i + \rho_i, \quad i = 1, \ldots, n, \tag{6.14}$$

where x_i is the agent state, f_i is a smooth nonlinear function, u_i is the control input, and ρ_i is the external disturbances. The control input is designed in [222] as

$$u_i = \frac{\sum_{j=1}^{n} a_{ij}\dot{x}_j + a_{i0}\dot{x}_0}{\sum_{j=1}^{n} a_{ij}} - \hat{f}_i(x_i) + k_i e_i + \psi_i, \tag{6.15}$$

where $k_i > 0$ is a control gain. Here, the radial basis function neural network (RBFNN) is used to estimate the unknown function f_i in (6.14) as follows:

$$\hat{f}_i(x_i) = \hat{W}_i^T S_i(x_i)$$

where $\hat{W}_i$ is a weight matrix and $S_i = [S_{i1}, \ldots, S_{il}]$ with $S_{ij}(x_i) \triangleq \exp\left[\frac{-(x_i-\mu_j)^T(x_i-\mu_j)}{\sigma_j^2}\right]$ with u_j the centring of the receptive field and $(\sqrt{2}\sigma_j)/2$ the width of the Gaussian function. The tracking error e_i is defined as $e_i \triangleq \bar{x}_i - x_i$ with $\bar{x}_i \triangleq \frac{\sum_{j=1}^{n} a_{ij}x_j + a_{i0}x_0}{\sum_{j=0}^{n} a_{ij}}$. The term $\psi_i = (\psi_{i1}, \ldots, \psi_{ip})^T$ is used to counteract the effect of the external disturbance ρ_i, where

$$\psi_{ij} = \delta_{Mi} \tanh\left(\frac{n k_u \delta_{Mi} e_{ij}}{\epsilon_i}\right), \tag{6.16}$$

where e_{ij} is the jth entry of e_i, $k_u = 0.2785$, ϵ_i is a design parameter, and δ_{Mi} is a sufficiently large positive constant. The reason for choosing $k_u = 0.2785$ is given in Lemma 1 of [223]. Under the proposed neural network based control input (6.15), it is shown that consensus is reached asymptotically [222].

6.3.3 Hybrid adaptive control

Hybrid adaptive control includes the adaptive design of the controller parameters and model parameters. Consider the following first-order linear parameterized agent dynamics:

$$\dot{x}_i = b_i u_i + \phi_i(x_i)^T \theta_i, \quad i = 1, \ldots, n. \tag{6.17}$$

where x_i is the position, u_i is the control input, b_i is an unknown constant, ϕ_i is a known continuous vector-valued function, and θ_i is

an unknown parameter. In the above model, there are two kinds of uncertainties: one is the unknown control direction b_i and the other is the unknown parameter θ_i.

The unknown parameter θ_i is updated by

$$\dot{\hat{\theta}}_i = \xi_i \phi_i(x_i) \left[\sum_{j=1}^{n} a_{ij}(x_i - x_j) \right] \tag{6.18}$$

with ξ_i a positive constant. Design the Nussbaum-type function [224]

$$N_0(\kappa) = \cosh(\lambda\kappa)\sin(\kappa),$$

where $\cosh(\lambda\kappa) = (e^{\lambda\kappa} + e^{-\lambda\kappa})/2$ with $\lambda > 0$ and $\kappa \in \mathbb{R}$. The adaptive control input is given by

$$u_i = -N_0(k_i) \left[\sum_{j=1}^{n} a_{ij}(x_i - x_j) + \phi_i(x_i)^T \hat{\theta}_i \right]. \tag{6.19}$$

Here, k_i is updated by

$$\dot{k}_i = \gamma_i \left[\sum_{j=1}^{n} a_{ij}(x_i - x_j) \right] \left[\sum_{j=1}^{n} a_{ij}(x_i - x_j) + \phi_i(x_i)^T \hat{\theta}_i \right] \tag{6.20}$$

with $\gamma_i > 0$ being a positive constant. The update law of k_i, Eq. (6.20), is direct adaptive control and the estimate of θ_i, Eq. (6.18), is indirect adaptive control. It is shown in [224] that if the parameter λ is sufficiently large, under the adaptive control law (6.19), a consensus is reached for the system (6.17).

6.4 Model predictive control

6.4.1 Distributed model predictive control

Consider a set of n agents with the following discrete-time dynamics [66]

$$\dot{x}_i(k+1) = x_i(k) + u_i(k), \quad i = 1, \ldots, n \tag{6.21}$$

where $x_i \in \mathbb{R}$ is the state and $u_i \in \mathbb{R}$ is the control input. Let A denote the normalized adjacency matrix of the communication graph $\mathcal{G}$. The

sum of the entries in each row of A equals to one. Consider a model predictive control (MPC) consensus algorithm on a finite horizon of length $T \geq 1$. We associate with each agent a control input vector $U_i(k) \triangleq [u_i^T(k), \ldots, u_i^T(k+T-1)]$ and the cost

$$J_i[x(k), U_i(k)] = J_i^x[x(k), U_i(k)] + J_i^u[U_i(k)], \tag{6.22}$$

where

$$J_i^x[x(k), U_i(k)] = q_i \sum_{j=1}^{T} \left\| x_i(k+j) - \sum_{l=1}^{n} a_{il} x_i(k) \right\|^2, \tag{6.23}$$

$$J_i^u(U_i(k)) = r_i \sum_{j=1}^{T} \|u_i(k+j)\|^2, \tag{6.24}$$

with q_i, r_i positive weights, and $x(k) = [x_1, \ldots, x_n]^T$. Eq. (6.23) measures the consensus error, and Eq. (6.24) measures the control effort over the sliding window $[k, k+T]$.

The objective of MPC is to solve the following finite-horizon control problem for each agent i governed by (6.21):

$$\begin{aligned} \text{minimize}_{U_i(k)} \quad & J_i[x(k), U_i(k)] \\ \text{subject to} \quad & \dot{x}_i(k+1) = x_i(k) + u_i(k), \\ & \|u_i(k+j)\| \leq \bar{u}, \ \forall i = 1, \ldots, n, \ j = 0, \ldots, T-1, \end{aligned} \tag{6.25}$$

where $\bar{u}$ is a positive constant. It is shown in [66] that under the MPC algorithm (6.25), a consensus is reached asymptotically if the graph $\mathcal{G}$ is connected across the interval $[k_0, \infty)$ for all $k_0 \in \mathbb{N}$.

6.4.2 Parameterized model predictive control

Consider the nonlinear agents' dynamics [225]

$$\dot{x}_i = f_i(x_i, u_i).$$

The control inputs of each agent is a feedback of the form $u_i = \kappa_i(x_i, x_{-i}^d, \theta_i)$ where x_{-i}^d is a set of the states upon which agent i's dynamics depend and θ_i is a tunable parameter vector. At each time step, we need choose the optimal parameters θ_i for the control input by

considering the following cost function

$$\text{minimize } J = \sum_{i=1}^{n} J_i,$$

where J_i is the cost of agent i given by

$$J_i = \int_{t_0}^{t_f} L_i(x_i, x_{-i}^c, \theta_{-i}^c)\mathrm{d}t + \phi_i(x_i(t_f), x_{-i}^c, \theta_{-i}^c).$$

Here, x_{-i}^c represents the states of the neighboring agents, θ_{-i}^c is the parameter vectors of neighboring agents, t_0 is the initial time, and $t_f = t_0 + \Delta$ with Δ the time horizon. The above scheme is called parameterized model predictive control, where the optimization variable is the parameter. The benefits of the parameterized MPC control scheme is that it reduces the communication and computational cost, because an agent can communicate/compute an entire trajectory by solely communicating/computing several parameters. .

To solve the problem, the dual-decomposition technique [226, 227] is used. This technique decomposes the problem into subproblems which can be handled by individual agents. To this aim, rewrite the problem as

$$\begin{aligned} &\text{minimize } J = \sum_{i=1}^{n} J_i \\ &\text{subject to } \theta_{ij} = \theta_{jj}, \end{aligned} \tag{6.26}$$

where θ_{ij} is the parameter of the jth neighbor of agent i. To form the dual, Lagrange multipliers are introduced, which gives

$$\max_{\mu} \min_{\theta} \hat{J} = \sum_{i=1}^{n} J_i + \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} \mu_{ij}^T (\theta_{ij} - \theta_{ii}) \tag{6.27}$$

where μ_{ij} are the Lagrange multiplier vectors.

6.5 Passivity-based control

6.5.1 Passivity control for output synchronization

Consider the following nonlinear affine multi-agent systems (see [228]):

$$\begin{aligned} \dot{x}_i &= f_i(x_i) + g_i(x_i)u_i, \\ y_i &= h_i(x_i) \end{aligned} \tag{6.28}$$

where x_i is the state, u_i is the input, and y_i is the output. The functions f_i, g_i and h_i are smooth. We assume that for each agent i, its dynamics (6.28) is passive, i.e., there exists a smooth storage function V_i such that $V_i(x_i) \geq 0$, $V_i(0) = 0$, and $S_i(x_i) \geq 0$ such that

$$\left[\frac{\partial V_i(x_i)}{\partial x_i}\right]^T f_i(x) = -S_i(x_i),$$
$$\left[\frac{\partial V_i(x_i)}{\partial x_i}\right]^T g_i(x) = h_i^T(x_i).$$

Design the output synchronization algorithm [228]

$$u_i = -\sum_{j=1}^{n} a_{ij}(y_i - y_j), \quad i = 1, \ldots, n. \tag{6.29}$$

Consider the following Lyapunov function candidate

$$V = \frac{1}{2}\sum_{i=1}^{n} V_i, \tag{6.30}$$

where V_i is the storage function of agent i. The derivative of V along trajectory of the system (6.28) is given by

$$\begin{aligned}
\dot{V} &= \sum_{i=1}^{n}[-S_i(x_i) + y_i^T u_i] \\
&= -\sum_{i=1}^{n} S_i(x_i) - \sum_{i=1}^{n} y_i^T \sum_{j=1}^{n} a_{ij}(y_i - y_j) \\
&= -\sum_{i=1}^{n} S_i(x_i) - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}(y_i - y_j)^T(y_i - y_j) \\
&\leq 0.
\end{aligned}$$

The second last inequality is due uses the fact that the graph is balanced. Applying Lasalle's invariance principle yields that output synchronization is achieved for the system (6.28) under the input (6.29), i.e., $\|y_i - y_j\| \to 0$ for all i, j.

Suppose that the dynamics of the agents are governed by the identical linear systems of the form

$$\dot{x}_i = Ax_i + Bu_i,$$
$$y_i = Cx_i.$$

Let the input be given by (6.29). It it straightforward to see that if (C, A) is observable, then output synchronization indicates state synchronization, i.e., $\|x_i - x_j\| \to 0$ [228].

6.5.2 Passivity control for group coordination

Consider the multi-agent systems with the dynamics:

$$\begin{aligned} \dot{\xi}_i &= f_i(\xi_i, u_i), \\ y_i &= h_i(\xi_i, u_i), \end{aligned} \tag{6.31}$$

where ξ_i is the state of agent i, u_i is the control input, and y_i is the output. Here, the functions f_i and h_i are locally Lipschitz functions such that $h_i(0, 0) = 0$ and

$$f_i(0, u_i) = 0 \to u_i = 0. \tag{6.32}$$

The system (6.31) is assumed to be strictly passive. For the system (6.31), the objective is to develop distributed coordination control law such that the following two objectives are achieved simultaneously:

1. Each agent achieves a prescribed common velocity vector, i.e., $\lim_{t\to\infty} |\dot{x}_i - v(t)| = 0$ for $i = 1, \ldots, n$.

2. If agent i and agent j are neighbors, then the difference variable

$$z_k \triangleq \sum_{l=1}^{n} d_{lk} x_l = \begin{cases} x_i - x_j & \text{if } i \text{ is the positive end} \\ x_j - x_i & \text{if } i \text{ is the negative end} \end{cases} \tag{6.33}$$

converges to a prescribed compact set $\mathcal{A}_k$. Here, d_{ik} is the (i, k)th entry of the incidence matrix.

The group coordination behaviors that can be achieved by the above two objectives include, but not limited to, formation control or consensus. A feedback control input is given in [229] as

$$u_i = -\sum_{k=1}^{m} d_{ik} \psi_k(z_k). \tag{6.34}$$

The nonlinear function $\psi_k(z_k)$ is of the following form

$$\psi_k(z_k) = \nabla P_k(z_k), \tag{6.35}$$

where $P_k(z_k) : \mathcal{G}_k \to \mathbb{R}^+$ satisfies

$$\begin{aligned} P_k(z_k) \to \infty, \text{ as } z_k \to \partial\mathcal{G}_k, \\ P_k(z_k) = 0 \leftrightarrow z_k \in \mathcal{A}_k, \\ \nabla P_k(z_k) = 0 \leftrightarrow z_k \in \mathcal{A}_k, \end{aligned}$$

with $\partial\mathcal{G}_k$ denoting the boundary of $\mathcal{G}_k$.

Ref. [230] addresses passivity-based motion coordination of rigid bodies in the special Euclidean group SE(3) under the assumption that the agents exchange information over strongly connected graphs. Ref. [231] investigates the problem of synchronization of networked robotic systems with uncertainties in both kinematics and dynamics. The passivity of the robotic agents is established through adaptation to both the kinematic and dynamic uncertainties. Ref. [232] studies the output synchronization problem of networked Euler–Lagrange(EL) systems subject to nonholonomic constraints. It is shown that with proper design of the control law and some coordinate transformation, it can obtain a new state-space representation of the Euler–Lagrange system which is passive. In [233], it is shown that how heterogeneous bidirectional vehicle strings can be modelled as port-Hamiltonian systems. Ref. [217] studies the output synchronization problem of networked passive systems with event-driven communication, in which the information exchange among the coupled agents are event-based.

6.6 Sliding-mode control

Consider a multi-agent system governed by

$$M_i(x_i)\ddot{x}_i + f_i(x_i, \dot{x}_i) = u_i, \tag{6.36}$$

where x_i is the position of agent i, M_i is the mass or inertial matrix, f_i denotes the disturbance, and u_i is the input. Let J be a potential function. The potential function is selected by the system designer, encoding the desired group behavior. In [234], the authors considered potential functions of the following form:

$$J(x) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} J_{ij}\left(\|x^i - x^j\|\right),$$

where J_{ij} is the potential between agent i and j, which satisfies the following conditions.

1. The potential J_{ij} is symmetric and satisfies
$$\nabla_{x^i} J_{ij}(\|x^i - x^j\|) = -\nabla_{x^j} J_{ij}(\|x^i - x^j\|).$$
2. There exists a corresponding function $g_{ar}^{ij} : \mathbb{R}^+ \mapsto \mathbb{R}^+$ such that
$$\nabla_y J_{ij}(\|y\|) = y g_{ar}^{ij}(\|y\|).$$
3. There exist a unique $\delta_{ij} \in \mathbb{R}^+$ such that $g_{ar}^{ij}(\delta_{ij}) = 0$ and $g_{ar}^{ij}(\|y\|) > 0$ for $\|y\| > \delta_{ij}$, and $g_{ar}^{ij}(\|y\|) < 0$ for $\|y\| < \delta_{ij}$.

Define the sliding-model surface of agent i as
$$s_i = \dot{x}_i + \nabla_{x_i} J(x) = 0.$$
Design the control input u_i as
$$u_i = -u_i^0(x)\text{sgn}(x_j - x_i), \tag{6.37}$$
where u_i^0 is a time-varying function and sgn is the signum function. It follows from the results in [234] that if $u_i^0(x)$ is properly designed, under the algorithm (6.37), the desired group behavior can take place for the system (6.36).

Another example is given in [77], which considers a multi-agent system with the following dynamics
$$\ddot{y}_i = -f_i(\dot{y}_i) + g_i(\dot{y}_i)u_i,$$
where u_i and y_i are the input and output of agent i. We are interested in designing the control input u_i such that an output consensus is achieved, that is,
$$y_i - y_j = 0, \quad t \to \infty.$$
Ref. [77] proposes the following static control input
$$u_i = -\sum_{j=1}^{n} \frac{a_{ij}}{d_i}\text{sgn}(y_i - y_j),$$
where a_{ij} is the (i, j)th entry of the adjacency matrix and $d_i = \sum_{j=1}^{n} a_{ij}$. It is shown that if the nonlinear function f_i are locally passive and satisfy a local sector bound and g_i are locally passive and bounded, a consensus can be reached.

6.7 Finite-time and fixed-time control

In this section, we introduce finite-time control and fixed time control. The key difference between these two notions is that in finite-time control, the convergence time depends on the initial state of the system, while in fixed-time control, the convergence time does not rely on the initial state.

Finite-time control

Consider a network of n agents with the single-integrator dynamics:

$$\dot{x}_i = u_i. \tag{6.38}$$

The interactions among the agents are described by an undirected graph. In [76], two different kinds of discontinuous finite-time consensus are proposed:

$$\dot{x}_i = \frac{\sum_{j=1}^{n} a_{ij}(x_j - x_i)}{\|Lx\|_2}, \tag{6.39}$$

$$\dot{x}_i = \operatorname{sgn}\left(\sum_{j=1}^{n}(x_j - x_i)\right). \tag{6.40}$$

It has been shown that both algorithms guarantee that the convergence takes place in a finite time. In addition, the convergence time depends on the initial consensus error and the second smallest eigenvalue of the Laplacian matrix.

In [235], a continuous finite-time consensus algorithm is proposed for the single-integrator dynamics (6.38). The algorithm is given by

$$u_i = \sum_{j=1}^{n} a_{ij}\operatorname{sgn}(x_j - x_i)|x_j - x_i|^{\alpha_{ij}}, \tag{6.41}$$

where $0 < \alpha_{ij} < 1$. If $\alpha_{ij} = 1$, then the algorithm reduces to the typical linear consensus algorithm; if $\alpha_{ij} = 0$, the algorithm reduces to the discontinuous consensus algorithm studied in [236].

Fixed-time control

In finite-time control, the convergence takes place in a finite time. However, the convergence time depends on the initial states. Fixed-time

control goes one step further, where the convergence time is independent of the initial states of the system. In [237], the following fixed-time consensus algorithm is proposed for single-integrator systems:

$$u_i = \sum_{j=1}^{n} a_{ij}\phi_{ij}(x_j - x_i), \tag{6.42}$$

where ϕ_{ij} is the so-called action functions which are given by

$$\phi_{ij} = \alpha(x_j - x_i)^{[\mu]} + \beta(x_j - x_i)^{[\nu]}.$$

Here, α and β are positive constants, $0 < \mu < 1$, and $\nu > 1$ are control parameters, and

$$(x_j - x_i)^{[\mu]} = \text{sign}(x_j - x_i)|x_j - x_i|^{\mu}.$$

It has been shown that under the control input (6.42), the convergence time is upper bounded by

$$T^1_{\max} \triangleq \frac{2}{\bar{\alpha}(1-\mu)} + \frac{2}{\bar{\beta}(\nu-1)}, \quad \forall x_0 \in \mathbb{R}^n,$$

where

$$\bar{\alpha} = \alpha 2^{\mu}(\lambda_*(\mathcal{L}_\mu))^{\frac{\mu+1}{2}}, \quad \bar{\beta} = \beta 2^{\nu} n^{\frac{1-\nu}{2}}(\lambda_*(\mathcal{L}_\nu))^{\frac{\nu+1}{2}}.$$

Here, $\mathcal{L}_\mu$ and $\mathcal{L}_\nu$ are, respectively, the Laplacians of the graphs

$$\mathcal{G}\left(A^{[2\mu/(\mu+1)]}\right) \quad \text{and} \quad \mathcal{G}\left(A^{[2\nu/(\nu+1)]}\right).$$

Ref. [238] studies semistability and finite-time stability analysis and synthesis of systems having a continuum of equilibria. The paper merges the theories of semistability and finite-time stability to develop a rigorous framework for finite-time semistability. In particular, finite-time semistability for a continuum of equilibria of continuous autonomous systems is established. Continuity of the settling-time function, as well as Lyapunov and converse Lyapunov theorems for semistability, are also developed. Ref. [239] presents a method for achieving consensus in a finite number of time-steps, which involves a linear iteration where, at each time-step, each node updates its value to be a weighted average of its previous value and those of its neighbors. Ref. [240] discusses

the possibility of reaching consensus in finite time using only linear iterations, with the additional restrictions that the update matrices must be stochastic with positive diagonals and consistent with a given graph structure.

7

Applications

In this chapter, we introduce several applications of multi-agent systems. In particular, we will present the applications of multi-agent systems in multi-robot systems, sensor networks, smart grids, machine learning, social networks, and task migration of many-core micro-processors.

7.1 Multi-robot systems

7.1.1 Search and rescue

In this section, we are interested in the task of searching and rescuing victims trapped in a hazardous space using multi-robot systems. This task is normally performed by humans. However, in most cases, the disaster environment poses a threat to the safety of rescue workers. Rescue robots provide a promising solution to the search and rescue mission. In particular, search and rescue via multi-robot systems have received considerable attention, which has high efficiency and increased robustness. To make a team of rescue robots work together, it is critical to assign different tasks to different robots.

Given a list of N_t tasks and N_u robots, the objective of task allocation is to assign the task to the robots to maximize some global reward

while avoiding conflicts [241]. An assignment is non-conflicting if it is not simultaneously assigned to two different agents. Assume that each robot can be assigned up to L_t tasks. The assignment is completed provided that $N_{\min} \triangleq \min\{N_t, N_u L_t\}$ tasks have been assigned. The problem can be described mathematically as

$$\text{maximize} \sum_{i=1}^{N_u} \left(\sum_{j=1}^{N_t} c_{ij}(x_i, p_i) x_{ij} \right),$$

subject to

$$\sum_{j=1}^{N_t} x_{ij} \leq L_t,$$

$$\sum_{i=1}^{N_u} x_{ij} \leq 1,$$

$$\sum_{i=1}^{N_u} \sum_{j=1}^{N_t} x_{ij} = N_{\min} \triangleq \min\{N_t, N_u L_t\},$$

$$x_{ij} \in \{0, 1\}.$$

Here, $x_{ij} = 1$ if task j is assigned to agent i and $x_{ij} = 0$ otherwise, and x_i is a vector formed by all its x_{ij}. The vector p_i is an ordered sequence of tasks for agent i. The function c_{ij} is a score function, which is a function of both x_i and p_i.

In [241], the authors propose a consensus-based distributed auction algorithm to solve the above problem. The algorithm consists of two phases. The first phase is the auction process, and the second is the consensus process. In the auction phase, each robot places a bid on a task in an asynchronous way. Let c_{ij} denote the bid of robot i on task j. Each robot stores and updates two vectors. One is x_i, where $x_{ij} = 1$ if task j is assigned to robot i. The other vector is the winning bids list y_i, where y_{ij} is an estimate of the highest bid for each task. The list of valid tasks $h_i = [h_{i1}, \ldots, h_{iN_t}]$ is defined as

$$h_{ij} \triangleq \mathbb{I}(c_{ij} > h_{ij}),$$

where $\mathbb{I}(\cdot)$ is the indicator function. The second phase is consensus, where the agents run a consensus algorithm to reach an agreement on the winning bids list y_i.

7.1.2 Distributed map merging

Consider a team of n robots exploring an unknown environment [215]. At each step, robot i estimates its own poses and the position of $m_i^k \in \mathbb{N}$ features of the environment. The estimates of robot i are stored as a stochastic map with the mean $\hat{x}_i^k \in \mathbb{R}^{M_k^k}$ and covariance matrix $\Sigma_i^k \in \mathbb{R}^{M_i^k}$, where $M_i^k = \text{szr} + m_i^k \text{szf}$ with szr and szf being, respectively, the size of a robot pose and a feature position. For example, when a robot move in a 2-D environment, szr $= 3$ (position (x, y) and orientation θ) and szf $= 2$. Let $x_i^k \in \mathbb{R}^{M_i^k}$ denote a vector containing the true pose of robot i and the true position of the m_i^k feature. We have

$$\hat{x}_i^k = x_i^k + v_i^k,$$

where v_i^k is a zero mean noise with the covariance matrix $\sum_i^k$. Let H_i^k be the observation matrix, i.e.,

$$x_i^k = H_i^k x,$$

where x is the vector with the true poses of the n robots and the true positions of all the m features. The local map of each agent is a partial observation of x:

$$\hat{x}_i^k = H_i^k x + v_i^k,$$

where v_i^ks are independent noises. Let I_i^k and $\mathbf{i}_i^k$ be the information matrix and information vector of the local map, i.e.,

$$I_i^k = (H_i^k)^T \left(\Sigma_i^k\right)^{-1} H_i^k, \quad \mathbf{i}_i^k = (H_i^k)^T \left(\Sigma_i^k\right)^{-1} \hat{x}_i^k,$$

for $i \in \{1, \ldots, n\}$. The information matrix and vector of the global map is the sum of the local ones:

$$I_G^k = \sum_{i=1}^{n} I_i^k, \quad \mathbf{i}_G^k = \sum_{i=1}^{n} \mathbf{i}_i^k.$$

It is equivalent to represent the global map in terms of the mean and covariance matrix:

$$\hat{x}_G^k = (I_G^k)^{-1} \mathbf{i}_G^k, \quad \sum_G^k = (I_G^k)^{-1}.$$

The objective is to design a distributed algorithm such that all the robots can track the global map via only local communication.

In [215], the authors employ the distributed average tracking algorithm to solve the above problem of merging feature-based stochastic maps. It is shown that the estimates provided by the distributed average tracking algorithm are unbiased estimates of the true map x. The authors use a dataset consisting of bearing information obtained with vision in an environment of 60×45 m^2. The total length of the robot path is 505 m, and the step number is 3297. It has been demonstrated that the proposed method has a lower computational and computational complexity compared with the existing centralized approach.

7.1.3 Collective transport

In [242], the authors consider the problem of designing a distributed control strategy for collective transport, where each agent independently grips a complex-shaped object and work cooperatively to move the object to a prescribed destination. The object is modeled as an arbitrary connected 2D shape. The mass of the object is m. It can slide across a planar environment with coefficients of static and kinetic friction μ_s and μ_k, respectively. There are n robots applying forces to the object at arbitrary points. Denote the force of robot i by $\vec{f}_i$, which is given by

$$\vec{f}_i = C(V_{\max} - \vec{v}_i \cdot \hat{d})\hat{d},$$

where $\vec{v}_i$ is the velocity of robot i, $V_{\max}$ is the maximum speed of the robots, $C = f_0/V_{\max}$ with f_0 the magnitude of the maximum force an agent can apply, $\hat{d}$ is a unit vector in the direction of the force.

It is assumed that not all the robots have the information of the goal location. A consensus algorithm, similar to the one in [243], is employed such that all the robots can finally align in the desired direction while only a few robots can observe the goal. The robots only communicate their relative heading with neighbors. A robot that does not know the goal uses its infra-red (IR) sensors to determine the heading difference between itself and its neighbors. Here, the consensus error is defined as

$$\sum_{i=1}^{n} a_{ij}\|\theta_j\|, \quad \text{where } \theta_i \text{ is the heading difference between robots } i \text{ and } j. \tag{7.1}$$

Each robot rotates to minimize the consensus error (7.1). The above strategy is tested using the r-one robot platform. It has been shown that in all the five trials, the object was successfully transported to the goal.

Ref. [244] establishes an architecture for urban search and rescue and a methodology for mixing real-world and simulation-based testing. Ref. [245] presents an implemented algorithm for a distributed team of autonomous mobile robots to search for an object. The algorithm is fully distributed, i.e., there is no central supervisor. Ref. [246] describes a framework for controlling and coordinating a group of robots for cooperative manipulation tasks, which allows the robots to approach the object and transport the object to the desired destination. In [247], the authors present two task-allocation strategies for a multi-robot transportation system: the first one is based on a centralized planner that uses domain knowledge to solve the assignment problem in linear time and the second one uses only locally obtainable information.

7.2 Sensor networks

7.2.1 Clock synchronization

Recent advances in miniaturization, wireless communications, and digital electronics have enabled the use of sensor networks for various applications of surveillance and monitoring. Most of these applications require that the sensor nodes share a common time clock, i.e., global clock synchronization is achieved.

In a sensor network, the dynamics of the local clock of node i is given by [248]

$$\tau_i(t) = \alpha_i t + \beta_i, \tag{7.2}$$

where τ_i is the local clock reading, α_i is the local clock drift which determines the clock speed, and β_i is the local clock offset. Due to the unavailability of the time t, it is not possible to obtain the parameters α_i and β_i. Solving t in (7.2), we have $t = \frac{\tau_i(t)-\beta_i}{\alpha_i}$, substitution of which into the dynamics of node j yields that

$$\tau_j = \alpha_{ij}\tau_i + \beta_{ij},$$

where $\alpha_{ij} \triangleq \frac{\alpha_j}{\alpha_i}$ and $\beta_{ij} \triangleq \beta_j - \frac{\alpha_j}{\alpha_i}\beta_i$. The objective is to synchronize the clocks of all the nodes to a virtual global clock:

$$\bar{\tau}(t) = \bar{\alpha}t + \bar{\beta}.$$

To this aim, construct an estimate of the global clock for each node as follows:

$$\hat{\tau}_i(t) = \hat{\alpha}_i \tau_i(t) + \hat{o}_i. \tag{7.3}$$

A clock synchronization algorithm consists of three steps: relative drift estimation, drift compensation, and offset compensation [248]. In the step of relative drift estimation, each node tries to estimate the relative drifts α_{ij}. Node j sends its current local time $\tau_j(t_l^j)$ to its neighbors, $i \in \mathcal{N}_j$. Node i records in its memory the pair $(\tau_i(t_l^j), \tau_j(t_l^j))$. Each node employs a low-passed filter to estimate α_{ij}, which is given by

$$\begin{aligned}
(\tau_{ij}^{\text{new}}, \tau_j^{\text{new}}) &= (\tau_i(t_l^j), \tau_j(t_l^j)), \\
\eta_{ij}(t^+) &= \rho_\eta \eta_{ij}(t) + (1-\rho_\eta)\frac{\tau_j^{\text{new}} - \tau_j^{\text{old}}}{\tau_{ij}^{\text{new}} - \tau_j^{\text{old}}}, \\
(\tau_{ij}^{\text{old}}, \tau_j^{\text{old}}) &= (\tau_{ij}^{\text{new}}, \tau_j^{\text{new}}), \\
t &= t_l^j, \\
\eta_{ij}(t) &= \eta_{ij}(t^+), \quad t \in (t^+, t_{l+1}^j),
\end{aligned}$$

where $\rho_\eta \in (0,1)$ and t^+ implies the update. The second step is drift compensation, which is used to drive all the nodes to a common clock rate $\bar{\alpha}$. The estimate of $\hat{\alpha}_i$ is given by

$$\hat{\alpha}_i(t^+) = \rho_v \hat{\alpha}(t) + (1-\rho_v)\eta_{ij}(t)\hat{\alpha}_j(t), \quad t = t_l^j, i \in \mathcal{N}_j.$$

The initial values of the estimates are set to $\hat{\alpha}_i(0) = 1$. The third step is offset compensation. Agent i updates the estimated clock offset as follows:

$$\hat{o}_i(t^+) = \hat{o}_i(t) + (1-\rho_0)(\hat{\tau}_j(t) - \hat{\tau}_I(t)), \quad t = t_l^j, i \in \mathcal{N}_j, \tag{7.4}$$

where $\hat{\tau}_j$ and $\hat{\tau}_i$ are computed at $t = t_l^i$ and $\hat{\tau}_i(t_l^i) = \hat{\alpha}_i(t_l^i)\tau_i(t_l^i) + o_i(t_l^i)$. It has been shown that under the proposed algorithm clock synchronization can take place exponentially fast.

7.2.2 Sensor fusion

Sensor fusion by average consensus

Consider the estimation of an unknown constant parameter $\theta \in \mathbb{R}^m$ using a sensor network [249]. The measurement of each node is

$$y_i = A_i\theta + v_i, \quad i = 1, \ldots, n, \tag{7.5}$$

where $y_i \in \mathbb{R}^{m_i}$, A_i is a known matrix, and v_i denotes the random noise, which is mutually independent. The noise vector v_i has zero mean with the covariance matrix Σ_i. The maximum-likelihood (ML) estimate of θ is given by

$$\hat{\theta}_{ML} = \left(\sum_{i=1}^{n} A_i^T \Sigma_i^{-1} A_i\right)^{-1} \sum_{i=1}^{n} A_i^T \Sigma_i^{-1} y_i.$$

The ML estimate is unbiased, i.e., $E(\hat{\theta}_{ML}) = \theta$ with the error covariance matrix

$$Q = \left(A^T \Sigma^{-1} A\right)^{-1}. \tag{7.6}$$

Suppose that each agent has a local composite information matrix $P_i(t)$ and a local composite information state $q_i(t)$ [249], which are initialized as

$$\begin{aligned} P_i(0) &= A_i^T \Sigma_i^{-1} A_i, \\ q_i(0) &= A_i^T \Sigma_i^{-1} y_i, \quad i = 1, \ldots, n. \end{aligned}$$

Each node runs an average consensus algorithm for P_i and q_i, respectively,

$$\begin{aligned} P_i(t+1) &= W_{ii}(t) P_i(t) + \sum_{j=1}^{n} W_{ij}(t) P_j(t), \\ q_i(t+1) &= W_{ii}(t) q_i(t) + \sum_{j=1}^{n} W_{ij}(t) q_j(t). \end{aligned}$$

The weights are chosen as either maximum-degree weights

$$W_{ij}(t) = \begin{cases} \frac{1}{n} & \text{if } \{i,j\} \in \mathcal{E}(t), \\ 1 - \frac{d_i(t)}{n} & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases}$$

where $d_i(t)$ is the degree of node i, or the Metropolis weights

$$W_{ij}(t) = \begin{cases} \frac{1}{1+\max\{d_i(t),d_j(t)\}} & \text{if } \{i,j\} \in \mathcal{E}(t), \\ 1 - \sum_{(i,k)\in\mathcal{E}(t)} W_{ik}(t) & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

It is shown that all the intermediate estimates $\hat{\theta}_i(t)$ are unbiased, i.e, $E\hat{\theta}_i(t) = \theta$ and the covariance matrix at each node converges to the global ML solution (7.6).

Sensor fusion by distributed average tracking

Suppose that each agent has a local measurement $z_i(t) \in \mathbb{R}^m$, which are independent, unbiased noisy observations of a physical variable $p(t) \in \mathbb{R}^m$. Let $W_i(t)$ denote the inverse covariance matrices. The objective is for each agent to track

$$\bar{z} = \left(\sum_i W_i(t)\right)^{-1} \left(\sum_i W_i(t) z_i(t)\right).$$

Consider the following dynamics [250]:

$$\begin{aligned} \dot{x}_i &= \sum_{j=1}^{n} a_{ij}(x_j - x_i) + W_i \dot{z}_i + \dot{W}_i z_i, \\ \dot{M}_i &= \sum_{j=1}^{n} a_{ij}(M_j - M_i) + \dot{W}_i, \\ y_i &= M_i^{-1} x_i, \end{aligned} \tag{7.7}$$

with the following initial conditions

$$\begin{aligned} \dot{x}_i(0) &= W_i(0) z_i(0), \\ M_i(0) &= W_i(0). \end{aligned}$$

It has been shown that as long as the steady states of z_i and W_i exist, the output y_i will track the $\bar{z}$ asymptotically. Note that the first two equations in (7.7) are, respectively, distributed average tracking algorithms for x_i and M_i [55].

7.2.3 Localization

Consider a network of n static wireless sensor nodes [251]. Let $x_i \in \mathbb{R}^D$ denote the position vector of the ith sensor nodes, and let $X = [x_1, \ldots, x_n]$. Assume that some pairs of sensor nodes (i, j) have the following measurements

$$r_{ij} = d_{ij} + \nu_{ij},$$

where $d_{ij} \triangleq \|x_i - x_j\|_2$ and ν_{ij} is an additive noise term with known probability distribution. In particular, let $p_{ij}(d_{ij}(x_i, x_j)|r_{ij})$ denote the inter-sensor sensing PDF. Suppose that there are some anchor nodes, whose positions a_k are known by all the neighboring sensor nodes as

$$v_{ik} = e_{ik} + \mu_{ik},$$

where $e_{ik} \triangleq \|x_i - a_k\|_2$ and $\mu_{i,k}$ is an additive noise term with known probability distribution. Let $p_{ika}(e_{ik}(x_i, a_k)|v_{ik})$ be the anchor-sensor sensing PDF. The localization problem can be written as the following optimization problem

$$X^*_{ML} = \arg\max_{X \in \mathbb{R}^{D \times n}} \Big\{ \sum_{(i,j) \in \mathcal{E}} \ln p_{ij}(d_{ij}(x_i, x_j)|r_{ij}) + \sum_{(i,k) \in \mathcal{E}_a} \ln p_{ika}(e_{ik}(x_i, a_k)|v_{ik}) \Big\}. \tag{7.8}$$

The above problem is non-convex. To transform the problem into a non-convex problem, introduce the following variables $Y = X^T X, \delta_{ij} = d_{ij}^2, \epsilon_{ik} = e_{ik}^2$. Let d, e, δ, ϵ be the stack vectors of d_{ij}, e_{ik}, δ_{ij}, and ϵ_{ik}. Define

$$f(d, e) \triangleq \left(\sum_{(i,j) \in \mathcal{E}} \ln p_{ij}(d_{ij}|r_{ij}) + \sum_{(i,k) \in \mathcal{E}_a} \ln p_{ika}(e_{ik}|v_{ik}) \right).$$

The problem (7.8) can be transformed to the following equivalent form:

$$\begin{aligned}
\text{minimize}_{X,Y,\Delta,\epsilon,d,e} \quad & f(d,e) \\
\text{subject to } & Y_{ii} + Y_{jj} - 2Y_{ij} = \delta_{ij}, \\
& \delta_{ij} = d_{ij}^2, d_{ij} \geq 0, \text{ for all } (i,j) \in \mathcal{E}, \\
& Y_{ii} - 2x_i^T a_k + \|a_k\|_2^2 = \epsilon_{ik}, \\
& \epsilon_{ik} = e_{ik}^2, e_{ik} \geq 0, \text{ for all } (i,k) \in \mathcal{E}_a, \\
& Y = X^T X. \qquad (7.9)
\end{aligned}$$

In (7.9), the function $f_{d,e}$ is convex. However, the constraints define a non-convex set. By using Schur complement, the problem can be further transformed to

$$\begin{aligned}
\text{minimize}_{X,Y,\Delta,\epsilon,d,e} \quad & f(d,e) && (7.10) \\
\text{subject to } & Y_{ii} + Y_{jj} - 2Y_{ij} = \delta_{ij}, && (7.11) \\
& \begin{bmatrix} 1 & d_{ij} \\ d_{ij} & \delta_{ij} \end{bmatrix} \geq 0, d_{ij} \geq 0, \forall (i,j) \in \mathcal{E}, && (7.12) \\
& Y_{ii} - 2x_i^T a_k + \|a_k\|_2^2 = \epsilon_{ik}, \ \epsilon_{ik} \geq 0, && (7.13) \\
& \begin{bmatrix} 1 & e_{ik} \\ e_{ik} & \epsilon_{ik} \end{bmatrix} \geq 0, e_{ik} \geq 0, \forall (i,k) \in \mathcal{E}_a, && (7.14) \\
& \begin{bmatrix} I_D & X \\ X^T & Y \end{bmatrix} \geq 0, Y \geq 0. && (7.15)
\end{aligned}$$

The above problem is convex and its optimal solution provides a lower bound for the original problem. For distributed implementation, the constraint (7.15) is further relaxed, which gives

$$\begin{aligned}
\text{minimize}_{X,Y,\Delta,\epsilon,d,e} \quad & f(d,e) && (7.16) \\
\text{subject to } & (7.11) - (7.14) && (7.17) \\
& \begin{bmatrix} I_D & x_i & x_j \\ x_i^T & Y_{ii} & Y_{ij} \\ x_j^T & Y_{ij} & Y_{jj} \end{bmatrix} \geq 0, \forall (i,j) \in \mathcal{E}. && (7.18)
\end{aligned}$$

The relaxation (7.18) employs the idea of edge-based semi-definite program relaxation [252].

Define the shared vector

$$z_{ij} \triangleq (Y_{ii}, Y_{jj}, Y_{ij}, \delta_{ij}, d_{ij}, x_i^T, x_j^T)^T$$

and the local vector

$$p_i \triangleq (\epsilon_i^T, e_i^T, Y_{ii}, x_i^T)^T.$$

In addition, define

$$Z_{ij} \triangleq \{z_{ij} | z_{ij} \text{ satisfies (7.11), (7.12), (7.18)}\},$$
$$P_i \triangleq \{p_i | p_i \text{ satisfies (7.13), (7.14)}\}.$$

The problem (7.16) can be written compactly as

$$\text{minimize}_{z,p} f_{z,p} \tag{7.19}$$
$$\text{subject to } z_{ij} \in Z_{ij}, \quad \forall (i,j) \in \mathcal{E}, \tag{7.20}$$
$$p_i \in P_i, \quad \forall i \in \mathcal{V},$$

where

$$f(z,p) \triangleq \sum_{i \in \mathcal{V}} f_i(z, p_i) \tag{7.21}$$

with $f_i(z,p_i) \triangleq \sum_{(i,j)\in\mathcal{E}} \ln p_{ij}(d_{ij}|r_{ij}) + \sum_{(i,k)\in\mathcal{E}_a} \ln p_{ika}(e_{ik}|v_{ik})$. Define the local copy of z_{ij} by node i y_{ij}^i. We can rewrite the problem (7.19) equivalently as

$$\text{minimize}_{y_1^1,\ldots,y_n^n,p,z} \quad \sum_{i \in \mathcal{V}} f_i(y_i^i, p_i) \tag{7.22}$$
$$\text{subject to } y_{ij}^i \in Z_{ij}, y_{ij}^j \in Z_{ij}, \quad \forall (i,j) \in \mathcal{E},$$
$$p_i \in P_i, \quad \forall i \in \mathcal{V},$$
$$y_{ij}^i - z_{ij} = 0,$$
$$y_{ij}^j - z_{ij} = 0.$$

An alternating direction method of multipliers (ADMM) [253] is then employed to solve the problem (7.22). Define the regularized Lagrangian of (7.22) as

$$L(y,p,z,\lambda) \triangleq \sum_{i \in \mathcal{V}} f_i(y_i^i, p_i) + \sum_{(i,j)\in\mathcal{E}} \left[\lambda_{ij}^{iT}(y_{ij}^i - z_{ij}) + \lambda_{ij}^{jT}(y_{ij}^j - z_{ij})\right]$$
$$+ \sum_{(i,j)\in\mathcal{E}} \frac{\rho}{2} \left(\|y_{ij}^i - z_{ij}\|^2 + \|y_{ij}^j - z_{ij}\|^2\right) \tag{7.23}$$

where $y = [y_1^{1T}, \ldots, y_n^{nT}]^T$ and λ is vector of the multipliers. The ADMM algorithm is then given by:

$$\begin{aligned}(y^{(t+1)}, p^{(t+1)}) &= \text{argmin}_{y,p} \left\{ L(y, p, z^{(t)}, \lambda^{(t)}) \right\}, \\ z^{(t+1)} &= \text{argmin}_{z} \left\{ L(y^{(t+1)}, p^{(t+1)}, z, \lambda^{(t)}) \right\}, \\ \lambda^{(t+1)} &= \lambda^{(t)} + \rho \nabla_{\lambda} \left[L(y^{(t+1)}, p^{(t+1)}, z^{(t+1)}, \lambda) \right]_{\lambda=\lambda^{(t)}}. \end{aligned} \quad (7.24)$$

It is also shown that the ADMM algorithm (7.24) can be implemented in a distributed way [251].

Ref. [254] describes a consensus-based clock synchronization algorithm that provides internal synchronization to a virtual consensus clock, which is robust to many of the challenges faced in dynamic wireless networks. The algorithm of [218] is based on a PI-like consensus protocol where the proportional part compensates the different clock speeds, and the integral part eliminates the different clock offsets. The distributed algorithm in [255] for clock synchronization is based on an extension of the linear consensus algorithm which can synchronize a family of identical double integrators. Ref. [256] considers the problem of distributed estimation of the poses of N cameras in a camera sensor network using image measurements only. The problem is solved by minimizing a cost function in a distributed fashion using a generalization of the classical consensus algorithm for averaging Euclidean data.

7.3 Smart grids

7.3.1 Economic dispatch

Suppose there are n power generators in a power network. The cost function of a power generator is given by

$$C_i(x_i) = \frac{(x_i - \alpha_i)^2}{2\beta_i} + \gamma_i, \quad (7.25)$$

where x_i is the power generated by generator i, $\alpha_i \leq 0$, $\beta_i > 0$, and $\gamma_i \leq 0$. The cost function (7.26) is equivalent to the one studied in [143]. The economic dispatch problem (EDP) is to solve the following

constrained optimization problem

$$\text{minimize } \sum_{i=1}^{n} C_i(x_i), \tag{7.26}$$

$$\text{subject to } \underline{x}_i \leq x_i \leq \bar{x}_i, \tag{7.27}$$

$$\sum_{i=1}^{n} x_i = D, \tag{7.28}$$

where D is the total demand. We call (7.27) and (7.28) generator constraint and demand constraint, respectively.

Define the incremental cost of the generator i as

$$\frac{dC_i(x_i)}{dx_i} = \frac{(x_i - \alpha_i)}{\beta_i}.$$

It can be observed that the incremental cost is independent of the parameter γ_i. The solution to the EDP is characterized by the following conditions [257]:

$$\begin{cases} \frac{x_i-\alpha_i}{\beta_i} = \lambda^* & \text{for } \underline{x}_i < x_i < \bar{x}_i, \\ \frac{x_i-\alpha_i}{\beta_i} < \lambda^* & \text{for } x_i = \bar{x}_i, \\ \frac{x_i-\alpha_i}{\beta_i} > \lambda^* & \text{for } x_i = \underline{x}_i, \end{cases} \tag{7.29}$$

where λ^* is the optimal incremental cost.

If the generation constraint (7.27) is not considered, when all the generators operate at the optimal configuration, we have

$$\frac{x_i^* - \alpha_i}{\beta_i} = \lambda^*, \ \forall i = 1, \ldots, n. \tag{7.30}$$

The optimal power generation for each generator is

$$x_i^* = \beta_i \lambda^* + \alpha_i, \ \forall i = 1, \ldots, n. \tag{7.31}$$

Let λ_i denote the estimation of optimal incremental cost by generator i, x_i the estimation of optimal power generation, and y_i the estimation of the mismatch between demand and total power generation. Let $\mathcal{G}$ be a directed graph, which describes the communication among the generators. For generator i, its in-neighbors and out-neighbors are denoted, respectively, by N_i^+ and N_i^-. Define two matrices associated

with the graph $\mathcal{G}$:

$$p_{ij} = \begin{cases} \frac{1}{d_i^-} & \text{if } j \in N_i^+ \\ 0 & \text{otherwise.} \end{cases} \tag{7.32}$$

and

$$q_{ij} = \begin{cases} \frac{1}{d_j^-} & \text{if } i \in N_j^- \\ 0 & \text{otherwise.} \end{cases} \tag{7.33}$$

In [258], the authors propose the following algorithm:

$$\begin{aligned} \lambda_i(k+1) &= \sum_{j \in N_i^-} p_{ij}\lambda_j(k) + \epsilon y_i(k), \\ x_i(k+1) &= \beta_i \lambda_i(k+1) + \alpha_i, \\ y_i(k+1) &= \sum_{j \in N_i^-} q_{ij} y_j(k) - [x_i(k+1) - x_i(k)], \end{aligned} \tag{7.34}$$

where ϵ is a sufficiently small positive constant.

When the generation constraint (7.27) is considered, the following projector operators is employed:

$$\phi_i(\lambda_i) = \begin{cases} \bar{x}_i & \text{if } \lambda_i > \bar{\lambda}_i \\ \beta_i \lambda_i + \alpha_i & \text{if } \underline{\lambda}_i \le \lambda_i \le \bar{\lambda}_i \\ \underline{x}_i & \text{if } \lambda_i \le \underline{\lambda}_i, \end{cases} \tag{7.35}$$

where $\underline{\lambda}_i = (\underline{x}_i - \alpha_i)/\beta_i$ and $\bar{\lambda}_i = (\bar{x}_i - \alpha_i)/\beta_i$. The algorithm is

$$\begin{aligned} \lambda_i(k+1) &= \sum_{j \in N_i^-} p_{ij}\lambda_j(k) + \epsilon y_i(k), \\ x_i(k+1) &= \phi_i\Big(\sum_{j \in N_i^-} p_{ij}\lambda_j(k) + \epsilon y_i(k)\Big), \\ y_i(k+1) &= \sum_{j \in N_i^-} q_{ij} y_j(k) - [x_i(k+1) - x_i(k)]. \end{aligned} \tag{7.36}$$

7.3.2 State estimation

State estimation is a key function for energy management systems of smart grids. Consider a multi-area power system. The measurement at

the ith area is

$$z_i = H_i\theta + e_i, \tag{7.37}$$

where H_i is the measurement Jacobian matrix at area i, e_i the associated measurement noise vector, and θ the vector of bus angles (the state to be estimated). The number of measurements in an area is typically much smaller than the dimension of θ. Therefore, it is not possible to obtain the estimate of θ in any area by using their measurement. This observation indicates that all the area should work cooperatively to estimate the state θ. Currently, this is achieved in a centralized manner. The supervisory control and data acquisition (SCADA) system collects all the measurements from the areas, and the estimation task is accomplished by solving the following centralized optimization problem:

$$\begin{aligned} &\text{minimize}_\theta \quad r^T R^{-1} r \\ &\text{subject to } r = z - H\theta, \end{aligned}$$

where $z = [z_1^T, \ldots, z_n^T]^T$ and $R = \text{cov}([e_1, \ldots, e_n]^T) = \text{diag}[R_1, \ldots, R_n]$, and $H^T = [H_1^T, \ldots, H_n^T]^T$.

In [259], the authors propose a distributed iterative scheme which guarantees that each control area converges almost surely to the centralized state θ. Specifically, the algorithm is given by

$$x_i(k+1) = x_i(k) - \left[\beta(k) \sum_{j \in \Omega_i} [x_i(k) - x_j(k)] - \alpha(k) H_i^T (z_i - H_i x_i(k))\right],$$

where Ω_i denotes the communication neighborhood of area i and $\alpha(k)$ and $\beta(k)$ are properly chosen time-varying weights.

Ref. [260] develops a fully distributed recursive least-squares algorithm for wireless sensor networks whereby sensors exchange messages with one-hop neighbors to consent to the network-wide estimates adaptively. Ref. [261] discusses the synchronization problem for the network-reduced model of a power system with nontrivial transfer conductances. Ref. [262] studies the classic slow coherency and area aggregation approach to model reduction in power networks. A time-scale separation and singular perturbation analysis is employed, which leads to a reduced low-order system, where coherent areas are collapsed into aggregate

variables. Ref. [263] presents a novel approach to solve the economic dispatch problem, where by selecting the incremental cost of each generation unit as the consensus variable, the algorithm can solve the conventional centralized control problem in a distributed manner.

7.4 Machine learning

7.4.1 Consensus-based distributed support vector machine

Consider an undirected network $\mathcal{G}$ with the node set $\mathcal{J} \triangleq \{1, \ldots, J\}$ and the edge set $\mathcal{E}$ describing the links between different nodes [264]. Each node $j \in \mathcal{J}$ has a labeled training set $S_j \triangleq \{(x_{jn}, y_{jn}) : n = 1, \ldots, |N_j|\}$, where $|N_j|$ denotes the size of the training set, $x_{jn} \in \mathcal{X}$ is a $p \times 1$ data vector with $\mathcal{X}$ denoting the input space, and $y_{in} \in \mathcal{Y}$ is the corresponding class label with $\mathcal{Y} \triangleq \{-1, 1\}$ being the label set. The objective is to find a maximum-margin linear discriminant function $g(x)$ in a distributed way such that each node is able to classify any new input vector x without communicating S_j to other nodes.

Suppose there is a centralized fusion center. Let x^* and b^* denote the centralized maximum-margin linear discriminant function

$$g^*(x) = x^T w^* + b^*.$$

The values of w^* and b^* are determined by

$$\begin{aligned} \{w^*, b^*\} = \arg\min_{w,b\{\xi_{jn}\}} \frac{1}{2}\|w\|^2 + C\sum_{j=1}^{J}\sum_{n=1}^{|N_j|}\xi_{jn}, \\ \text{subject to } y_{jn}(w^T x_{jn} + b) \geq 1 - \xi_{jn}, \quad \forall j \in \mathcal{J}, n = 1, \ldots, |N_j|, \\ \xi_{jn} \geq 0, \quad \forall j \in \mathcal{J}, n = 1, \ldots, |N_j|, \end{aligned} \tag{7.38}$$

where ξ_{jn} are the slack variables, and $C > 0$ is a tunable constant.

However, in many cases, the existence of such a central unit might not be desired. The reasons include that the cost of building such a central unit might not be affordable, or that the training data is distributed among the nodes and the nodes are not willing to share the their own data due to privacy preservation. Hence, we need to consider the distributed formulation of the linear classifier problem in (7.38) in a

distributed manner. To this end, define the local variables $\{w_j, b_j\}_{j=1}^J$, which are the local copy of the global variable (w, b) in (7.38). We want to add consensus constraint to force these local variables to reach an agreement. The proposed consensus-based distributed formulation of (7.38) is

$$\begin{aligned} \min_{w_j, b_j, \xi_{jn}} & \frac{1}{2} \sum_{j=1}^{J} \|w_j\|^2 + JC \sum_{j=1}^{J} \sum_{n=1}^{|N_j|} \xi_{jn} \\ \text{subject to } & y_{jn}(w_j^T x_{jn} + b_j) \geq 1 - \xi_{jn}, \forall j \in J, n = 2, \ldots, |N_j|, \\ & \xi_{jn} \geq 0, \forall j \in \mathcal{J}, n = 1, \ldots, |N_j|, \\ & w_j = w_i, b_j = b_i, \forall j \in \mathcal{J}, i \in N_j, \end{aligned} \tag{7.39}$$

where N_j denotes the neighborhood of node j. The problem (7.39) is equivalent to (7.38) as long as the network $\mathcal{G}$ is connected. The problem (7.39) can be solved in a distributed manner, since each node optimizes its own variables by exchanging information with its local neighbors. It is shown in [264] that the distributed problem (7.39) can be solved by the alternating direction method of multipliers.

7.4.2 Distributed principal component analysis

Consider a network where each node has n_i measurement vectors

$$Y_i = [y_{i1}, \ldots, y_{in_i}] \in \mathbb{R}^{n \times n_i}.$$

Let $Y = [Y_1, \ldots, Y_N] \in \mathbb{R}^{m \times n}$. Then, principal component analysis can be employed to reduce the dimension of the measurement set Y. We can approximate the vectors using the following affine subspace model

$$y_{ij} \approx \hat{y}_{ij} = \hat{V} c_{ij} + \hat{\mu}, \tag{7.40}$$

where $\hat{y}_{ij}$ is the approximation of y_{ij}, $\hat{\mu}$ is the offset, and $\hat{V}$ is an orthonormal basis for an $\hat{r}$-dimensional space, and c_{ij} are the coefficients for $\hat{y}_{ij}$ in this basis. The objective of principal component analysis is to find $\hat{V}$, c_{ij}, and $\hat{\mu}$ such that total reconstruction error is minimized. The process of principal component analysis mainly consists of three steps [265]:

1. Compute $\hat{\mu}$ as the average of the entire dataset and computer a centered version of the data $\tilde{y}_{ij} = y_{ij} - \hat{\mu}$.

2. Compute the singular value decomposition of the centered data matrix $\tilde{Y} = V\Sigma U^T$ and set $\hat{V}$ as the first $\hat{r}$ columns of V.

3. Compute the coefficients as $c_{ij} = \hat{V}^T \tilde{y}_{ij}$.

In Step 1, the average $\hat{\mu}$ can be computed by a traditional average consensus algorithm. Let

$$C = \frac{1}{N}\tilde{Y}^T\tilde{Y} = \frac{1}{N}\sum_{i=1}^{N} A_i^T A_i = \frac{1}{N}\sum_{i=1}^{N} C_i,$$

where $C_i \triangleq A_i^T A_i$. It thus follows that

$$C = V\left(\frac{1}{N}\Sigma^2\right)V^T.$$

Therefore, in Step 2, each node can compute $\tilde{Y}$ via distributed averaging. Step 3 is local computation. In summary, the process of principal component analysis can be implemented in a fully distributed way.

7.5 Social networks

7.5.1 Degroot model

Consider a group of n individuals, each having its subjective probability distribution of an unknown parameter θ. Let F_i denote individual i's subjective probability distribution. Suppose that individual i is apprised of other individuals' subjective distributions. It is natural for it to revise its subjective distribution F_i to accommodate the information of other individuals. It is common to assume that the revised distribution is a linear combination of the distribution $F_1, \ldots, F_n$ [266]. In particular, let $p_{ij} \geq 0$ denote the weight that individual i assigns to the distribution of individual j. It is assumed that $P = [p_{ij}]$ is a stochastic matrix. This gives the so-called Degroot model:

$$F_i(k+1) = \sum_{j=1}^{n} p_{ij} F_j(k). \tag{7.41}$$

In a compact form, the model (7.41) can be rewritten as

$$F(k+1) = PF(k),$$

where $F(k)$ is the stack vector of F_j, $j = 1, \ldots, n$. It has been shown that if there exists a positive integer n such that every entry in at least one column of the matrix P^n is positive, then an opinion consensus is reached asymptotically [266]. Let $\pi = [\pi_1, \ldots, \pi_n]$ be a stationary distribution vector, i.e., $\pi P = \pi$. It is shown that the final consensus value is $\sum_{i=1}^{n} \pi_i F_i$.

Example 16 ([266]). Suppose that there are two individuals in the group, i.e., $n = 2$. Suppose that

$$P = \begin{bmatrix} 1/2 & 1/2 \\ 1/4 & 3/4 \end{bmatrix}. \tag{7.42}$$

Individual 1 assigns equal weights to its own distribution and individual 2's distribution. Individual 2 is more confident with itself and assigns three times weight to its own distribution. The stationary probability distribution vector is $\pi = (1/3, 2/3)$. Therefore, the final consensus value is $(1/3)F_1(0) + (2/3)F_2(0)$.

7.5.2 Friedkin–Johnsen model

One limitation of DeGroot model is that the agents' behavior does not change. For instance, if the interaction graph among the individuals is connected, then a consensus will always take place. Friedkin–Johnsen model [267] addresses the limitation of DeGroot model by allowing each agent to have different susceptibilities to persuasion. Friedkin–Johnsen model is given by

$$y(t+1) = AWy(t) + (I - A)y(0), \tag{7.43}$$

where $y(t) = [y_1(t), \ldots, y_n(t)]^T$ denotes individuals' opinions at time t, $W = [w_{ij}]$ is an $n \times n$ matrix of interpersonal influences satisfying $0 \leq w_{ij} \leq 1, \sum_{j=1}^{n} w_{ij} = 1$, and $A = \text{diag}(a_{11}, \ldots, a_{nn})$ is a diagonal matrix describing individuals' susceptibilities to interpersonal influence on the issue. Applying (7.43) iteratively, we have

$$y(t+1) = V(t)y(0), \tag{7.44}$$

where

$$V(t) = (AW)^t + \sum \left[\sum_{k=1}^{t-1} (AW)^k \right] (I - A).$$

Assume that an equilibrium exists, i.e., $\lim_{t\to\infty} y(t) = y(\infty)$ exists. Then, it follows from (7.43) that

$$y(\infty) = AWy(\infty) + (I - A)y(0),$$

and hence

$$(I - AW)y(\infty) = (I - A)y(0). \tag{7.45}$$

If $I - AW$ is invertible, we have

$$y(\infty) = Vy(0),$$

where

$$V = \lim_{t\to\infty} V(t) = (I - AW)^{-1}(I - A).$$

Example 17. [267] Consider a group of four individuals. Each is presented with an issue on which opinions could range from 1 to 100. The initial opinions are given by

$$y(0) = [25, 25, 75, 85]^T.$$

The weight matrix W is

$$W = \begin{bmatrix} .220 & .120 & .359 & .300 \\ .147 & .215 & .344 & .294 \\ 0 & 0 & 1 & 0 \\ .089 & .178 & .446 & .286 \end{bmatrix}, \tag{7.46}$$

which describes the distribution of relative interpersonal influences on the issue. The matrix $A = \text{diag}(.780, .785, 0, .714)$. Hence, the matrix V is given by

$$V = \begin{bmatrix} .280 & .045 & .551 & .124 \\ .047 & .278 & .549 & .126 \\ 0 & 0 & 1 & 0 \\ .030 & .048 & .532 & .390 \end{bmatrix}.$$

The predicted final opinions are

$$y(\infty) = Vy(0) = [60, 60, 75, 75]^T. \tag{7.47}$$

7.5.3 Polar opinion dynamics

In [268], the authors introduce the following general model of polar opinion dynamics:

$$\dot{x} = -A(x)Lx, \tag{7.48}$$

where $x \in [-1, 1]^n$ represent the agents' states, $A(x(t)) \in \text{diag}([0, 1]^n)$ is a diagonal matrix with the diagonal entry denoting agents' susceptibility functions, $L = I - W$ is the Laplacian matrix, and $W = [w_{ij}] \in [0, 1]^{n\times n}$ is the row stochastic matrix with w_{ij} denoting the amount of relative influence of agent j upon agent i. Polar opinion dynamics describes either degrees of proclivity toward one of two competing alternatives. The model (7.48) consists of two components: the averaging component $-Lx$ driving the agents towards a consensus and the susceptibility component $A(x)$ impeding the convergence process.

Under the general model (7.48), by choosing different susceptibility matrix $A(x(t))$, we have three specialized models [268].

- Model with stubborn extremists: Let $A(x) = (I - \text{diag}(x)^2)$. We have the following model

$$\dot{x} = -(I - \text{diag}(x)^2)Lx. \tag{7.49}$$

  The model (7.49) assumes that extreme opinions are more resistant to change than neutral opinions.

- Model with stubborn positives: Let $A(x) = \frac{1}{2}(I - \text{diag}(x))$. We have the following model

$$\dot{x} = -\frac{1}{2}(I - \text{diag}(x))Lx. \tag{7.50}$$

  The model (7.50) assumes that only the individuals at one end of the opinion spectrum are stubborn.

- Model with stubborn neutrals: Let $A(x) = \text{diag}(x)^2$. We have the following model

$$\dot{x} = -\text{diag}(x)^2 Lx, \tag{7.51}$$

  which assumes that the neutral opinion 0 corresponds to a social norm, and the agents are not willing to deviate form it.

For the model (7.50), it is shown that if $x(0) < \mathbf{1}$ [1], then $\lim_{t\to\infty} x(t) = \alpha\mathbf{1}$, where $\alpha \in [0, 1)$; if there exists some i such that $x_i(0) = 1$, then $\lim_{t\to\infty} x(t) = \mathbf{1}$. For the model (7.51), it is shown that if $x(0) > 0$, then $\lim_{t\to\infty} x(t) = \alpha\mathbf{1}$, $\alpha \in (0, 1]$; if $x(0) < 0$, then $\lim_{t\to\infty} x(t) = \alpha\mathbf{1}$, $\alpha \in [-1, 0)$; otherwise, $\lim_{t\to\infty} x(t) = 0$. For the model (7.49), let

$$I_{\text{open}} = \{i \mid A_{ii}(x(0)) > 0\},$$
$$I_{\text{closed}} = \{i \mid A_{ii}(x(0)) = 0\},$$

It is shown that if $I_{\text{closed}} = \emptyset$, then $\lim_{t\to\infty} = \alpha\mathbf{1}$; if $I_{\text{closed}} \neq \emptyset$, yet, $\forall i, j \in I_{\text{closed}}$, $x_i = x_j = \alpha$, then $\lim_{t\to\infty} = \alpha\mathbf{1}$; if $I_{\text{closed}} \neq \emptyset$ and $\exists i, j \in I_{\text{closed}} : x_i \neq x_j$,

$$\lim_{t\to\infty} x(t) = x^* = [(x_c^*)^T, (x_o^*)^T]^T,$$

with $x_c^* = x_c(0)$ the initial state of the closed agents, and $x_o^* = (I - W_{oo})^{-1} W_{oc} x_o^*$.

Ref. [269] provides an overview of recent research on belief and opinion dynamics in social networks, where both Bayesian and non-Bayesian models of social learning are discussed. Ref. [270] gives an affirmative answer to the question "is it possible to achieve a form of the agreement also in the presence of antagonistic interactions, modeled as negative weights on the communication graph?". Necessary and sufficient conditions are obtained to describe the cases in which this is possible. Ref. [271] develops a dynamic model of opinion formation in social networks, where the information required for learning a parameter may not be at the disposal of any single agent. Ref. [272] provides a model to investigate the tension between information aggregation and spread of misinformation, where individuals meet pairwise and exchange information.

7.6 Task migration of many-core microprocessors

Task migration policy is the most important part of the thermal management system for many-core microprocessors. In [273], the authors proposed a distributed task migration method based on the distributed

[1] The inequality $x(0) < \mathbf{1}$ means that $x_i(0) < 1$ for all i.

average tracking algorithm in [57]. The distributed average tracking (DAT) algorithm is used to track the average of the temperatures of all the cores, which is given by

$$\dot{z}_i = \alpha \sum_{j=1}^{n} a_{ij} \text{sgn}[x_j - x_i], \tag{7.52}$$

$$x_i = z_i + T_i \tag{7.53}$$

where T_i is the temperature of core i, z_i is an auxiliary variable, x_i is the state, and sgn denotes the signum function. It is shown in [57] that if α is properly chosen, all the cores can track the averaged temperature in a finite time, i.e., $x_i \to \frac{1}{n}\sum_{j=1}^{n} T_j$.

In the proposed approach, the task migration policy will be activated for core i if the temperature of core i is above the average temperature. Define the current core as the one to which task migration is applied, and the destined core as the one that could potentially exchange task with the current core. The task migration policy will be activated for a processor core if the temperature of the core is above the average temperature T_{avg}. Task migration happens only when the following conditions are met [273]:

- $T_{\text{des}} < T_{\text{cur}}$, where T_{des} is the temperature of the destined core and T_{cur} is the temperature of the current core.
- $P_{\text{des}} < P_{\text{cur}}$, where P_{des} is the task load of the destined core, and P_{cur} is the one for the current core.
- $TNP_{\text{des}} < TNP_{\text{cur}}$, where TNP_{des} is the total task load including the immediate neighboring cores of the destined core, and TNP_{cur} is the one of the current core.

The proposed distributed task migration policy is illustrated in Fig. 7.1, where the current node is node 0 and its four neighbors are labeled as 1–4. The experimental results on a 36-core microprocessor demonstrate that the proposed task migration method can reduce 30% more thermal hot spots compared with the existing approaches.

The DAT-based task migration policy takes into account the average temperature of all the cores, instead of arbitrarily choosing a fixed

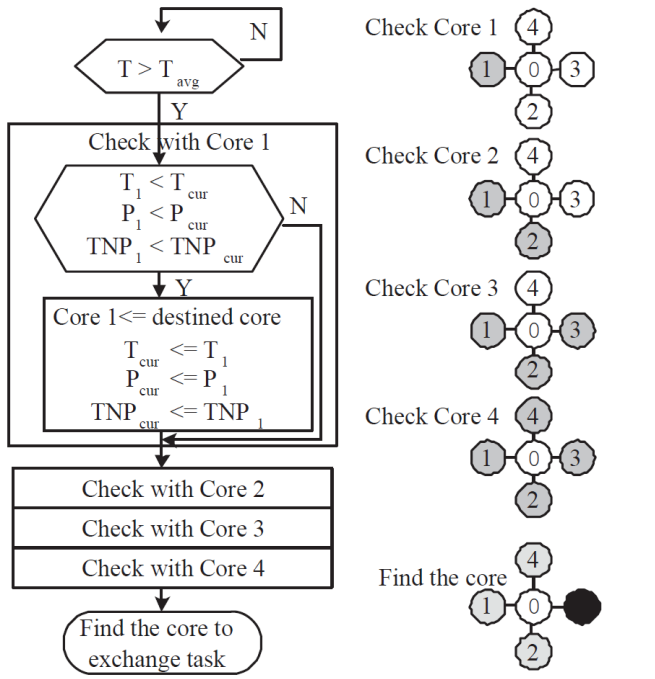


Figure 7.1: The proposed task migration policy in [273].

temperature threshold. It is used in the distributed dynamic thermal management (DDTM) system [274, 275], which is suitable for the next-generation many-core systems.

7.7 Coordination of the charging of multiple electric vehicles

Suppose that there are multiple buildings (see Fig. 7.2), equipped with both wind power (the cost is negligible) and power grid [276]. There are also a number of electric vehicles (EVs), driven among the buildings and parked in the buildings. The objective is to minimize the charging cost, while satisfying the travel demand of the EVs. Define the state of EV n as $S_t^n = [L_t^n, E_t^n, D_t^n]$. Here, E_t^n denotes the reaming charging energy to support its next trip; D_t^n denotes the location, and

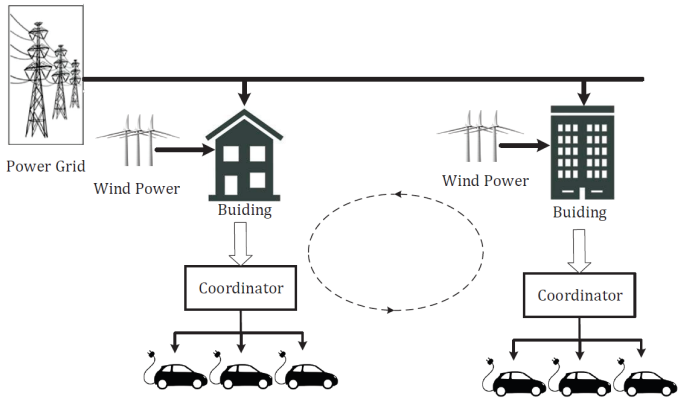


Figure 7.2: Illustration of the problem [276].

$D_t^n \in \mathcal{D} = \{0, 1, \ldots, M\}$, where $1 \sim M$ correspond to M buildings and $D_t^n = 0$ indicates that EV n is on travel at time t; L_t^n represents the remaining parking time ($D_t^n > 0$) or remaining trip time ($D_t^n = 0$). Let $A_t = [a_t^n](n = 1, \ldots, N)$ be the action space, where $a_t^n = 1$ if EV n is selected to get charged at time t; $a_t^n = 0$ otherwise.

The remaining parking time or trip time is govern by

$$L_{t+1}^n = \begin{cases} L_t^n - 1 & \text{if } L_t^n > 0 \\ \tau_{t+1}^n & \text{if } L_t^n = 0, D_t^n = 0 \\ \eta_{t+1}^n & \text{if } L_t^n = 0, D_t^n > 0, \end{cases}$$

where τ_{t+1}^n represents the parking time, and η_{t+1}^n denotes the trip time. The location transition is determined by

$$D_{t+1}^n = \begin{cases} D_t^n & \text{if } L_t^n > 0 \\ R_{t+1}^n & \text{if } L_t^n = 0, D_t^n = 0 \\ 0 & \text{if } L_t^n = 0, D_t^n > 0, \end{cases}$$

where R^n_{t+1} denotes the building of EV n at next time. The dynamics of the remaining charging energy is given by

$$E^n_{t+1} = \begin{cases} E^n_t - a^n_t P \Delta_t & \text{if } L^n_t \geq 0, D^n_t > 0 \\ E^n_t + f_n(\eta^n_{t+1}) & \text{if } L^n_t = 0, D^n_t = 0 \\ E^n_t & \text{if } L^n_t > 0, D^n_t = 0, \end{cases}$$

where P represents the charging rate and f_n is the energy consumption.

There are several constraints need to be taken into account. First, the remaining charging energy cannot exceed the maximum possible charging energy during the remaining parking time, yielding

$$E^n_t - a^n_t P \Delta_t \leq (L^n_t - 1) P \Delta_t.$$

Second, the remaining charging energy cannot exceed battery capacities, yielding

$$E^n_t \leq E_{\text{cap}}.$$

Third, the charing processes of EVs are constrained by their parking durations, i.e.,

$$a^n_t \leq D^n_t.$$

The one-step cost for the buildings is

$$C_t(S_t, A_t) = \sum_{j=1}^{M} c_t \max(P^j_t - W^j_t, 0) \Delta_t,$$

where c_t denotes the price of power grid. The total cost is given by

$$\min J(\pi, S_1) = E \left\{ \sum_{t=1}^{T} \sum_{j=1}^{M} c_t \max(P^j_t - W^j_t, 0) \Delta t \right\}.$$

It is shown that the above problem can be solved via a tree-based dynamic programming strategy.

7.8 Distributed heating, ventilation, and air-conditioning (HVAC) optimization

Consider multiple buildings $b = 1, \ldots, N$, each of which has multiple zones, $i = 1, \ldots, n^b$. Suppose that only the zones, $i = 1, \ldots, n^b_1$ are

equipped with VAV HVAC systems. The thermal dynamics of the building b is given by

$$C^b\dot{T}^b = R^bT^a\mathbf{1} - (R^b + L^b)T^b + B^bG^b(T^b)m^b + B^bq^b, \tag{7.54}$$

where T^b is the temperature of the building b, T^a is the ambient temperature, m^b is the mass flow rate, q^b is the heat gain from external sources. The linearized and transformed model of (7.54) is given by

$$\dot{x}^b = -A^bx^b + B^bu^b + B^bw_q^b + w_a^b,\ x^b = \begin{bmatrix} x_1^b \\ x_2^b \end{bmatrix},$$

where x^b is the transformed variable, A^b is a positive definite matrix, w_q^b is the vector corresponding to the heat gain, and w_a^b is the vector corresponding to the ambient temperature.

The objective of HVAC optimization is to solve the following problem [277]

$$\begin{aligned} \min_{z^1,\ldots,z^N} & \sum_{b=1}^{N}\{\frac{1}{2}\|z_x^b - h^b\|^2 + f^b(z_u^b)\} \\ \text{s.t. } & g^b(z_u^b) \le 0, \\ & -A^bz_x^b + B^bz_u^b + B^bd_q^b + d_a^b = 0,\ b = 1,\ldots,N, \\ & g^0(z_u^1,\ldots,z_u^N) = \sum_{b=1}^{N}\bar{f}^b(z_u^b) - \gamma \le 0. \end{aligned}$$

The first term in the objective function evaluates the human comfort, while the second term evaluates the power consumption; $g^b(\cdot)$ represents hardware constraint on the mass flow rate; the equality constraint is the thermal dynamic equation at equilibrium; the last one gives the constraint on power consumption. By eliminating z_x^b, the optimization

model reduces to

$$\begin{aligned}
&\min_{z_u^1,\ldots,z_u^N} \sum_{b=1}^{N} \{\phi^b(z_u^b) + f^b(z_u^b)\} \\
&\text{s.t.} \quad g^b(z_u^b) \le 0,\ b = 1\ldots, N, \\
&\qquad g^0(z_u^1,\ldots,z_u^N) = \sum_{b=1}^{N} \bar{f}^b(z_u^b) - \gamma \le 0, \\
&\qquad \phi^b(z_u^b) = \frac{1}{2}\|M^b(z_u^b + d_q^b) + (B^b)^T(A^b)^{-1}(d_a^b - \bar{h}^b)\|^2
\end{aligned}$$

where

$$M^b = (B^b)^T(A^b)^{-1}B^b, \quad \bar{h}^b = A^bB^bh^b.$$

The problem is a resource allocation problem with separable constraints and cost.

Design the following primal-dual gradient algorithm [278] to solve the above problem

$$\begin{aligned}
\dot{\hat{z}}_u^b &= -\alpha\{(M^b)^2(\hat{z}_u^b + w_q^b) + K^b(w_a^b - \bar{h}^b) + \nabla f^b(\hat{z}_u^b) \\
&\quad + \nabla g^b(\hat{z}_u^b)\hat{\lambda}^b + p_b^0\}, \\
\dot{\hat{\lambda}}^b &= [g^b(\hat{z}_u^b)]_{\hat{\lambda}^b}^+, \quad b = 1,\ldots,N, \\
\dot{\hat{\lambda}}^0 &= [g^0(\hat{z}_u^1,\ldots,\hat{z}_u^N)]_{\hat{\lambda}^0}^+.
\end{aligned}$$

In practice, d_q^b, the vector corresponding to the heat gain, is updated in real time, which gives

$$\begin{aligned}
\dot{\hat{z}}_u^b &= -\alpha\{(M^b)^2(\hat{z}_u^b + \hat{w}_q^b) + K^b(w_a^b - \bar{h}^b) + \nabla f^b(\hat{z}_u^b) \\
&\quad + \nabla g^b(\hat{z}_u^b)\hat{\lambda}^b + p_b^0\}.
\end{aligned}$$

Define the input $v_0^b = M^b\hat{w}_q^b$, one has the following passive system

$$\begin{aligned}
\dot{\hat{z}}_u^b &= -\alpha\{(M^b)^2\hat{z}_u^b + M^bv_0^b + K^b(w_a^b - \bar{h}) + \nabla f^b(\hat{z}_u^b) \\
&\quad + \nabla g^b(\hat{z}_u^b)\hat{\lambda}^b + p_b^0\}, \\
\dot{\hat{\lambda}}^b &= [g^b(\hat{z}_u^b)]_{\hat{\lambda}^b}^+, \\
y_0^b &= M^b\hat{z}_u + v_0^b + (B^b)^T(A^b)^{-1}w_a^b.
\end{aligned}$$

Design the following controller to let x_1^b track v_p^b

$$\begin{aligned}\dot{\xi}^b &= k_1^b(v_p^b - x_1^b),\\ u^b &= k_P^b(v_p^b - x_1^b) + \xi^b + \kappa^b v_p^b + F^b w_a^b.\end{aligned}$$

The closed-loop system is

$$\begin{aligned}\dot{x}^b &= -A^b x^b + k_P^b B^b(v_P^b - x_1^b) + B^b\xi^b - \kappa^b B^b v_P^b + B^b w_q^b\\ &\quad + (B^b F^b + I_{n^b})w_a^b,\\ \dot{\xi}^b &= k_1^b(v_P^b - x_1^b).\end{aligned}$$

The output signal is given by

$$y_p^b = \{(k_P^b + 2\kappa^b)M^b - I_{n_1^b}\}v_p^b - (k_P^b + \kappa^b)M^b x_1^b + M^b\xi^b,$$

which gives an estimation of the heat gain.

8

Conclusions and prospects

In this paper, we have attempted, from a control perspective, to give the reader an overview for the developments, along with preliminaries, of MASs in the past two decades. As a survey paper, the primary focus of this paper is to present the most fundamental results of MASs. Some interesting topics, e.g., distributed network games [279–281], are not included. Following the tradition of control theory, we have adopted a mathematical language in presenting the main results. One thing that is hidden behind those mathematical results is the optimism and excitement that we feel for the future of MASs, partially due to the tremendous innovation and developments that have been made and are being made. The research for multi-agent systems has reached one plateau with the ubiquitousness of intelligent sensors, UAVs, and UGVs that are encountered more and more in our daily life, and we believe that MASs will continue to play a significant role in future civilian and military arenas, where its crossing with other disciplines will spark new ideas and new methodologies. Nevertheless, it is noteworthy that there are several unsolved issues in MASs whose studies are still in their infancy, and we dedicate this chapter to some of them and make a quick and possibly biased assessment.

One question that is often come up with is "will there be fully autonomous MASs?". Although it is elusive to give a precise definition of "fully autonomous", there is no doubt that current MASs encountered in the engineering world are all partially autonomous, where the knowledge of human beings is deeply involved, directly or indirectly, in order to design the system, e.g., specify the interaction rules of agents. This is hardly surprising, since the evolution of a technology is indeed a process of reducing the dependence on humans. It is arguable that the role of humans in MASs would be replaced with intelligent agents in the future.

The above question is also pertinent to the term "fully distributed", which has been used rather frequently in the MAS literature. The term defines a MAS of which each agent only employs local information. We maintain that without full autonomy, it is seldom possible to build a "fully distributed" MAS, since, as a matter of fact, the knowledge from humans is global. Let us recall the most fundamental consensus algorithm for continuous-time systems:

$$\dot{x}_i = \sum_{j=1}^{n} a_{ij}(x_j - x_i), \quad i = 1, \ldots, n, \tag{8.1}$$

where x_i is the state of agent i, a_{ij} is the (i, j)th entry of the adjacency matrix, and n is the number of agents. Even though the above consensus algorithm is claimed to be fully distributed in most, if not all, literature, there exists global information behind (8.1), e.g., the agents all follow a linear interaction protocol.

Although much progress has been reported for MASs, results are still lacking to overcome some challenges raised by MASs. In the rest of the chapter, we highlight some of them.

Adaptation of conventional control notions for MASs which highlights the role of network topologies

Compared with traditional control systems, MASs introduce another fundamental control element—the network topology. One primary objective of MAS study is to figure out the role of network topologies in the control of MASs. One would expect the results to be in purely graphic

languages, such that they are not coupled with other perspectives of the systems, e.g., agent dynamics. For simple problems, e.g., consensus, this is possible; nevertheless, for more involved problems, existing techniques and means generally fail to work. One example is the controllability problem, which is notably one of the most fundamental problems for control systems. In linear systems theory, there exist several necessary and sufficient conditions for controllability. However, a direct application of the results to MASs with a generic network topology will fail in most cases and does not lead to very useful results. In [282], the authors proposed a controllability notion, called structural controllability. To some extent, structural controllability can be viewed as a "weakened" version of the conventional controllability notion, but it incorporates the network topologies in its definition, which opens the possibility to derive meaningful results [283, 284] with purely graphic languages.

Global stability for a generic rigid formation in the three-dimensional space

In rigid formation control, rigid graph theory plays a central role, where agents are modeled as nodes, and inter-agent distances are modeled as weighted edges. Roughly speaking, if the smooth motions of a formation are those corresponding to translation and rotation, we say the formation is rigid. A precise definition for rigid formation can be found in [285, 286]. Global rigidity have been well established for two-dimensional formations; however, its extension to the three-dimensional space is not straightforward—there remains a huge gap between two-dimensional and three dimensional rigid formations. Meanwhile, it is of great interest to generalize Laman's theorem [287–289] to the three-dimensional space.

The human factors in MASs

Many potential applications of MASs require operation in uncertain and highly dynamic environments. It thus becomes imperative to include a certain degree of human control in order to tackle complex scenarios that are beyond the capability of agents. While the model of agents is thoroughly investigated, a well-developed mathematical model for

human factors is still lacked. This requires clear definitions of the input, output, and processing logics for humans, which could be extremely hard, because we have to consider all the possibilities in uncertain environments. Nevertheless, there are good reasons to do so.

1. The computational capability of humans is limited. Their responses to sensor data might be slow or even wrong.
2. MASs might work in hazardous environments, which might not be suitable for humans to stay.
3. In some scenarios, there exist non-negligible communication delays between humans and agents, e.g., cooperative robotic arms work together to repair satellites in space.

If the aforementioned issue is solved, it might become possible to replace the role of humans with an intelligent agent, which will eventually lead to fully autonomous and distributed MASs without direct intervention from humans.

Combination of control and learning

Control techniques, such as PID or model predictive control, rely heavily on intricate, explicit agent models to be successful. They are crucial for safety-critical MASs, for which the robustness to uncertainty and disturbance is typically insured by a model-based design approach. In learning control, the system designer does not need to build a precise model for the agents; instead, the designer employs a large amount of data to generate an abstract system model, which is not necessarily described by differential or difference equations. This Lazymans approach is desirable in the scenario where the agent dynamics or the environment is too complicated to be modeled. Learning techniques have achieved the state-of-the-art performance for a variety of multi-agent applications [290–292], particularly when computer vision is involved. For MASs, some sort of hybrid schemes is essential in order to fuse and process the vast amounts of sensor data collected by agents into timely and agile control actions. It is noteworthy that while substantial progress has been reported in classical control as well as in learning-based control,

critical fundamental questions to fill the gap between these two control schemes remain poorly understood. Consequently, great efforts are to be made to leveraging control design with learning techniques in the MAS context.

With these unsolved problems, we believe that MASs will continue to be a promising arena for control theorists and engineers.

References

[1] E. Eisenberg and D. Gale, "Consensus of subjective probabilities: The pari-mutuel method," *The Annals of Mathematical Statistics*, vol. 30, no. 1, pp. 165–168, 1959.

[2] J. Tsitsiklis and M. Athans, "Convergence and asymptotic agreement in distributed decision problems," *IEEE Transactions on Automatic Control*, vol. 29, no. 1, pp. 42–50, 1984.

[3] J. Tsitsiklis, "Problems in decentralized decision making and computation," *PhD, Massachusetts Institute of Technology*, 1984.

[4] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4. ACM, 1987, pp. 25–34.

[5] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, "Novel type of phase transition in a system of self-driven particles," *Physical Review Letters*, vol. 75, no. 6, p. 1226, 1995.

[6] R. M. Murray, "Recent research in cooperative control of multivehicle systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, no. 5, pp. 571–583, 2007.

[7] V. Gazi and B. Fidan, "Coordination and control of multi-agent dynamic systems: Models and approaches," in *International Workshop on Swarm Robotics*. Springer, 2006, pp. 71–102.

[8] P. Y. Chebotarev and R. P. Agaev, "Coordination in multiagent systems and laplacian spectra of digraphs," *Automation and Remote Control*, vol. 70, no. 3, pp. 469–483, 2009.

[9] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial informatics*, vol. 9, no. 1, pp. 427–438, 2013.

[10] F. Garin and L. Schenato, "A survey on distributed estimation and control applications using linear consensus algorithms," in *Networked Control Systems*. Springer, 2010, pp. 75–107.

[11] J. Shamma, *Cooperative control of distributed multi-agent systems*. John Wiley & Sons, 2008.

[12] W. Ren and R. W. Beard, *Distributed consensus in multi-vehicle cooperative control*. Springer, 2008.

[13] S. Butenko, R. Murphey, and P. M. Pardalos, *Cooperative control: models, applications and algorithms*. Springer Science & Business Media, 2013, vol. 1.

[14] F. Bullo, J. Cortes, and S. Martinez, *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press, 2009.

[15] Z. Qu, *Cooperative control of dynamical systems: applications to autonomous vehicles*. Springer Science & Business Media, 2009.

[16] H. Bai, M. Arcak, and J. Wen, *Cooperative control design: a systematic, passivity-based approach*. Springer Science & Business Media, 2011.

[17] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

[18] W. Ren and Y. Cao, *Distributed coordination of multi-agent networks: emergent problems, models, and issues*. Springer Science & Business Media, 2010.

[19] F. L. Lewis, H. Zhang, K. Hengster-Movric, and A. Das, *Cooperative control of multi-agent systems: optimal and adaptive design approaches*. Springer Science & Business Media, 2013.

[20] W. Yu, G. Wen, G. Chen, and J. Cao, *Distributed cooperative control of multi-agent systems*. John Wiley & Sons, 2017.

[21] Z. Li and Z. Duan, *Cooperative control of multi-agent systems: a consensus region approach*. CRC Press, 2014.

[22] H. Su and X. Wang, *Pinning control of complex networked systems: Synchronization, consensus and flocking of networked systems via pinning*. Springer Science & Business Media, 2013.

[23] W. Ren, R. W. Beard, and E. M. Atkins, "Information consensus in multivehicle cooperative control," *IEEE Control Systems*, vol. 27, no. 2, pp. 71–82, 2007.

[24] S. Martinez, J. Cortes, and F. Bullo, "Motion coordination with distributed information," *IEEE Control Systems*, vol. 27, no. 4, pp. 75–88, 2007.

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

[26] G. Antonelli, "Interconnected dynamic systems: An overview on distributed control," *IEEE Control Systems Magazine*, vol. 33, no. 1, pp. 76–88, 2013.

[27] D. A. Paley, N. E. Leonard, R. Sepulchre, D. Grunbaum, and J. K. Parrish, "Oscillator models and collective motion," *IEEE Control Systems*, vol. 27, no. 4, pp. 89–105, 2007.

[28] N. E. Leonard, D. A. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. E. Davis, "Collective motion, sensor networks, and ocean sampling," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 48–74, 2007.

[29] W. Ren, "Averaging algorithms and consensus," *Encyclopedia of Systems and Control*, pp. 55–64, 2015.

[30] J. Cortés, "Networked systems," *Encyclopedia of Systems and Control*, pp. 849–853, 2015.

[31] M. Reza Davoodi, Z. Gallehdari, I. Saboori, H. Rezaee, E. Semsarkazerooni, N. Meskin, F. Abdollahi, and K. Khorasani, "An overview of cooperative and consensus control of multiagent systems," *Wiley Encyclopedia of Electrical and Electronics Engineering*, 2016.

[32] A. V. Proskurnikov and M. Cao, "Consensus in multi-agent systems," *Wiley Encyclopedia of Electrical and Electronics Engineering*, 2016.

[33] F. Chen and W. Ren, "Distributed consensus in networks," *Wiley Encyclopedia of Electrical and Electronics Engineering*, 2016.

[34] A. Torreño, E. Onaindia, and V. Botti, "Planning and coordination in multiagent environments," *Wiley Encyclopedia of Electrical and Electronics Engineering*.

[35] A. Jadbabaie, "Flocking in networked systems," *Encyclopedia of Systems and Control*, pp. 458–463, 2015.

[36] J. P. Hespanha and A. R. Mesquita, "Networked control systems: estimation and control over lossy networks," *Encyclopedia of Systems and Control*, pp. 842–849, 2015.

[37] S. Yüksel, "Information and communication complexity of networked control systems," *Encyclopedia of Systems and Control*, pp. 560–567, 2015.

[38] L. Bushnell and H. Ye, "Networked control systems: architecture and stability issues," *Encyclopedia of Systems and Control*, pp. 835–842, 2015.

[39] M. Mesbahi and M. Egerstedt, "Graphs for modeling networked interactions," *Encyclopedia of Systems and Control*, pp. 510–514, 2015.

[40] A. Nedić, "Distributed optimization," *Encyclopedia of Systems and Control*, pp. 308–317, 2015.

[41] B. A. Francis, "Oscillator synchronization," *Encyclopedia of Systems and Control*, pp. 1015–1020, 2015.

[42] A. D. Domínguez-García and C. N. Hadjicostis, "Coordination of distributed energy resources for provision of ancillary services: Architectures and algorithms," *Encyclopedia of Systems and Control*, pp. 241–246, 2015.

[43] E. Frazzoli and M. Pavone, "Multi-vehicle routing," *Encyclopedia of Systems and Control*, pp. 821–830, 2015.

[44] R. Srikant, "Network games," *Encyclopedia of Systems and Control*, pp. 831–835, 2015.

[45] P. Antsaklis and J. Baillieul, "Guest editorial special issue on networked control systems," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1421–1423, 2004.

[46] ——, "Special issue on technology of networked control systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 5–8, 2007.

[47] T. Arai, E. Pagello, and L. E. Parker, "Advances in multi-robot systems," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 655–661, 2002.

[48] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[49] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[50] W. Ren and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 655–661, 2005.

[51] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 169–182, 2005.

[52] Z. Lin, M. Broucke, and B. Francis, "Local control strategies for groups of mobile autonomous agents," *IEEE Transactions on Automatic Control*, vol. 49, no. 4, pp. 622–629, 2004.

[53] S. Mastellone, J. S. Mejía, D. M. Stipanović, and M. W. Spong, "Formation control and coordinated tracking via asymptotic decoupling for lagrangian multi-agent systems," *Automatica*, vol. 47, no. 11, pp. 2355–2363, 2011.

[54] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401–420, 2006.

[55] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Dynamic consensus on mobile networks," in *IFAC World Congress*. Prague Czech Republic, 2005, pp. 1–6.

[56] H. Bai, R. A. Freeman, and K. M. Lynch, "Robust dynamic average consensus of time-varying inputs," in *49th IEEE Conference on Decision and Control (CDC)*. IEEE, 2010, pp. 3104–3109.

[57] F. Chen, Y. Cao, W. Ren *et al.*, "Distributed average tracking of multiple time-varying reference signals with bounded derivatives," *IEEE Transactions on Automatic Control*, vol. 57, no. 12, pp. 3169–3174, 2012.

[58] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[59] A. Nedic, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.

[60] J. Ferber and G. Weiss, *Multi-agent systems: an introduction to distributed artificial intelligence*. Addison-Wesley, Reading, 1999, vol. 1.

[61] "Generation robots," https://www.generationrobots.com, 2018.

[62] P. J. Antsaklis, K. M. Passino, and S. Wang, "An introduction to autonomous control systems," *IEEE Control Systems*, vol. 11, no. 4, pp. 5–13, 1991.

[63] F. Chen, L. Xiang, W. Lan, and G. Chen, "Coordinated tracking in mean square for a multi-agent system with noisy channels and switching directed network topologies," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, no. 11, pp. 835–839, 2012.

[64] D. Lee and M. W. Spong, "Stable flocking of multiple inertial agents on balanced graphs," *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1469–1475, 2007.

[65] T. Rodseth, *Models for multispecies management*. Springer Science & Business Media, 2012.

[66] G. Ferrari-Trecate, L. Galbusera, M. P. E. Marciandi, and R. Scattolini, "Model predictive control schemes for consensus in multi-agent systems with single-and double-integrator dynamics," *IEEE Transactions on Automatic Control*, vol. 54, no. 11, pp. 2560–2572, 2009.

[67] G. S. Seyboth, D. V. Dimarogonas, and K. H. Johansson, "Event-based broadcasting for multi-agent average consensus," *Automatica*, vol. 49, no. 1, pp. 245–252, 2013.

[68] A. Abdessameud and A. Tayebi, "On consensus algorithms design for double integrator dynamics," *Automatica*, vol. 49, no. 1, pp. 253–260, 2013.

[69] J. H. Seo, H. Shim, and J. Back, "Consensus of high-order linear systems using dynamic output feedback compensator: Low gain approach," *Automatica*, vol. 45, no. 11, pp. 2659–2664, 2009.

[70] Z. Lin, B. Francis, and M. Maggiore, "Necessary and sufficient graphical conditions for formation control of unicycles," *IEEE Transactions on Automatic Control*, vol. 50, no. 1, pp. 121–127, 2005.

[71] A. Abdessameud, A. Tayebi, and I. G. Polushin, "Attitude synchronization of multiple rigid bodies with communication delays," *IEEE Transactions on Automatic Control*, vol. 57, no. 9, pp. 2405–2411, 2012.

[72] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice Hall, Englewood Cliffs, NJ, 1989, vol. 23.

[73] J. Tsitsiklis and D. Bertsekas, "Distributed asynchronous optimal routing in data networks," *IEEE Transactions on Automatic Control*, vol. 31, no. 4, pp. 325–332, 1986.

[74] E. Nuno, R. Ortega, L. Basanez, and D. Hill, "Synchronization of networks of nonidentical Euler–Lagrange systems with uncertain parameters and communication delays," *IEEE Transactions on Automatic Control*, vol. 56, no. 4, pp. 935–941, 2011.

[75] E. Nuno, I. Sarras, and L. Basanez, "Consensus in networks of nonidentical Euler–Lagrange systems using P+d controllers," *IEEE Transactions on Robotics*, vol. 29, no. 6, pp. 1503–1508, 2013.

[76] J. CortéS, "Finite-time convergent gradient flows with applications to network consensus," *Automatica*, vol. 42, no. 11, pp. 1993–2000, 2006.

[77] U. Munz, A. Papachristodoulou, and F. Allgower, "Robust consensus controller design for nonlinear relative degree two multi-agent systems with communication constraints," *IEEE Transactions on Automatic Control*, vol. 56, no. 1, pp. 145–151, 2011.

[78] L. Fang and P. J. Antsaklis, "Asynchronous consensus protocols using nonlinear paracontractions theory," *IEEE Transactions on Automatic Control*, vol. 53, no. 10, pp. 2351–2355, 2008.

[79] R. Sepulchre, "Consensus on nonlinear spaces," *Annual Reviews in Control*, vol. 35, no. 1, pp. 56–64, 2011.

[80] K. Krishnanand and D. Ghose, "Theoretical foundations for rendezvous of glowworm-inspired agent swarms at multiple locations," *Robotics and Autonomous Systems*, vol. 56, no. 7, pp. 549–569, 2008.

[81] H. Kopetz and W. Ochsenreiter, "Clock synchronization in distributed real-time systems," *IEEE Transactions on Computers*, vol. 100, no. 8, pp. 933–940, 1987.

[82] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: a survey," *Ad hoc Networks*, vol. 3, no. 3, pp. 281–323, 2005.

[83] C. Xu and F. C. Lau, *Load balancing in parallel computers: theory and practice*. Springer Science & Business Media, 1996, vol. 381.

[84] A. T. Kamal, J. A. Farrell, A. K. Roy-Chowdhury *et al.*, "Information weighted consensus filters and their application in distributed camera networks." *IEEE Transactions on Automatic Control*, vol. 58, no. 12, pp. 3112–3125, 2013.

[85] D. Devarajan, Z. Cheng, and R. J. Radke, "Calibrating distributed camera networks," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1625–1639, 2008.

[86] U. Niethammer, S. Rothmund, M. James, J. Travelletti, and M. Joswig, "Uav-based remote sensing of landslides," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 38, no. 5, pp. 496–501, 2010.

[87] N. Biggs, N. L. Biggs, and E. N. Biggs, *Algebraic graph theory*. Cambridge University Press, 1993, vol. 67.

[88] W. Ren and E. Atkins, "Distributed multi-vehicle coordinated control via local information exchange," *International Journal of Robust and Nonlinear Control*, vol. 17, no. 10-11, pp. 1002–1033, 2007.

[89] W. Ren, "Consensus algorithms for double-integrator dynamics," *IEEE Transactions on Automatic Control*, vol. 53, no. 6, pp. 1503–1509, 2008.

[90] Y. Hong, G. Chen, and L. Bushnell, "Distributed observers design for leader-following control of multi-agent networks," *Automatica*, vol. 44, no. 3, pp. 846–850, 2008.

[91] M. Park, O. Kwon, J. H. Park, S. a. Lee, and E. Cha, "Randomly changing leader-following consensus control for Markovian switching multi-agent systems with interval time-varying delays," *Nonlinear Analysis: Hybrid Systems*, vol. 12, pp. 117–131, 2014.

[92] Z. Meng, D. V. Dimarogonas, and K. H. Johansson, "Leader–follower coordinated tracking of multiple heterogeneous lagrange systems using continuous control," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 739–745, 2014.

[93] J. Back and J.-S. Kim, "A disturbance observer based practical coordinated tracking controller for uncertain heterogeneous multi-agent systems," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 14, pp. 2254–2278, 2015.

[94] S. Khoo, L. Xie, and Z. Man, "Robust finite-time consensus tracking algorithm for multirobot systems," *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 2, pp. 219–228, 2009.

[95] M. Egerstedt, S. Martini, M. Cao, K. Camlibel, and A. Bicchi, "Interacting with networks: How does structure relate to controllability in single-leader, consensus networks?" *IEEE Control Systems*, vol. 32, no. 4, pp. 66–73, 2012.

[96] C. D. Godsil, "Compact graphs and equitable partitions," *Linear Algebra and its Applications*, vol. 255, no. 1-3, pp. 259–266, 1997.

[97] S. Martini, M. Egerstedt, and A. Bicchi, "Controllability decompositions of networked systems through quotient graphs," in *47th IEEE Conference on Decision and Control (CDC 2008)*. IEEE, 2008, pp. 5244–5249.

[98] M. Zamani and H. Lin, "Structural controllability of multi-agent systems," in *American Control Conference (ACC'09)*. IEEE, 2009, pp. 5743–5748.

[99] Z. Ji, Z. Wang, H. Lin, and Z. Wang, "Controllability of multi-agent systems with time-delay in state and switching topology," *International Journal of Control*, vol. 83, no. 2, pp. 371–386, 2010.

[100] M. Pósfai, Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Effect of correlations on network controllability," *Scientific Reports*, vol. 3, p. 1067, 2013.

[101] J. Sun and A. E. Motter, "Controllability transition and nonlocality in network control," *Physical Review Letters*, vol. 110, no. 20, p. 208701, 2013.

[102] C.-L. Pu, W.-J. Pei, and A. Michaelson, "Robustness analysis of network controllability," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 18, pp. 4420–4425, 2012.

[103] A. Chapman, M. Nabi-Abdolyousefi, and M. Mesbahi, "Controllability and observability of network-of-networks via cartesian products," *IEEE Transactions on Automatic Control*, vol. 59, no. 10, pp. 2668–2679, 2014.

[104] X. Chen, S. Pequito, G. J. Pappas, and V. M. Preciado, "Minimal edge addition for network controllability," *IEEE Transactions on Control of Network Systems*, 2018.

[105] H. Su, X. Wang, and Z. Lin, "Flocking of multi-agents with a virtual leader," *IEEE Transactions on Automatic Control*, vol. 54, no. 2, pp. 293–307, 2009.

[106] H. G. Tanner, A. Jadbabaie, and G. J. Pappas, "Flocking in fixed and switching networks," *IEEE Transactions on Automatic control*, vol. 52, no. 5, pp. 863–868, 2007.

[107] M. Porfiri, D. G. Roberson, and D. J. Stilwell, "Tracking and formation control of multiple autonomous agents: A two-level consensus approach," *Automatica*, vol. 43, no. 8, pp. 1318–1328, 2007.

[108] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE Transactions on robotics and automation*, vol. 14, no. 6, pp. 926–939, 1998.

[109] M. A. Lewis and K.-H. Tan, "High precision formation control of mobile robots using virtual structures," *Autonomous robots*, vol. 4, no. 4, pp. 387–403, 1997.

[110] T. H. Van den Broek, N. van de Wouw, and H. Nijmeijer, "Formation control of unicycle mobile robots: a virtual structure approach," in *Proceedings of the 48th IEEE Conference on Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference (CDC/CCC 2009)*. IEEE, 2009, pp. 8328–8333.

[111] W. Dong and J. A. Farrell, "Cooperative control of multiple nonholonomic mobile agents," *IEEE Transactions on Automatic Control*, vol. 53, no. 6, pp. 1434–1448, 2008.

[112] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE transactions on Automatic Control*, vol. 49, no. 9, pp. 1465–1476, 2004.

[113] D. V. Dimarogonas and K. J. Kyriakopoulos, "A connection between formation infeasibility and velocity alignment in kinematic multi-agent systems," *Automatica*, vol. 44, no. 10, pp. 2648–2654, 2008.

[114] J. Cortés, "Global and robust formation-shape stabilization of relative sensing networks," *Automatica*, vol. 45, no. 12, pp. 2754–2762, 2009.

[115] A. Abdessameud and A. Tayebi, "On consensus algorithms for double-integrator dynamics without velocity measurements and with input constraints," *Systems & Control Letters*, vol. 59, no. 12, pp. 812–821, 2010.

[116] H. Rezaee and F. Abdollahi, "Pursuit formation of double-integrator dynamics using consensus control approach," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 7, pp. 4249–4256, 2015.

[117] S. E. Tuna, "LQR-based coupling gain for synchronization of linear systems," *arXiv preprint arXiv:0801.3390*, 2008.

[118] L. Consolini, F. Morbidi, D. Prattichizzo, and M. Tosques, "Leader–follower formation control of nonholonomic mobile robots with input constraints," *Automatica*, vol. 44, no. 5, pp. 1343–1349, 2008.

[119] R. Vidal, O. Shakernia, and S. Sastry, "Following the flock [formation control]," *IEEE Robotics & Automation Magazine*, vol. 11, no. 4, pp. 14–20, 2004.

[120] K.-K. Oh, M.-C. Park, and H.-S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, 2015.

[121] B. Hendrickson, "Conditions for unique graph realizations," *SIAM Journal on Computing*, vol. 21, no. 1, pp. 65–84, 1992.

[122] Z. Sun, B. D. Anderson, M. Deghat, and H.-S. Ahn, "Rigid formation control of double-integrator systems," *International Journal of Control*, vol. 90, no. 7, pp. 1403–1419, 2017.

[123] R. Olfati-Saber and R. M. Murray, "Graph rigidity and distributed formation stabilization of multi-vehicle systems," in *Proceedings of the 41st IEEE Conference on Decision and Control, 2002*, vol. 3. IEEE, 2002, pp. 2965–2971.

[124] M. Cao, C. Yu, and B. D. Anderson, "Formation control using range-only measurements," *Automatica*, vol. 47, no. 4, pp. 776–781, 2011.

[125] L. Krick, M. E. Broucke, and B. A. Francis, "Stabilisation of infinitesimally rigid formations of multi-robot networks," *International Journal of Control*, vol. 82, no. 3, pp. 423–439, 2009.

[126] Z. Lin, L. Wang, Z. Han, and M. Fu, "Distributed formation control of multi-agent systems using complex laplacian," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1765–1777, 2014.

[127] F. Dorfler and B. Francis, "Geometric analysis of the formation problem for autonomous robots," *IEEE Transactions on Automatic Control*, vol. 55, no. 10, pp. 2379–2384, 2010.

[128] T. Mylvaganam and A. Astolfi, "A differential game approach to formation control for a team of agents with one leader," in *American Control Conference (ACC, 2015)*. IEEE, 2015, pp. 1469–1474.

[129] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.

[130] A. Gusrialdi, S. Hirche, T. Hatanaka, and M. Fujita, "Voronoi based coverage control with anisotropic sensors," in *American Control Conference, 2008*. IEEE, 2008, pp. 736–741.

[131] I. I. Hussein and D. M. Stipanovic, "Effective coverage control for mobile sensor networks with guaranteed collision avoidance," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 4, pp. 642–657, 2007.

[132] L. C. Pimenta, V. Kumar, R. C. Mesquita, and G. A. Pereira, "Sensing and coverage for a network of heterogeneous robots," in *47th IEEE Conference on Decision and Control (CDC 2008)*. IEEE, 2008, pp. 3947–3952.

[133] K. Laventall and J. Cortés, "Coverage control by multi-robot networks with limited-range anisotropic sensory," *International Journal of Control*, vol. 82, no. 6, pp. 1113–1121, 2009.

[134] Y. Wang and I. I. Hussein, "Awareness coverage control over large-scale domains with intermittent communications," *IEEE Transactions on Automatic Control*, vol. 55, no. 8, pp. 1850–1859, 2010.

[135] R. A. Freeman, P. Yang, and K. M. Lynch, "Stability and convergence properties of dynamic average consensus estimators," in *45th IEEE Conference on Decision and Control, 2006*. IEEE, 2006, pp. 338–343.

[136] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.

[137] S. S. Kia, J. Cortés, and S. Martinez, "Dynamic average consensus under limited control authority and privacy requirements," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 13, pp. 1941–1966, 2015.

[138] S. Nosrati, M. Shafiee, and M. B. Menhaj, "Dynamic average consensus via nonlinear protocols," *Automatica*, vol. 48, no. 9, pp. 2262–2270, 2012.

[139] F. Chen, G. Feng, L. Liu, and W. Ren, "Distributed average tracking of networked Euler–Lagrange systems," *IEEE Transactions on Automatic Control*, vol. 60, no. 2, pp. 547–552, 2015.

[140] F. Chen, W. Ren, W. Lan, and G. Chen, "Distributed average tracking for reference signals with bounded accelerations," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 863–869, 2015.

[141] E. Montijano, J. I. Montijano, C. Sagüés, and S. Martínez, "Robust discrete time dynamic average consensus," *Automatica*, vol. 50, no. 12, pp. 3131–3138, 2014.

[142] F. Chen and W. Ren, "A connection between dynamic region-following formation control and distributed average tracking," *IEEE Transactions on Cybernetics*, 2017.

[143] S. Kar, J. M. Moura, and K. Ramanan, "Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication," *IEEE Transactions on Information Theory*, vol. 58, no. 6, pp. 3575–3605, 2012.

[144] H. T. Banks and K. Kunisch, *Estimation techniques for distributed parameter systems*. Springer Science & Business Media, 2012.

[145] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*. ACM, 2004, pp. 1–10.

[146] S. Sundhar Ram, A. Nedić, and V. V. Veeravalli, "A new class of distributed optimization algorithms: Application to regression of distributed data," *Optimization Methods and Software*, vol. 27, no. 1, pp. 71–88, 2012.

[147] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," in *44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference (CDC-ECC'05)*. IEEE, 2005, pp. 8179–8184.

[148] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *46th IEEE Conference on Decision and Control, 2007*. IEEE, 2007, pp. 5492–5498.

[149] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri, "Distributed Kalman filtering based on consensus strategies," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, 2008.

[150] R. Olfati-Saber, "Kalman-consensus filter: Optimality, stability, and performance," in *Proceedings of the 48th IEEE Conference on Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference (CDC/CCC 2009)*. IEEE, 2009, pp. 7036–7042.

[151] F. S. Cattivelli and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering and smoothing," *IEEE Transactions on Automatic Control*, vol. 55, no. 9, pp. 2069–2084, 2010.

[152] S. Kar and J. M. Moura, "Gossip and distributed Kalman filtering: Weak consensus under weak detectability," *IEEE Transactions on Signal Processing*, vol. 59, no. 4, pp. 1766–1784, 2011.

[153] H. Bai, R. A. Freeman, and K. M. Lynch, "Distributed Kalman filtering using the internal model average consensus estimator," in *American Control Conference (ACC, 2011)*. IEEE, 2011, pp. 1500–1505.

[154] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Approximate distributed Kalman filtering in sensor networks with quantifiable performance," in *Fourth International Symposium on Information Processing in Sensor Networks (IPSN 2005)*. IEEE, 2005, pp. 133–139.

[155] J. Cortés, "Distributed Kriged Kalman filter for spatial estimation," *IEEE Transactions on Automatic Control*, vol. 54, no. 12, pp. 2816–2827, 2009.

[156] M. Ji, G. Ferrari-Trecate, M. Egerstedt, and A. Buffa, "Containment control in mobile networks," *IEEE Transactions on Automatic Control*, vol. 53, no. 8, pp. 1972–1975, 2008.

[157] A. Bensoussan and J.-L. Menaldi, "Difference equations on weighted graphs," *Journal of Convex Analysis*, vol. 12, no. 1, pp. 13–44, 2005.

[158] D. V. Dimarogonas, M. Egerstedt, and K. J. Kyriakopoulos, "A leader-based containment control strategy for multiple unicycles," in *45th IEEE Conference on Decision and Control, 2006*. IEEE, 2006, pp. 5968–5973.

[159] S. J. Yoo, "Distributed adaptive containment control of uncertain nonlinear multi-agent systems in strict-feedback form," *Automatica*, vol. 49, no. 7, pp. 2145–2153, 2013.

[160] H. Haghshenas, M. A. Badamchizadeh, and M. Baradarannia, "Containment control of heterogeneous linear multi-agent systems," *Automatica*, vol. 54, pp. 210–216, 2015.

[161] F. Chen, W. Ren, and Y. Cao, "Surrounding control in cooperative agent networks," *Systems & Control Letters*, vol. 59, no. 11, pp. 704–712, 2010.

[162] Y. Lou and Y. Hong, "Distributed surrounding design of target region with complex adjacency matrices," *IEEE Transactions on Automatic Control*, vol. 60, no. 1, pp. 283–288, 2015.

[163] Y. Shi, R. Li, and K. L. Teo, "Cooperative enclosing control for multiple moving targets by a group of agents," *International Journal of Control*, vol. 88, no. 1, pp. 80–89, 2015.

[164] S. Shoja, M. Baradarannia, F. Hashemzadeh, M. Badamchizadeh, and P. Bagheri, "Surrounding control of nonlinear multi-agent systems with non-identical agents," *ISA Transactions*, vol. 70, pp. 219–227, 2017.

[165] Z. Chen and H.-T. Zhang, "No-beacon collective circular motion of jointly connected multi-agents," *Automatica*, vol. 47, no. 9, pp. 1929–1937, 2011.

[166] S. Rahili and W. Ren, "Distributed continuous-time convex optimization with time-varying cost functions," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1590–1605, 2017.

[167] M. Zhu and S. Martínez, "On distributed convex optimization under inequality and equality constraints," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 151–164, 2012.

[168] B. Gharesifard and J. Cortés, "Distributed continuous-time convex optimization on weight-balanced digraphs," *IEEE Transactions on Automatic Control*, vol. 59, no. 3, pp. 781–786, 2014.

[169] K. Srivastava and A. Nedic, "Distributed asynchronous constrained stochastic optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 772–790, 2011.

[170] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic control*, vol. 57, no. 3, pp. 592–606, 2012.

[171] J.-P. Richard, "Time-delay systems: an overview of some recent advances and open problems," *Automatica*, vol. 39, no. 10, pp. 1667–1694, 2003.

[172] R. Sipahi, S.-I. Niculescu, C. T. Abdallah, W. Michiels, and K. Gu, "Stability and stabilization of systems with time delay," *IEEE Control Systems*, vol. 31, no. 1, pp. 38–65, 2011.

[173] U. Munz, A. Papachristodoulou, and F. Allgower, "Consensus in multi-agent systems with coupling delays and switching topology," *IEEE Transactions on Automatic Control*, vol. 56, no. 12, pp. 2976–2982, 2011.

[174] A. Papachristodoulou, A. Jadbabaie, and U. Munz, "Effects of delay in multi-agent consensus and oscillator synchronization," *IEEE Transactions on Automatic Control*, vol. 55, no. 6, pp. 1471–1477, 2010.

[175] A. Seuret, D. V. Dimarogonas, and K. H. Johansson, "Consensus under communication delays," in *47th IEEE Conference on Decision and Control (CDC 2008)*. IEEE, 2008, pp. 4922–4927.

[176] R. Cepeda-Gomez and N. Olgac, "An exact method for the stability analysis of linear consensus protocols with time delay," *IEEE Transactions on Automatic Control*, vol. 56, no. 7, pp. 1734–1740, 2011.

[177] P.-A. Bliman and G. Ferrari-Trecate, "Average consensus problems in networks of agents with delayed communications," *Automatica*, vol. 44, no. 8, pp. 1985–1995, 2008.

[178] U. Münz, A. Papachristodoulou, and F. Allgöwer, "Delay robustness in consensus problems," *Automatica*, vol. 46, no. 8, pp. 1252–1265, 2010.

[179] D. F. Delchamps, "Stabilizing a linear system with quantized state feedback," *IEEE Transactions on Automatic Control*, vol. 35, no. 8, pp. 916–924, 1990.

[180] N. Elia and S. K. Mitter, "Stabilization of linear systems with limited information," *IEEE Transactions on Automatic Control*, vol. 46, no. 9, pp. 1384–1400, 2001.

[181] A. Kashyap, T. Başar, and R. Srikant, "Quantized consensus," *Automatica*, vol. 43, no. 7, pp. 1192–1203, 2007.

[182] R. Carli, F. Bullo, and S. Zampieri, "Quantized average consensus via dynamic coding/decoding schemes," *International Journal of Robust and Nonlinear Control*, vol. 20, no. 2, pp. 156–175, 2010.

[183] J. Lavaei and R. M. Murray, "Quantized consensus by means of gossip algorithm," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 19–32, 2012.

[184] K. Cai and H. Ishii, "Quantized consensus and averaging on gossip digraphs," *IEEE Transactions on Automatic Control*, vol. 56, no. 9, pp. 2087–2100, 2011.

[185] S. Kar and J. M. Moura, "Distributed consensus algorithms in sensor networks: Quantized data and random link failures," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1383–1400, 2010.

[186] M. Franceschelli, A. Giua, and C. Seatzu, "Quantized consensus in hamiltonian graphs," *Automatica*, vol. 47, no. 11, pp. 2495–2503, 2011.

[187] F. Ceragioli, C. De Persis, and P. Frasca, "Discontinuities and hysteresis in quantized average consensus," *Automatica*, vol. 47, no. 9, pp. 1916–1928, 2011.

[188] A. Speranzon, C. Fischione, B. Johansson, and K. H. Johansson, "Adaptive distributed estimation over wireless sensor networks with packet losses," in *46th IEEE Conference on Decision and Control, 2007*. IEEE, 2007, pp. 5472–5477.

[189] M. Huang and J. H. Manton, "Stochastic consensus seeking with noisy and directed inter-agent communication: Fixed and randomly varying topologies," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 235–241, 2010.

[190] Y.-H. Ni and X. Li, "Consensus seeking in multi-agent systems with multiplicative measurement noises," *Systems & Control Letters*, vol. 62, no. 5, pp. 430–437, 2013.

[191] D. Bauso, L. Giarré, and R. Pesenti, "Consensus for networks with unknown but bounded disturbances," *SIAM Journal on Control and Optimization*, vol. 48, no. 3, pp. 1756–1770, 2009.

[192] T. C. Aysal and K. E. Barner, "Convergence of consensus models with stochastic disturbances," *IEEE Transactions on Information Theory*, vol. 56, no. 8, pp. 4101–4113, 2010.

[193] M. Franceschelli, A. Pisano, A. Giua, and E. Usai, "Finite-time consensus with disturbance rejection by discontinuous local interactions in directed graphs," *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 1133–1138, 2015.

[194] R. Rajagopal and M. J. Wainwright, "Network-based consensus averaging with general noisy channels," *IEEE Transactions on Signal Processing*, vol. 59, no. 1, pp. 373–385, 2011.

[195] M. Franceschelli, A. Giua, A. Pisano, and E. Usai, "Finite-time consensus for switching network topologies with disturbances," *Nonlinear Analysis: Hybrid Systems*, vol. 10, pp. 83–93, 2013.

[196] J. A. Bondy, U. S. R. Murty *et al.*, *Graph theory with applications*. Citeseer, 1976, vol. 290.

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

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

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

[200] A. Ajorlou, A. Momeni, and A. G. Aghdam, "A class of bounded distributed control strategies for connectivity preservation in multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 55, no. 12, pp. 2828–2833, 2010.

[201] S. Bhattacharya, T. Başar, and N. Hovakimyan, "Singular surfaces in multi-agent connectivity maintenance games," in *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC'11)*. IEEE, 2011, pp. 261–266.

[202] L. Sabattini, N. Chopra, and C. Secchi, "Decentralized connectivity maintenance for cooperative control of mobile robotic systems," *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1411–1423, 2013.

[203] S. Bhattacharya and T. Başar, "Graph-theoretic approach for connectivity maintenance in mobile networks in the presence of a jammer," in *49th IEEE Conference on Decision and Control (CDC 2010)*. IEEE, 2010, pp. 3560–3565.

[204] P. Robuffo Giordano, A. Franchi, C. Secchi, and H. H. Bülthoff, "A passivity-based decentralized strategy for generalized connectivity maintenance," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 299–323, 2013.

[205] L. Sabattini, C. Secchi, and N. Chopra, "Decentralized connectivity maintenance for networked lagrangian dynamical systems," in *IEEE International Conference on Robotics and Automation (ICRA 2012)*. IEEE, 2012, pp. 2433–2438.

[206] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital control of dynamic systems*. Addison-Wesley, Menlo Park, CA, 1998, vol. 3.

[207] Y. Cao and W. Ren, "Multi-vehicle coordination for double-integrator dynamics under fixed undirected/directed interaction in a sampled-data setting," *International Journal of Robust and Nonlinear Control*, vol. 20, no. 9, pp. 987–1000, 2010.

[208] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson, "Distributed event-triggered control for multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1291–1297, 2012.

[209] Y. Fan, G. Feng, Y. Wang, and C. Song, "Distributed event-triggered control of multi-agent systems with combinational measurements," *Automatica*, vol. 49, no. 2, pp. 671–675, 2013.

[210] W. Lu, Y. Han, and T. Chen, "Synchronization in networks of linearly coupled dynamical systems via event-triggered diffusions," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 12, pp. 3060–3069, 2015.

[211] E. Garcia, Y. Cao, and D. W. Casbeer, "Decentralized event-triggered consensus with general linear dynamics," *Automatica*, vol. 50, no. 10, pp. 2633–2640, 2014.

[212] W. Zhu, Z.-P. Jiang, and G. Feng, "Event-based consensus of multi-agent systems with general linear models," *Automatica*, vol. 50, no. 2, pp. 552–558, 2014.

[213] G. Shi and K. H. Johansson, "Multi-agent robust consensus-part ii: Application to distributed event-triggered coordination," in *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC, 2011)*. IEEE, 2011, pp. 5738–5743.

[214] C. Nowzari and J. Cortés, "Zeno-free, distributed event-triggered communication and control for multi-agent average consensus," in *American Control Conference (ACC, 2014)*. IEEE, 2014, pp. 2148–2153.

[215] R. Aragues, J. Cortes, and C. Sagues, "Distributed consensus on robot networks for dynamically merging feature-based maps," *IEEE Transactions on Robotics*, vol. 28, no. 4, pp. 840–854, 2012.

[216] Y. Fan, L. Liu, G. Feng, and Y. Wang, "Self-triggered consensus for multi-agent systems with zeno-free triggers," *IEEE Transactions on Automatic Control*, vol. 60, no. 10, pp. 2779–2784, 2015.

[217] H. Yu and P. J. Antsaklis, "Output synchronization of networked passive systems with event-driven communication," *IEEE Transactions on Automatic Control*, vol. 59, no. 3, pp. 750–756, 2014.

[218] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri, "A pi consensus controller for networked clocks synchronization," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 10 289–10 294, 2008.

[219] D. A. B. Lombana and M. Di Bernardo, "Distributed pid control for consensus of homogeneous and heterogeneous networks," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 2, pp. 154–163, 2015.

[220] D. Burbano and M. di Bernardo, "Consensus and synchronization of complex networks via proportional-integral coupling," in *IEEE International Symposium on Circuits and Systems (ISCAS 2014)*. IEEE, 2014, pp. 1796–1799.

[221] P. De Lellis, M. Di Bernardo, F. Sorrentino, and A. Tierno, "Adaptive synchronization of complex networks," *International Journal of Computer Mathematics*, vol. 85, no. 8, pp. 1189–1218, 2008.

[222] L. Cheng, Z.-G. Hou, M. Tan, Y. Lin, and W. Zhang, "Neural-network-based adaptive leader-following control for multiagent systems with uncertainties," *IEEE Transactions on Neural Networks*, vol. 21, no. 8, pp. 1351–1358, 2010.

[223] M. M. Polycarpou, "Stable adaptive neural control scheme for nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 41, no. 3, pp. 447–451, 1996.

[224] W. Chen, X. Li, W. Ren, and C. Wen, "Adaptive consensus of multi-agent systems with unknown identical control directions based on a novel Nussbaum-type function," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1887–1892, 2014.

[225] G. Droge and M. Egerstedt, "Distributed parameterized model predictive control of networked multi-agent systems," in *American Control Conference (ACC, 2013)*. IEEE, 2013, pp. 1332–1337.

[226] H. Terelius, U. Topcu, and R. Murray, "Decentralized multi-agent optimization via dual decomposition," in *18th IFAC World Congress, 28 August 2011 through 2 September 2011, Milano, Italy*, 2011, pp. 11 245–11 251.

[227] A. Rantzer, "Dynamic dual decomposition for distributed control," in *American Control Conference (ACC'09)*. IEEE, 2009, pp. 884–888.

[228] N. Chopra and M. W. Spong, "Passivity-based control of multi-agent systems," *Advances in Robot Control: From Everyday Physics to Human-like Movements*, vol. 107, p. 134, 2006.

[229] M. Arcak, "Passivity as a design tool for group coordination," *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1380–1390, 2007.

[230] Y. Igarashi, T. Hatanaka, M. Fujita, and M. W. Spong, "Passivity-based attitude synchronization in $se(3)$," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1119–1134, 2009.

[231] H. Wang, "Passivity based synchronization for networked robotic systems with uncertain kinematics and dynamics," *Automatica*, vol. 49, no. 3, pp. 755–761, 2013.

[232] H. Yu and P. J. Antsaklis, "Passivity-based output synchronization of networked Euler–Lagrange systems subject to nonholonomic constraints," in *American Control Conference (ACC, 2010)*. IEEE, 2010, pp. 208–213.

[233] S. Knorn, A. Donaire, J. C. Agüero, and R. H. Middleton, "Passivity-based control for multi-vehicle systems subject to string constraints," *Automatica*, vol. 50, no. 12, pp. 3224–3230, 2014.

[234] V. Gazi, "Swarm aggregations using artificial potentials and sliding-mode control," *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1208–1214, 2005.

[235] L. Wang and F. Xiao, "Finite-time consensus problems for networks of dynamic agents," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 950–955, 2010.

[236] Y. Cao and W. Ren, "Finite-time consensus for multi-agent networks with unknown inherent nonlinear dynamics," *Automatica*, vol. 50, no. 10, pp. 2648–2656, 2014.

[237] S. Parsegov, A. Polyakov, and P. Shcherbakov, "Fixed-time consensus algorithm for multi-agent systems with integrator dynamics," *IFAC Proceedings Volumes*, vol. 46, no. 27, pp. 110–115, 2013.

[238] Q. Hui, W. M. Haddad, and S. P. Bhat, "Finite-time semistability and consensus for nonlinear dynamical networks," *IEEE Transactions on Automatic Control*, vol. 53, no. 8, pp. 1887–1900, 2008.

[239] S. Sundaram and C. N. Hadjicostis, "Finite-time distributed consensus in graphs with time-invariant topologies," in *American Control Conference (ACC'07)*. IEEE, 2007, pp. 711–716.

[240] J. M. Hendrickx, G. Shi, and K. H. Johansson, "Finite-time consensus using stochastic matrices with positive diagonals," *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 1070–1073, 2015.

[241] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE transactions on robotics*, vol. 25, no. 4, pp. 912–926, 2009.

[242] M. Rubenstein, A. Cabrera, J. Werfel, G. Habibi, J. McLurkin, and R. Nagpal, "Collective transport of complex objects by simple robots: theory and experiments," in *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 47–54.

[243] I. D. Couzin, J. Krause, N. R. Franks, and S. A. Levin, "Effective leadership and decision-making in animal groups on the move," *Nature*, vol. 433, no. 7025, pp. 513–516, 2005.

[244] I. R. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, and S. Burion, "Human-robot teaming for search and rescue," *IEEE Pervasive Computing*, vol. 4, no. 1, pp. 72–79, 2005.

[245] J. S. Jennings, G. Whelan, and W. F. Evans, "Cooperative search and rescue with a team of mobile robots," in *8th International Conference on Advanced Robotics (ICAR'97)*. IEEE, 1997, pp. 193–200.

[246] P. Song and V. Kumar, "A potential field based approach to multi-robot manipulation," in *IEEE International Conference on Robotics and Automation (ICRA'02)*, vol. 2. IEEE, 2002, pp. 1217–1222.

[247] J. Wawerla and R. T. Vaughan, "A fast and frugal method for team-task allocation in a multi-robot transportation system," in *IEEE International Conference on Robotics and Automation (ICRA, 2010)*. IEEE, 2010, pp. 1432–1437.

[248] L. Schenato and F. Fiorentin, "Average timesynch: A consensus-based protocol for clock synchronization in wireless sensor networks," *Automatica*, vol. 47, no. 9, pp. 1878–1886, 2011.

[249] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*. IEEE Press, 2005, p. 9.

[250] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Distributed sensor fusion using dynamic consensus," in *IFAC World Congress*, Prague Czech Republic, 2005.

[251] A. Simonetto and G. Leus, "Distributed maximum likelihood sensor network localization." *IEEE Transactions on Signal Processing*, vol. 62, no. 6, pp. 1424–1437, 2014.

[252] Z. Wang, S. Zheng, S. Boyd, and Y. Ye, "Further relaxations of the SDP approach to sensor network localization," *SIAM Journal on Optimization*, vol. 19, no. 2, pp. 655–673, 2008.

[253] S. Boyd, "Alternating direction method of multipliers," in *Talk at NIPS Workshop on Optimization and Machine Learning*, 2011.

[254] M. K. Maggs, S. G. O'keefe, and D. V. Thiel, "Consensus clock synchronization for wireless sensor networks," *IEEE Sensors Journal*, vol. 12, no. 6, pp. 2269–2277, 2012.

[255] R. Carli and S. Zampieri, "Network clock synchronization based on the second-order linear consensus algorithm," *IEEE Transactions on Automatic Control*, vol. 59, no. 2, pp. 409–422, 2014.

[256] R. Tron and R. Vidal, "Distributed image-based 3-D localization of camera sensor networks," in *Proceedings of the 48th IEEE Conference on Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference (CDC/CCC 2009)*. IEEE, 2009, pp. 901–908.

[257] A. J. Wood and B. F. Wollenberg, *Power generation, operation, and control*. John Wiley & Sons, 2012.

[258] S. Yang, S. Tan, and J.-X. Xu, "Consensus based approach for economic dispatch problem in a smart grid," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4416–4426, 2013.

[259] S. Kar, G. Hug, J. Mohammadi, and J. M. Moura, "Distributed state estimation and energy management in smart grids: A consensus + innovations approach," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 6, pp. 1022–1038, 2014.

[260] G. Mateos, I. D. Schizas, and G. B. Giannakis, "Distributed recursive least-squares for consensus-based in-network adaptive estimation," *IEEE Transactions on Signal Processing*, vol. 57, no. 11, pp. 4583–4588, 2009.

[261] F. Dorfler and F. Bullo, "Synchronization and transient stability in power networks and nonuniform Kuramoto oscillators," *SIAM Journal on Control and Optimization*, vol. 50, no. 3, pp. 1616–1642, 2012.

[262] D. Romeres, F. Dorfler, and F. Bullo, "Novel results on slow coherency in consensus and power networks," in *European Control Conference (ECC, 2013)*. IEEE, 2013, pp. 742–747.

[263] Z. Zhang and M.-Y. Chow, "Incremental cost consensus algorithm in a smart grid environment," in *IEEE Power and Energy Society General Meeting, 2011*. IEEE, 2011, pp. 1–6.

[264] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *Journal of Machine Learning Research*, vol. 11, no. May, pp. 1663–1707, 2010.

[265] R. Tron and R. Vidal, "Distributed computer vision algorithms through distributed averaging," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR, 2011)*. IEEE, 2011, pp. 57–63.

[266] M. H. DeGroot, "Reaching a consensus," *Journal of the American Statistical Association*, vol. 69, no. 345, pp. 118–121, 1974.

[267] N. Fredkin and E. Johnson, "Social influence networks and opinion change," *Adv Group Process*, vol. 16, pp. 1–29, 1999.

[268] V. Amelkin, F. Bullo, and A. K. Singh, "Polar opinion dynamics in social networks," *IEEE Transactions on Automatic Control*, 2017.

[269] D. Acemoglu and A. Ozdaglar, "Opinion dynamics and learning in social networks," *Dynamic Games and Applications*, vol. 1, no. 1, pp. 3–49, 2011.

[270] C. Altafini, "Consensus problems on networks with antagonistic interactions," *IEEE Transactions on Automatic Control*, vol. 58, no. 4, pp. 935–946, 2013.

[271] A. Jadbabaie, P. Molavi, A. Sandroni, and A. Tahbaz-Salehi, "Non-bayesian social learning," *Games and Economic Behavior*, vol. 76, no. 1, pp. 210–225, 2012.

[272] D. Acemoglu, A. Ozdaglar, and A. ParandehGheibi, "Spread of (mis) information in social networks," *Games and Economic Behavior*, vol. 70, no. 2, pp. 194–227, 2010.

[273] Z. Liu, X. Huang, S. X.-D. Tan, H. Wang, and H. Tang, "Distributed task migration for thermal hot spot reduction in many-core microprocessors," in *10th International Conference on ASIC (ASICON, 2013)*. IEEE, 2013, pp. 1–4.

[274] M. U. Sardar, O. Hasan, M. Shafique, and J. Henkel, "Theorem proving based formal verification of distributed dynamic thermal management schemes," *Journal of Parallel and Distributed Computing*, vol. 100, pp. 157–171, 2017.

[275] S. A. A. Bukhari, F. K. Lodhi, O. Hasan, M. Shafique, and J. Henkel, "FAMe-TM: Formal analysis methodology for task migration algorithms in many-core systems," *Science of Computer Programming*, vol. 133, pp. 154–174, 2017.

[276] Y. Yang, Q.-S. Jia, and X. Guan, "Stochastic coordination of aggregated electric vehicle charging with on-site wind power at multiple buildings," in *IEEE 56th Annual Conference on Decision and Control (CDC, 2017)*. IEEE, 2017, pp. 4434–4439.

[277] T. Hatanaka, X. Zhang, W. Shi, M. Zhu, and N. Li, "Physics-integrated hierarchical/distributed HVAC optimization for multiple buildings with robustness against time delays," in *IEEE 56th Annual Conference on Decision and Control*. IEEE, 2017, pp. 6573–6579.

[278] T. Hatanaka, X. Zhang, W. Shi, M. Zhu, and N. Li, "Physics-integrated hierarchical/distributed HVAC optimization for multiple buildings with robustness against time delays," in *IEEE 56th Annual Conference on Decision and Control (CDC, 2017)*. IEEE, 2017, pp. 6573–6579.

[279] T. Mylvaganam, M. Sassano, and A. Astolfi, "Constructive ϵ-Nash equilibria for nonzero-sum differential games," *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 950–965, 2015.

[280] E. Semsar-Kazerooni and K. Khorasani, "Multi-agent team cooperation: A game theory approach," *Automatica*, vol. 45, no. 10, pp. 2205–2213, 2009.

[281] K. G. Vamvoudakis, F. L. Lewis, and G. R. Hudas, "Multi-agent differential graphical games: Online adaptive learning solution for synchronization with optimality," *Automatica*, vol. 48, no. 8, pp. 1598–1611, 2012.

[282] C.-T. Lin, "Structural controllability," *IEEE Transactions on Automatic Control*, vol. 19, no. 3, pp. 201–208, 1974.

[283] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Controllability of complex networks," *Nature*, vol. 473, no. 7346, p. 167, 2011.

[284] S. Gu, F. Pasqualetti, M. Cieslak, Q. K. Telesford, B. Y. Alfred, A. E. Kahn, J. D. Medaglia, J. M. Vettel, M. B. Miller, S. T. Grafton *et al.*, "Controllability of structural brain networks," *Nature Communications*, vol. 6, p. 8414, 2015.

[285] T.-S. Tay and W. Whiteley, "Generating isostatic frameworks," *Structural Topology, 1985, Núm. 11.*

[286] W. Whiteley, "Some matroids from discrete applied geometry," *Contemporary Mathematics*, vol. 197, pp. 171–312, 1996.

[287] T.-S. Tay and W. Whiteley, "Recent advances in the generic rigidity of structures," *Structural Topology, 1984, Núm. 9.*

[288] T.-S. Tay, "A new proof of Laman's theorem," *Graphs and combinatorics*, vol. 9, no. 2, pp. 365–370, 1993.

[289] A. Recski, "A network theory approach to the rigidity of skeletal structures part II. Laman's theorem and topological formulae," *Discrete Applied Mathematics*, vol. 8, no. 1, pp. 63–68, 1984.

[290] J. Choi, S. Oh, and R. Horowitz, "Distributed learning and cooperative control for multi-agent systems," *Automatica*, vol. 45, no. 12, pp. 2802–2814, 2009.

[291] A. H. Sayed *et al.*, "Adaptation, learning, and optimization over networks," *Foundations and Trends® in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, 2014.

[292] S. Kapetanakis and D. Kudenko, "Reinforcement learning of coordination in cooperative multi-agent systems," *AAAI/IAAI*, vol. 2002, pp. 326–331, 2002.